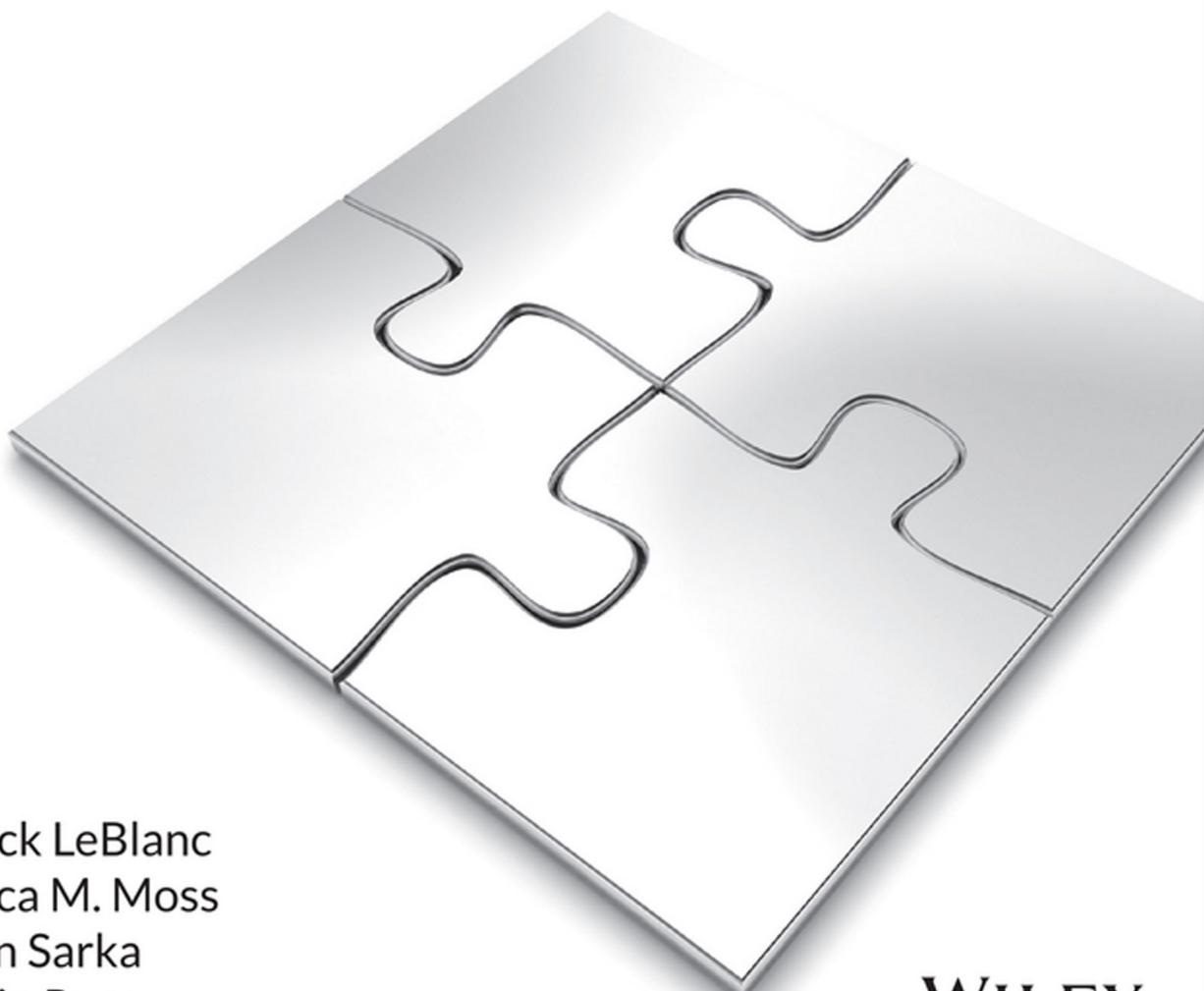


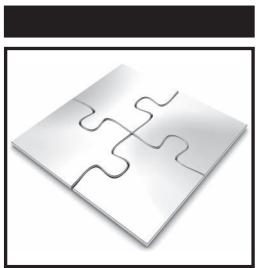
Applied Microsoft® Business Intelligence



Patrick LeBlanc
Jessica M. Moss
Dejan Sarka
Dustin Ryan

WILEY

Applied Microsoft® Business Intelligence



Applied Microsoft® Business Intelligence

Patrick LeBlanc
Jessica M. Moss
Dejan Sarka
Dustin Ryan

WILEY

Applied Microsoft® Business Intelligence

Published by

John Wiley & Sons, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Copyright © 2015 by John Wiley & Sons, Inc., Indianapolis, Indiana
Published simultaneously in Canada

ISBN: 978-1-118-96177-3
ISBN: 978-1-118-96179-7 (ebk)
ISBN: 978-1-118-96178-0 (ebk)

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or website may provide or recommendations it may make. Further, readers should be aware that Internet websites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services please contact our Customer Care Department within the United States at (877) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2015939526

Trademarks: Wiley and the Wiley logo are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. Microsoft is a registered trademark of Microsoft Corporation. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

Credits

Executive Editor

Robert Elliott

Project Editor

Maureen Tullis

Technical Editor

Julie Koesmarno

Production Manager

Kathleen Wisor

Copy Editor

Scott Tullis

**Manager of Content Development
& Assembly**

Mary Beth Wakefield

Marketing Director

David Mayhew

Marketing Manager

Carrie Sherrill

Professional Technology &**Strategy Director**

Barry Pruitt

Business Manager

Amy Knies

Associate Publisher

Jim Minatel

Project Coordinator, Cover

Brent Savage

Proofreader

Nancy Carrasco

Indexer

Johnna VanHoose Dinse

Cover Designer

Wiley

Cover Image

©iStock.com/alexsi



About the Authors

Patrick LeBlanc is a Microsoft SQL Server and Business Intelligence Technical Solution Professional. He has written several blogs and articles on his blog at <http://patrickleblanc.com>, www.sqlservercentral.com and www.bidn.com. Along with his 10+ years of experience, he holds a masters of science degree from Louisiana State University. He is the author and coauthor of four SQL Server books. His past work experiences include Sr. Consultant at Pragmatic Works and Database Architect at several companies. Prior to joining Microsoft he was awarded the Microsoft MVP award for his contributions to the community.

Jessica M. Moss, a Microsoft SQL Server MVP, is a well-known practitioner, author, and speaker in Microsoft SQL Server business intelligence. She has created numerous data warehousing solutions for companies in the retail, Internet, health services, finance, and energy industries. She has also authored technical content for multiple magazines, websites, and five technical books. Jessica enjoys working with the central Virginia community and speaks regularly at user groups, code camps, and conferences. You can read about her work on her blog, <http://www.jessicamoss.com>.

Dejan Sarka, MCT and SQL Server MVP, focuses on development of database and business intelligence applications. Besides projects, he spends about half his time training and mentoring colleagues. He is the founder of the Slovenian SQL Server and .NET Users Group. He has authored or coauthored 13 books about databases and SQL Server, as well as developed many courses and seminars for Microsoft, SolidQ, and Pluralsight.

Dustin Ryan is a Senior Business Intelligence Consultant and Trainer for Pragmatic. He has worked in the business intelligence field since 2008, has spoken at community events such as SQL Saturday, SQL Rally, and PASS Summit, and has a wide range of experience using the Microsoft business intelligence stack of products across multiple industries. Dustin resides in Jacksonville, Florida with his wife, three children, and a three-legged cat, and enjoys spending time with his family and serving at his local church. You can learn more about him at <http://SQLDusty.wordpress.com>.



About the Technical Editor

Julie Koesmarno, Microsoft MVP in SQL Server, MCSE Data Platform, and MCSE Business Intelligence, is an international SQL Server and business intelligence consultant with a bachelor's degree in IT. She has more than 10 years of experience working with SQL Server for large-scale and multi-million-dollar OLTP and ETL projects as a SQL Server developer and system analyst/designer. Her recent projects include delivering SQL Server 2012 data warehouse and business intelligence solutions for a number of high-profile clients in the U.S. and Australia. She is a blogger at MSSQLGirl.com and an experienced speaker at various well-known international SQL Server conferences in the U.S., Australia, and Europe, such as PASS Summit, PASS Rally, and PASS Business Analytics Conference. She also speaks at various local events such as SQL Saturday and User Groups. She currently leads the PASS Data Warehousing/Business Intelligence Virtual Chapter.

Acknowledgments

I would like to start by thanking God and my family. My family—what can I say? My wife always finds a way to motivate me and keep me going throughout each book as well as my entire career. She constantly encourages and reminds me that mediocracy is not part of the LeBlanc family. So, how can I fail? Next, I would like to thank my two kids. They always do or say funny things that give me the lift I need when I just want to give up. I also want to thank my coauthors on the book; this could not have happened without them. Finally, I would like to thank my great technical editor and content editors who caught all my mistakes and grammatical errors. Thanks everyone.

— Patrick LeBlanc

Thanks to:

- Rich, my family, and friends, for letting me disappear for days on end while writing.
- Patrick, Dejan, and Dustin, my amazing coauthors who made this book possible.
- Julie, our technical editor who provided wonderful feedback.
- Bob Elliott, whose editing guidance will never be forgotten by hundreds of authors.
- Maureen and Jim, whose editing and management made this book possible.

— Jessica M. Moss

Thanks to:

- My family and friends for understanding my lack of time during this engagement.
 - Maureen Tullis, the Project Editor for this book, for her invaluable help especially with my first steps in the project.
 - Coauthors for letting me be a part of this incredible group.
 - Technical editor, Julie, for her meticulous work.
- **Dejan Sarka**

Thanks to:

- My Lord and Savior, Jesus Christ, and for His perfect grace and mercy.
- My loving and forgiving wife, Angela, whose support and understanding during the late nights and long weekends made this book possible. Thanks, babe.
- My amazing children, Dallas, Bradley, and Andrew, whose love and beautiful smiles make all this worthwhile.
- My parents, Terry and Sybil, for your prayers, love, and encouragement.
- Brian Knight, who believed in me and gave me my first shot.
- Patrick and Maureen, for your support and hard work during this endeavor.

— **Dustin Ryan**

Contents at Glance

Introduction	xiii
Part I Overview of the Microsoft Business Intelligence Toolset	1
Chapter 1 Which Analysis and Reporting Tools Do You Need?	3
Chapter 2 Designing an Effective Business Intelligence Architecture	21
Chapter 3 Selecting the Data Architecture that Fits Your Organization	33
Part II Business Intelligence for Analysis	49
Chapter 4 Searching and Combining Data with Power Query	51
Chapter 5 Choosing the Right Business Intelligence Semantic Model	73
Chapter 6 Discovering and Analyzing Data with Power Pivot	89
Chapter 7 Developing a Flexible and Scalable Tabular Model	127
Chapter 8 Developing a Flexible and Scalable Multidimensional Model	151
Chapter 9 Discovering Knowledge with Data Mining	173
Part III Business Intelligence for Reporting	209
Chapter 10 Choosing the Right Business Intelligence Visualization Tool	211
Chapter 11 Designing Operational Reports with Reporting Services	227
Chapter 12 Visualizing Your Data Interactively with Power View	245
Chapter 13 Exploring Geographic and Temporal Data with Power Map	273
Chapter 14 Monitoring Your Business with PerformancePoint Services	293

Part IV	Deploying and Managing the Business Intelligence Solution	329
Chapter 15	Implementing a Self-Service Delivery Framework	331
Chapter 16	Designing and Implementing a Deployment Plan	351
Chapter 17	Managing and Maintaining the Business Intelligence Environment	363
Chapter 18	Scaling the Business Intelligence Environment	379
Index		391

Contents

Introduction	xiii
Part I Overview of the Microsoft Business Intelligence Toolset	1
Chapter 1 Which Analysis and Reporting Tools Do You Need?	3
Selecting a SQL Server Database Engine	4
Building a Data Warehouse	4
Selecting an RDBMS	5
Selecting SQL Server Analysis Services	6
Working with SQL Server Reporting Services	7
Understanding Operational Reports	8
Understanding Ad Hoc Reporting	10
Working with SharePoint	11
Working with Performance Point	12
Using Excel for Business Intelligence	14
What Is Power Query?	14
What Is Power Pivot?	14
What Is Power View?	14
Power Map	15
Which Development Tools Do You Need?	16
Using SQL Server Data Tools	16
Using SQL Management Studio	17
Using Dashboard Designer	18
Using Report Builder	19
Summary	20
Chapter 2 Designing an Effective Business Intelligence Architecture	21
Identifying the Audience and Goal of the Business Intelligence Solution	21
Who's the Audience?	22
What Is the Goal(s)?	23

What Are the Data Sources?	23
Using Internal Data Sources	23
Using External Data Sources	24
Using a Data Warehouse (or Not)	24
Implementing and Enforcing Data Governance	26
Planning an Analytical Model	28
Planning the Business Intelligence Delivery Solution	29
Considering Performance	30
Considering Availability	31
Summary	32
Chapter 3 Selecting the Data Architecture that Fits Your Organization	33
Why Is Data Architecture Selection Important?	34
Challenges	34
Benefits	35
How Do You Pick the Right Data Architecture?	36
Understanding Architecture Options	36
Understanding Research Selection Factors	42
Interviewing Key Stakeholders	44
Completing the Selection Form	45
Finalizing and Approving the Architecture	46
Summary	48
Part II Business Intelligence for Analysis	49
Chapter 4 Searching and Combining Data with Power Query	51
Downloading and Installing Power Query	52
Importing Data	56
Importing from a Database	57
Importing from the Web	59
Importing from a File	61
Transforming Data	62
Combining Data from Multiple Sources	62
Splitting Data	64
Aggregating Data	66
Introducing M Programming	70
A Glance at the M Language	70
Adding and Removing Columns Using M	72
Summary	72
Chapter 5 Choosing the Right Business Intelligence Semantic Model	73
Understanding the Business Intelligence Semantic Model Architecture	74
Understanding the Data Access Layer	75
Using Power Pivot	77
Using the Multidimensional Model	78
Using the Tabular Model	78

Implementing Query Languages and the Business Logic Layer	79
Data Analytics Expressions (DAX)	79
Multidimensional Expressions (MDX)	81
Direct Query and ROLAP	81
Data Model Layer	82
Comparing the Different Types of Models	83
Which Model Fits Your Organization?	84
Departmental	84
Team	86
Organizational	87
Summary	88
Chapter 6 Discovering and Analyzing Data with Power Pivot	89
Understanding Hardware and Software Requirements	90
Enabling Power Pivot	90
Designing an Optimal Power Pivot Model	92
Importing Only What You Need	92
Understanding Why Data Types Matter	99
Working with Columns or DAX Calculated Measures	103
Optimizing the Power Pivot Model for Reporting	104
Understanding Power Pivot Model Basics	104
Adding All Necessary Relationships	107
Adding Calculated Columns and DAX Measures	114
Creating Hierarchies and Key Performance Indicators (KPIs)	118
Sorting Your Data to Meet End-User Needs	121
Implementing Role-Playing Dimensions	122
Summary	125
Chapter 7 Developing a Flexible and Scalable Tabular Model	127
Why Use a Tabular Model?	127
Understanding the Tabular Model	128
Using the Tabular Model	128
Comparing the Tabular and Multidimensional Models	130
Understanding the Tabular Development Process	130
How Do You Design the Model?	131
Importing Data	131
Designing Relationships	134
Calculated Columns and Measures	135
How Do You Enhance the Model?	137
Adding Hierarchies	137
Designing Perspectives	140
Adding Partitions	141
How Do You Tune the Model?	144
Optimizing Processing	144
Optimizing Querying	147
Summary	149

Chapter 8	Developing a Flexible and Scalable Multidimensional Model	151
Why Use a Multidimensional Model?		151
Understanding the Multidimensional Model		152
Understanding the Multidimensional Model Process		153
How Do You Design the Model?		153
Creating Data Sources and the Data Source View		153
Using the Cube Creation Wizard		156
Adjusting Measures		159
Completing Dimensions		160
How Do You Enhance the Model?		162
Adding Navigation with Hierarchies		162
Using the Business Intelligence Wizard for Calculations		164
Using Partitions and Aggregations		166
How Do You Tune the Model?		169
Resolving Processing Issues		169
Querying		171
Summary		172
Chapter 9	Discovering Knowledge with Data Mining	173
Understanding the Business Value of Data Mining		174
Understanding Data Mining Techniques		174
Common Business Use Cases		175
Driving Decisions, Strategies, and Processes Through Data Mining		176
Getting the Basics Right		179
Understanding the Data		180
Training and Test Datasets		182
Defining the Data Mining Structure		184
The Data Mining Model		184
Applying the Microsoft Data Mining Techniques with Best Practices		185
Using Microsoft Association Rules		186
Grouping Data with Microsoft Clustering		190
Building Mining Models with Microsoft Naïve Bayes		192
Using the Microsoft Decision Trees		193
Using Microsoft Neural Network and Microsoft Logistic Regression		195
Using Microsoft Linear Regression and Microsoft Regression Trees		197
Microsoft Sequence Clustering		199
Forecasting with Microsoft Time Series		200
Developing and Deploying a Scalable and Extensible Data Mining Solution		201
Choosing Between a Relational or a Cube Source for Your Data Mining Structure		202
Deploying Data Mining Models		202
Using DMX to Query Data Mining Models		204

Maintaining Data Mining Models	205
Fine-Tuning the Data Mining Structure	205
Keeping the Data Model Relevant	205
Continuous Learning Cycle	205
Integrating Data Mining with Your BI Solution	206
Integrating Data Mining in Your DW and ETL Processes	206
Integrating Data Mining with Reporting Services	207
Data Mining in Excel	207
Summary	208
Part III Business Intelligence for Reporting	209
Chapter 10 Choosing the Right Business Intelligence Visualization Tool	211
Why Do You Need to Choose?	211
Identifying Users	212
Selecting Tools	213
What Are the Selection Criteria?	213
Business Capabilities	214
Technical Capabilities	214
How Do You Gather the Necessary Information?	215
What Are the Business Intelligence Visualization Options?	215
Using SQL Server Reporting Services	215
Using Power View	218
Using Power Map	219
How Do You Create and Complete the Evaluation Matrix?	221
How Do You Verify and Complete the Process?	223
Evaluation Matrix #1	224
Evaluation Matrix #2	224
Summary	225
Chapter 11 Designing Operational Reports with Reporting Services	227
What Are Operational Reports and Reporting Services?	227
Understanding Analytical versus Operational Reports	228
Using Reporting Services	228
What Are Development Best Practices?	230
Using Source and Version Control	231
Using Shared Data Sources and Datasets	234
Creating Templates	236
What Are Performance Best Practices?	237
Investigating Performance	237
Performance Tuning	238
What Are Functionality Best Practices?	239
Using Visualizations	239
Using Filters and Parameters	240
Providing Drilldown and Drillthrough	241
Summary	244

Chapter 12	Visualizing Your Data Interactively with Power View	245
	Where Does Power View Fit with Your Reporting Solution?	246
	Power View System Requirements	246
	Creating Power View Data Source Connections	247
	Creating Data Sources Inside Excel	247
	Creating Data Sources Inside SharePoint	249
	Creating Power View Reports	251
	Using SharePoint to Create Power View Reports	251
	Using Multiple Views in Power View	252
	Creating Power View Visualizations	253
	Creating Tables	253
	Converting Visualizations	254
	Creating Matrices	255
	Creating Charts	256
	Creating Multiples	261
	Creating Cards	261
	Creating Maps	262
	Using Excel to Create Power View Reports	263
	Filtering Data with Power View	264
	Adding Filters	264
	Using Advanced Filters	266
	Adding Slicers	266
	Invoking Cross-Filters	267
	Adding Tiles	268
	Adding Filters to a Report URL	270
	Exporting Power View Reports	271
	Summary	272
Chapter 13	Exploring Geographic and Temporal Data with Power Map	273
	How Power Map Fits into Reporting Solutions	274
	Understanding Power Map Features and Advantages	274
	Comparing Power Map to Other SQL Server Geospatial Reporting Tools	275
	Understanding Power Map Requirements	279
	Optimizing Your Data Model for Power Map	280
	Using Tours, Scenes, and Layers in Power Map	280
	Defining Geography Fields in Your Data Model	282
	Defining Date and Time Fields in Your Data Model	283
	Working with Geospatial and Temporal Data	284
	Visualizing Data Aggregation	284
	Creating a Power Map Tour	285
	Visualizing Data Over Time with Rich Animations	288
	Deploying and Sharing Power Map Visualizations	290
	Sharing Power Map Tours	291
	Enhancing Power Map Deployment and Configurations in Office 365	291
	Summary	292

Chapter 14	Monitoring Your Business with PerformancePoint Services	293
Where Does PerformancePoint Services Fit with Your Reporting Solution?	294	
Understanding PPS Features	295	
When Is PPS the Right Choice?	298	
Implementing PPS Requirements for SharePoint	300	
Extending PPS Dashboards	301	
Adding PerformancePoint Time Intelligence	301	
Using Interactivity Features	304	
Adding Reporting Services Reports to PerformancePoint	311	
Extending Filters and KPIs	313	
Deployment Best Practices	317	
Following Best Practices for PerformancePoint Data Connections and Content Libraries	317	
Deploying Dashboards Across Dev, Test, and Production Environments	319	
Customizing PerformancePoint SharePoint Web Parts	321	
Security and Configuration Best Practices	325	
Configuring the Unattended Service Account in SharePoint	325	
Optimizing PerformancePoint Services Application Settings	326	
Summary	328	
Part IV	Deploying and Managing the Business Intelligence Solution	329
Chapter 15	Implementing a Self-Service Delivery Framework	331
Planning a Self-Service Delivery Framework	331	
Creating a Data Governance Plan for Enterprise, Team, and Personal BI	332	
Identifying Stakeholders, Subject Matter Experts, and Data Stewards	334	
Understanding Industry Compliance Considerations	334	
Managing Data Quality and Master Data	337	
Identifying Target Audience and Roles	339	
Developing a Training Plan	340	
Inventory Tools and Skillset	340	
Understanding Data Quality Services	340	
Understanding Master Data Services	342	
Managing Data Quality and Master Data in Excel	345	
Business Intelligence Features Across the Microsoft Data Platform Versions and Editions	347	
Defining Success Criteria	348	
Summary	349	
Chapter 16	Designing and Implementing a Deployment Plan	351
What Is a Deployment Plan?	351	
How Do You Deploy Business Intelligence Code?	353	
Using Analysis Services (Multidimensional or Tabular)	354	
Using Reporting Services	357	

How Do You Implement the Deployment Plan?	359
Planning the Deployment	359
Designing Scripts	360
Documenting Steps	360
Testing the Plan	361
Training Your Staff	362
Summary	362
Chapter 17 Managing and Maintaining the Business Intelligence Environment	363
Using SQL Server Reporting Services	363
Configuring Memory	365
Caching Data and Pre-Rendering Reports	368
Using ExecutionLog Views	369
Working with SQL Server Analysis Services	372
Using Multidimensional Models	372
Using Tabular Models	374
Using SharePoint to Improve Performance	375
Summary	378
Chapter 18 Scaling the Business Intelligence Environment	379
Why Would You Scale the Business Intelligence Environment?	379
How Do You Scale Each Tool?	381
Using Analysis Services (Multidimensional or Tabular)	381
Reporting Services	385
Using Power Pivot and Power View	387
Summary	390
Index	391

Introduction

Business intelligence, including reporting and analytics, is essential to an organization's survival and growth. Understanding your data and turning it into actionable insights is the key to sustaining and growing your business. As companies recognize this need, the employees that facilitate these insights, also known as "data scientists," are highly sought after across the board.

Microsoft's business intelligence suite contains tools that help the data scientists and developers perform their duties quickly and efficiently. Understanding which tool to use and how to use it in the best manner is required to provide the data. This book discusses each of the business intelligence tools and best practices associated with the Microsoft business intelligence stack, including reporting and analysis.

Overview of the Book and Technology

Microsoft's business intelligence landscape is changing at a faster rate than ever before. Several new methodologies have been incorporated into the stack, including self-service, big data, and the cloud. With the advent of these new methodologies affecting business intelligence, the number of tools is increasing. With all these changes, we recognized a need to have one place that describes each of these business intelligence tools and how they fit into a business intelligence solution.

The book also takes you outside the boundary of just the tools. You learn about different data and business intelligence architectures and when to use each type. You learn how to pick the tool right for your organization. You learn how to design and develop in each of the tools. You even learn what to do after you've

developed everything and need to maintain the business intelligence solution! By the time you've completed this book, you will be comfortable implementing and administering any type of Microsoft business intelligence solution.

How This Book Is Organized

This book contains four different sections. The first section provides an overview of business intelligence solutions and a discussion of some of the tools you may need. The next two sections focus on the two halves of business intelligence: reporting and analysis. And the final section takes you through the administration and maintenance of business intelligence solutions.

In Part I, Overview of the Microsoft Business Intelligence Toolset, you learn about business intelligence tools available from Microsoft and how those tools fit into an effective and useful data and business intelligence architecture. Chapter 1, "Which Analysis and Reporting Tools Do You Need?," introduces each of the business intelligence tools, some of which have been around for many years and some of which are new to the Microsoft business intelligence stack. You will also learn what development tool to use with each of the tools.

Next, Chapter 2, "Designing an Effective Business Intelligence Architecture," takes you to the architecture level of business intelligence solutions. Designing a business intelligence architecture involves knowing your audience, defining the goals for the solution, and understanding where your information resides. You then align that information with the delivery strengths and limitations of your organization to decide on the best business intelligence architecture for your needs.

The final chapter in Part I, Chapter 3, "Selecting the Data Architecture that Fits Your Organization," completes out the architecture discussion by covering data architecture. Because a huge part of business intelligence depends on the underlying data, you will learn the available structures and when to use each one.

In Part II, Business Intelligence for Analysis, you learn about the business intelligence tools meant for analyzing and gathering insights about your data. Chapter 4, "Searching and Combining Data with Power Query," introduces the self-service data integration tool, Power Query, which allows you to combine data from a variety of sources for analysis.

Chapters 5 to 8 cover the semantic modeling tools within the Microsoft stack: Power Pivot, Analysis Services multidimensional, and Analysis Services tabular. Chapter 5, "Choosing the Right Business Intelligence Semantic Model," discusses the difference between the three products and when you would use one over the others. Chapter 6, "Discovering and Analyzing Data with Power Pivot,"

explains how to design and use a Power Pivot model. Chapter 7, “Developing a Flexible and Scalable Tabular Model,” discusses the Analysis Services tabular model, and Chapter 8, “Developing a Flexible and Scalable Multidimensional Model,” discusses the Analysis Services multidimensional model.

Chapter 9, “Discovering Knowledge with Data Mining,” which is the final chapter in the analysis section, explains Analysis Services data mining. You will learn about the different mining structures and models available within the tool and the best practices for populating the datasets. Finally, you will learn how to integrate your results from the mining into the rest of the business intelligence solution.

Part III, Business Intelligence for Reporting, discusses how to use the different Microsoft business intelligence tools for reporting. Chapter 10, “Choosing the Right Business Intelligence Visualization Tool,” opens this section by discussing which reporting visualization tool you should pick. You will learn a little about each of the Microsoft reporting tools, and then learn how to complete an evaluation matrix and process to pick the right tool for your organization.

Chapters 11 to 14 conclude the business intelligence for reporting section by covering each of the reporting tools in the Microsoft business intelligence stack. Chapter 11, “Designing Operational Reports with Reporting Services,” teaches you about Reporting Services, specifically how to design operational reports and the best practices associated with using this tool. Chapter 12, “Visualizing Your Data Interactively with Power View,” moves onto this newer reporting tool. You will learn what the requirements are to create a report in Power View and then walk through the report creation. Power Map is the focus of Chapter 13, “Exploring Geographic and Temporal Data with Power Map,” which discusses how to display your geographical and temporal data. Finally, Chapter 14, “Monitoring Your Business with PerformancePoint Services,” introduces you to PerformancePoint Services within SharePoint and how you can include this tool in your business intelligence solution.

Part IV, Deploying and Managing the Business Intelligence Solution, wraps up the book by covering how to administer and maintain the business intelligence solution that you have created from earlier chapters in this book. Chapter 15, “Implementing a Self-Service Delivery Framework,” kicks off this section by describing data governance within the Microsoft business intelligence framework, and includes a discussion about the Data Quality Services and Master Data Services tools.

Chapter 16, “Designing and Implementing a Deployment Plan,” discusses deployment plans for business intelligence solutions. You learn about ways to deploy the corporate business intelligence tools and the best way to document the plan.

Next, in Chapter 17, “Managing and Maintaining the Business Intelligence Environment,” you learn how to keep your business intelligence up and running

once deployed. You will learn how to monitor your solution to ensure that it performs well and any changes you need to make to keep it running.

Finally, Chapter 18, “Scaling the Business Intelligence Environment,” covers how to handle performance issues by scaling the business intelligence tools. You learn how to scale up and scale out each of the tools.

Who Should Read This Book

This book is intended for business intelligence developers and architects, and those who are interested in learning more about the Microsoft business intelligence suite. If you need to create reports for your day-to-day operational work, design business-friendly analytics models for end users, or perform advanced analysis to make big business decisions, this book is for you.

It is assumed that you have some basic programming or SQL knowledge before picking up this book. You should understand query constructs and basic programming principles. You don’t need experience with any of the business intelligence tools discussed here, but if you do have some experience, there is still quite a bit to learn!

If you are new to Microsoft’s business intelligence tools, you would be best served by reading this book from start to finish. However, if you have some background with the business intelligence layout and need to learn about analysis versus reporting, you may want to look at just Part II or Part III, respectively. Finally, if you already have a business intelligence solution, but need to ensure that it is being managed properly, turn to the final section.

Tools You Will Need

This book is based on the SQL Server 2014 business intelligence tools, Excel 2013, and the November 2014 edition of the cloud-based software. All examples use the AdventureWorks 2012 databases and projects found on codeplex: <http://msftdbprodsamples.codeplex.com/releases/view/55330>.

What’s on the Website

Some of the chapters within this book provide sample code for you to download and use. All information is found on Wiley’s website: <http://www.wiley.com/go/appliedmicrosoftbi>.

Summary

Microsoft business intelligence tools provide a lot of power when it comes to your reporting and analysis needs. You must understand each of the tools to ensure you're harnessing that power properly. If so, you will help your organization and your own career move forward!

Overview of the Microsoft Business Intelligence Toolset

In This Part

Chapter 1: Which Analysis and Reporting Tools Do You Need?

Chapter 2: Designing an Effective Business Intelligence Architecture

Chapter 3: Selecting the Data Architecture that Fits Your Organization

Which Analysis and Reporting Tools Do You Need?

When embarking on a business intelligence (BI) project, you should consider several things. Should a centralized data warehouse be built or can the existing operational database act as the source for business intelligence? Once that hurdle has been leaped, the next question is: Should time be spent building a semantic model (cube) or again back to the original question: Can the existing operational database act as the source for business intelligence? Finally, once you've answered those questions, you need to decide how to deliver the data to end users. In other words, which reporting tool will be used? The focus throughout this book is on selecting, designing, and delivering a business intelligence solution based on the Microsoft business intelligence tools stack.

Regardless of the approach, you must make a decision concerning which tools to use to ultimately deliver the business intelligence solution. If a data warehouse is built, which Relational Database Management System (RDBMS) will store the data? Now that you have a data warehouse, is a cube or semantic model needed? If so, which type of model should you use: Power Pivot, tabular, or multidimensional? You then need to determine if the solution offers self-service reporting and/or operational reporting capabilities.

Selecting a SQL Server Database Engine

After all the politics have been hashed out, the first step in your business intelligence solution is identifying the data sources. In most scenarios, the solution will include a plethora of data sources, ranging from flat files to relational databases. After that, you must build an Extraction, Transformation, and Loading (ETL) system, which centralizes that data into a data warehouse. The data warehouse is typically housed on an RDBMS.

Building a Data Warehouse

A valid argument could be made against building a data warehouse. However, you should consider whether you prefer to report against a centralized, single-source pristine dataset or to report against multiple, disparate questionable data sources. In other words, are reports more effective leveraging data that is definitely accurate or possibly inaccurate? Another thing to consider is the responsiveness of the business intelligence solution without centralizing the data into a single repository. Often, organizations attempt to analyze data directly against source data and quickly realize that, even though simple, this approach is not efficient nor effective. Figure 1-1 shows a sample topology of this solution.

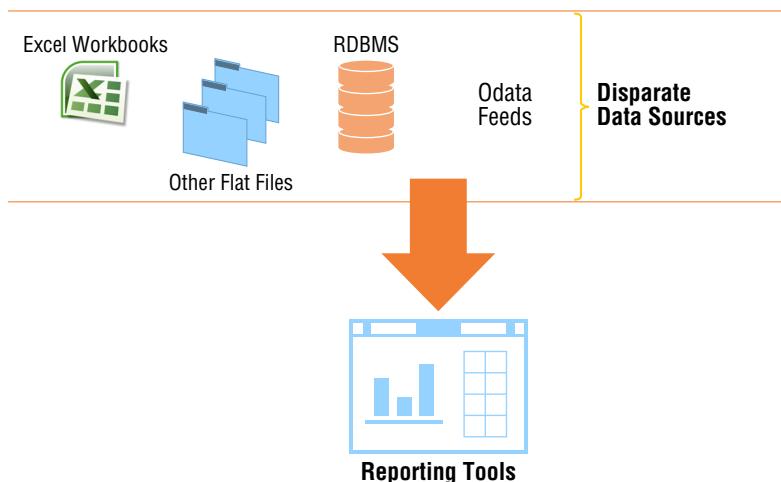


Figure 1-1: Reporting against disparate data sources

As a result, most organizations often decide to build a data warehouse. Figure 1-2 depicts a sample of a business intelligence solution that includes a data warehouse. Notice in this figure that instead of attempting to build reports against multiple data sources, a single source is used.

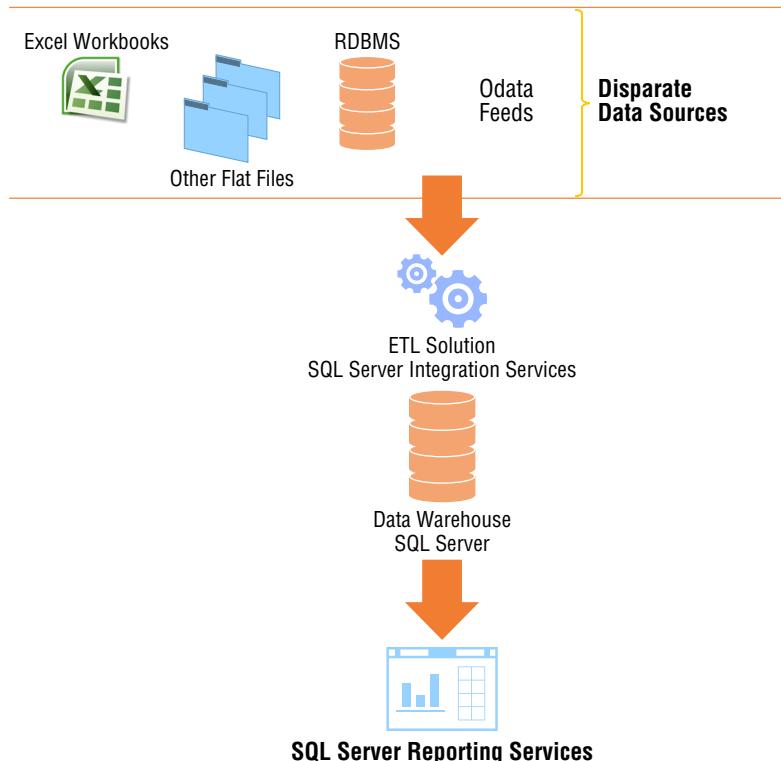


Figure 1-2: Business intelligence solution that includes ETL solution and data warehouse

Selecting an RDBMS

Once you've built a data warehouse, the next step is to select an RDBMS. The market for RDBMS systems has a wide range of choices. Selecting the correct system depends on several factors: number of users, disk space, data size, rate of growth, and frequency of data load to mention a few. Microsoft's RDBMS—SQL Server—includes several features that make it one of the more appealing systems available on the market. As of the writing of this book, SQL Server includes an in-memory Columnstore index which is designed specifically for data warehousing workloads. When included in the data warehouse design, you can achieve significant query performance and data compression. Another feature, Change Data Capture (CDC), assists in minimizing the amount of time required to load the data warehouse by providing mechanisms that detect inserts, updates, and deletes. These two features alone make SQL Server a viable Database Management System for hosting your data warehouse.

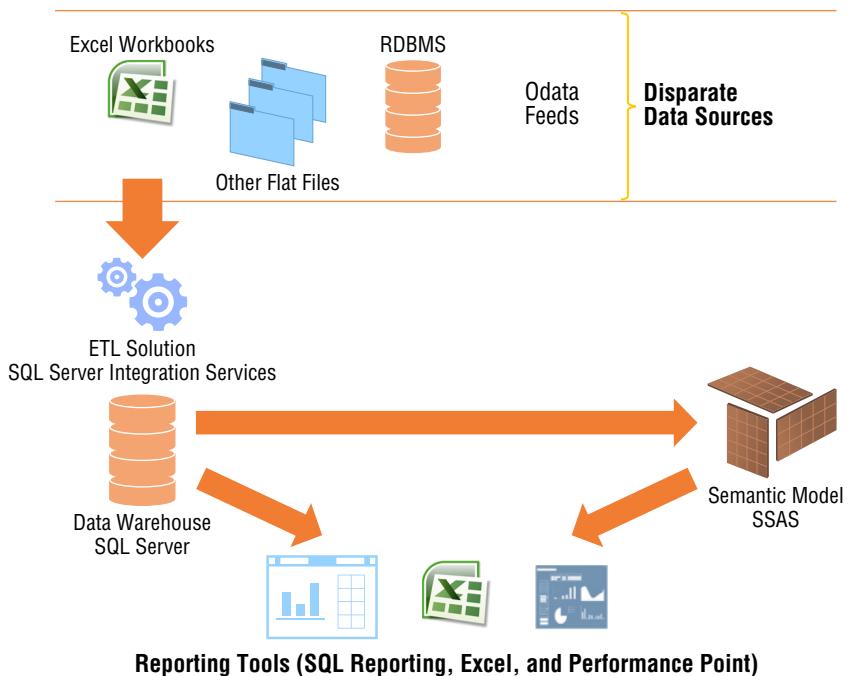
Selecting SQL Server Analysis Services

Now that a database engine is selected to host the data warehouse, the decision to build an analytical model or, in the case of a Microsoft Solution, semantic model must be made. With the latest release of SQL Server, semantic models have three choices from which you can select:

- Power Pivot
- Tabular
- Multidimensional

So not only must you decide how to build a semantic model, but also which model to use.

If the business intelligence solution requires very fast response times, ad-hoc capabilities, or predictive analytics, leveraging SQL Server Analysis Services (SSAS) is a great option. Whereas the aforementioned list is not inclusive of all factors that may drive the need for a semantic model, they definitely make a strong case in favor of it. SSAS offers a wide range of capabilities that assist in streamlining and reducing report requests, centralizing analytical formulas and key performance indicators, and—probably one of the more important robust capabilities—intuitively handling security at different levels. Figure 1-3 illustrates a business intelligence solution that includes a semantic model. Notice how the reporting tools are expanded when you compare them with Figure 1-2.



Reporting Tools (SQL Reporting, Excel, and Performance Point)

Figure 1-3: Business intelligence solution that includes SSAS semantic model

Although it is possible to report directly against a data warehouse using Excel and Performance Point (discussed later in the chapter), SSAS provides a more innate design experience with these tools. In addition, using SSAS provides end users with a larger surface of self-service capabilities that are unavailable when only a data warehouse is available. Therefore, they are excluded from Figure 1-2, but included in Figure 1-3.

For example, if you are the CEO of a company, you may require access to every aspect of data in the model. However, if you are a regional or departmental manager, you may only require access to data that is pertinent to your region or department. SSAS includes built-in capabilities that let you control access to data at the row level. In many cases, this is one of the most important and often overlooked requirements of a business intelligence solution. During most projects, you don't realize this until very late in the development process. However, when using SSAS, the implementation process is neither very difficult nor disruptive.

Working with SQL Server Reporting Services

Up to this point, all the data discussions have involved movement, transformation, and management of data. This section shifts to more data visualization and interactivity. Once the processes to implement the data warehouse and/or the semantic model are in place, your next decision is how the end users will access the data. When leveraging the Microsoft business intelligence stack, organizations have several reporting options. From an operational perspective, probably the most utilized is SQL Server Reporting Services (SSRS).

SSRS operates in two modes, which have a few slight differences, but are mostly similar in regard to features and tasks:

- **Native mode:** Access and management of reports are available via a web-based platform, also known as Report Manager.
- **SharePoint Integrated mode:** This is a site collection within SharePoint that has the same purpose as the Report Manager.

SSRS also provides two very different types of reporting experiences. Deciding which to use often poses the biggest challenge for most projects:

- **The first, Operational Reports, are typically used when delivering highly-formatted, table-based and pixel perfect reports.** They are designed to answer a specific question and are usually static in nature. In this case, you would use SSRS.

- **The second type is of a more ad-hoc nature.** End users typically access the underlying source directly, which would be a semantic model in this case, and build reports as needed. The reports are more visual containing charts, maps, gauges and scorecards. For these types of reports end-users would leverage Power View. Each one is discussed in the following sections.

In addition to developing these types of reports, SSRS provides additional capabilities that make it a complete solution. Features include:

- Report export
- Subscription report delivery
- Data alerts (SharePoint Integrated mode only)
- Data caching
- Report printing
- Report snapshots
- Shared datasets
- Report parts
- Geospatial mapping

While this is not an exhaustive list of all the features, it should provide an overview of what is possible when developing and managing reports using SSRS.

Understanding Operational Reports

Operational reports, available since the inception of SSRS, can help you develop, deploy, and manage standard operational reports. What are operational reports? These are typically row- and column-based reports containing data that answers or meets a specific need. For example, the report shown in Figure 1-4 shows a sample Operating Summary developed using SSRS.

Contoso Education System > SQL Reports Demo > Colleges and Universities > Reports	
Actions	
1	of 1
<	>
Find Next	100%
EDUBOS\PLeBlanc	
JUL-13 Operating Statement Summary	
*blue text denotes drill down	Actual
Beginning Equity	43,015,495,386.66
Less prior years capital	<u>(890,495,450.61)</u>
Subtotal Beginning Equity	42,124,999,936.05
Revenue	
Student tuition and fees	179,631,755.39
Tuition Discounts	(870,088.32)
Net tuition and fee revenue	178,961,669.07
Federal, state, and private grants	262,992.38
Contributions	3,242,204.80
Sales of Educational Departments	2,113,184.45
Investment Income	16,288.80
Endowment income used in operations	0.00
Other income	24,874,519.83
Total revenue	209,470,947.13
Expenditures	
Compensation:	
Salaries	11,279,098.47
Fringe benefits	4,081,301.53
Total compensation	15,360,400.00

Figure 1-4: Operating Statement Summary using SSRS

This particular report was designed for a specific audience to solve a specific problem, which in this case was a need to dynamically view operating summaries by month for those individuals in the Accounting department.

Using SSRS, developers can also build very visual reports that resemble high-level dashboards often used by executives; Figure 1-5 illustrates this.

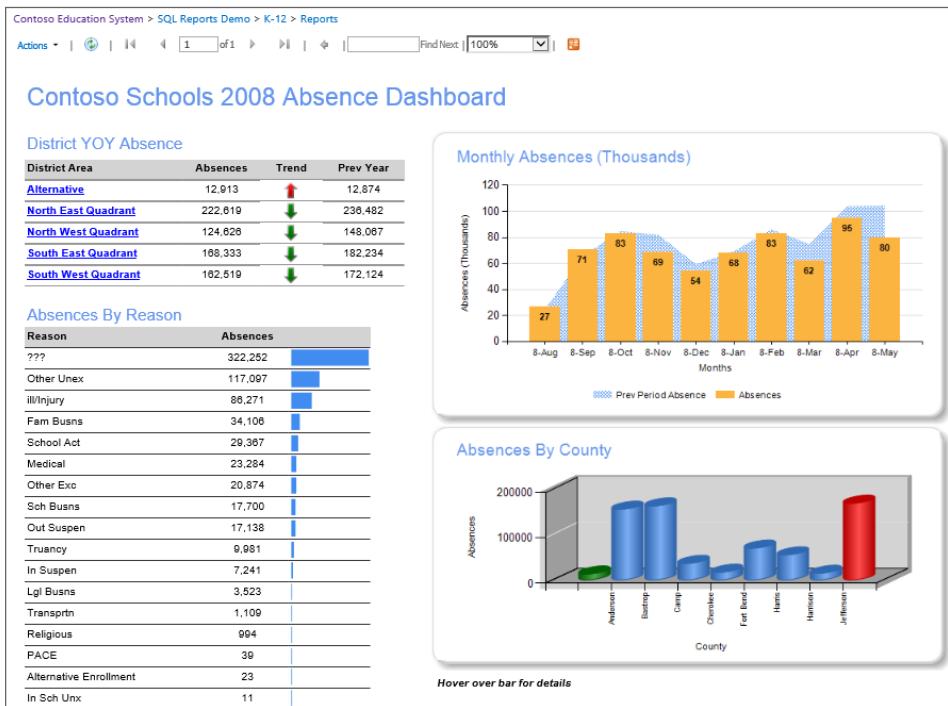


Figure 1-5: High-level dashboard created using SSRS

SSRS includes a complete toolbox of items that allow report developers to build complete reporting solutions including high-level dashboards that provide end users with drill-through ability to more detailed data.

Understanding Ad Hoc Reporting

Suppose end users want some control over the look and feel of reports. More specifically, what if they want ad-hoc access to data, which allows them to create and deploy reports as needed, instead of relying on a group of report developers creating canned reports. The latest release of SSRS integrated into SharePoint, discussed later in the book, exposes a new feature named Power View.

Power View is an ad-hoc, interactive, and presentation-ready self-service reporting tool designed specifically for end users. Instead of waiting on reports from the report development team, end users can quickly access data that is stored in either type of semantic mode and build highly visual and interactive reports. Figure 1-6 displays a sample of a Power View report.

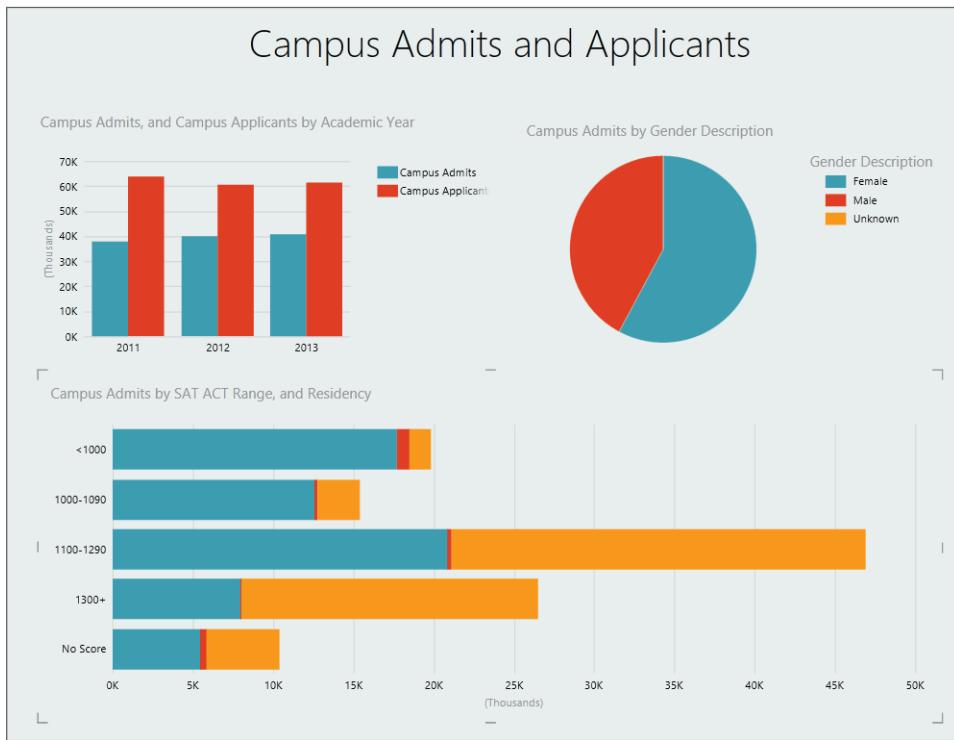


Figure 1-6: Sample Power View report

With Power View, end users become the report authors. A view of the data is made available in a field list, which provides easy access to each data element exposed via a single semantic model.

Working with SharePoint

Traditionally, when Microsoft business intelligence is included in a business intelligence project, most people automatically think of SharePoint. That is because SharePoint acts as a central portal or repository that you can easily access using any modern web browser. Instead of using multiple disparate technologies for the business intelligence solution, SharePoint can render both Microsoft and non-Microsoft reporting visualizations.

It is a common practice that individuals, teams, departments, or organizations will evaluate and select a reporting tool based on a variety of factors. These tools then tend to proliferate themselves within the department and become part of daily operations. Once the discussion of a business intelligence solution begins, it is often difficult to persuade the groups away from their established tool of choice. However, using SharePoint means the IT group responsible for the business intelligence project can couple existing report artifacts with new

technologies to produce a fully functional and comprehensive solution via a single interface. Figure 1-7 provides a pseudo view of what a SharePoint-developed solution may look like.



Figure 1-7: Pseudo SharePoint page displaying multiple technologies

In SharePoint, you can develop custom pages that consume other technologies. Figure 1-7 integrates four different technologies into a single view. The resulting web page centralizes four distinct views of data spanning features that are made available by different software vendors. In addition to customized pages, business intelligence developers can leverage Performance Point, a dashboarding feature of SharePoint. The details of Performance Point are discussed in the next section.

While SharePoint does offer this extended capability of integrating disparate technologies, the Microsoft business intelligence stack does provide a sufficient number of tools and features for deploying a holistic business intelligence solution. Therefore, if an organization is evaluating vendor solutions, leveraging SharePoint typically addresses all the business intelligence needs for a given project.

Working with Performance Point

The previous section focused on SharePoint as a whole. However, when SharePoint is deployed, you have an option to configure Performance Point services. Using Performance Point, developers can create dashboards that aggregate data from a

collection of sources such as Analysis Services, Excel Workbooks, and SharePoint lists. Similar to Power View, Performance Point gives your users a very interactive interface for analyzing data. Where Performance Point really shines over Power View is that it automatically exposes the metadata from the underlying data model as part of the end-user experience. In other words, once a dashboard deploys, end users can simply right-click a given visualization and change the look by drilling down to a different level of the data. Figure 1-8 shows a sample Performance Point dashboard.



Figure 1-8: Performance Point dashboard

This particular dashboard is a high-level view of medical discharges for a given year and service area. By right-clicking a bar in the bar graph (shown in Figure 1-9) or changing a filter on the dashboard, end users can dynamically analyze the data based on the underlying data model.

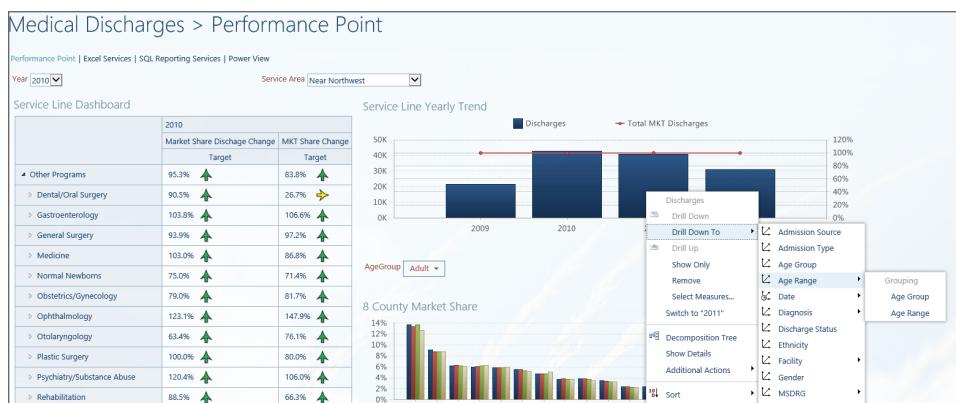


Figure 1-9: Performance Point dashboard with drill-down menu displayed

Therefore, instead of IT developing several reports of varying levels of the same data, you can create a single dashboard that provides end users with different views from one entry point.

Using Excel for Business Intelligence

When most people think of Excel, they think of spreadsheets and pivot tables. However, the latest release of Excel has definitely matured into a full-grown business intelligence authoring tool. Excel 2013 now comes with two new plug-ins in the product (Power Pivot and Power View) and has two additional plug-ins available for download (Power Query and Power Map), thus transforming Excel into a full-fledged business intelligence solution. By leveraging all four plug-ins, users of Excel can discover, model, and visualize data from a single tool. By including Excel as part or all of the business intelligence solution, you gain the primary advantage of providing a familiar tool to the entire population of end users. This results in a lower likelihood of resistance to adoption. The following sections provide a brief overview of each add-in.

What Is Power Query?

Microsoft Power Query is a self-service data discovery and data access tool. It enables end users to easily combine, transform, and share data. When Power Query is installed, not only can end users access structured data from within Excel, but they can also perform public searches. This search is similar to a Bing search; however; instead of returning a list of web page results, Power Query returns a list of datasets that match the entered query.

What Is Power Pivot?

Once all the data has been identified, end users can use Power Pivot to build in-memory data models—meaning they can perform data analysis directly inside Excel. Power Pivot has the ability to consume and process large amounts of data, beyond the normal Excel limits, while including those features of Excel familiar to most Excel users. In addition, Power Pivot introduces a new expression language, Data Analytic Expression (DAX), which provides new data analytic capabilities.

What Is Power View?

Now that you've transformed and modeled all the data, the next step is visualization. To accomplish this task, two new add-ins are now available: Power View

and Power Map. Power View, previously available only via SharePoint, has been added to Excel. The Power View experience is very similar in Excel when you compare it to the discussion in the section “Working with SQL Server Reporting Services.” The primary difference is that authoring is done directly inside of Excel rather than SharePoint. Although this may not seem like a huge difference, if you are an Excel user, it may determine whether you’ll use Power View or not. Besides that, there are a few slight variations, but nothing too significant.

Power Map

Power Map is a 3-D visualization add-in for Excel. This add-in consumes geographical and temporal data and maps it on a 3-D representation of the earth, shown in Figure 1-10.

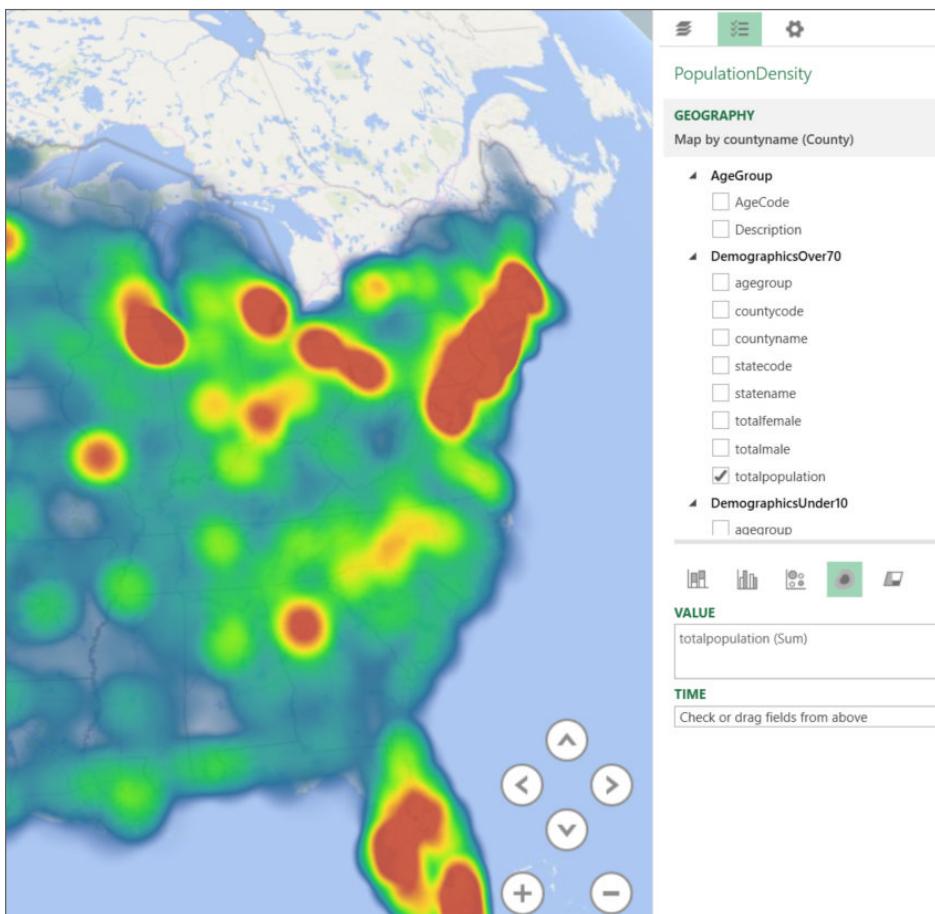


Figure 1-10: Sample Power View map with field list

This means that end users can interact with and derive new insights from their data. The end-user result can be a video that tells a complete story of the data in different scenes that represent various views.

Which Development Tools Do You Need?

So far the discussion has focused on those Microsoft tools that host data and provide end-user consumption. The focus now shifts to the tools you actually need to develop the solution. Primarily four tools are used in the development process:

- SQL Server Data Tools (SSDT)
- SQL Server Management Studio (SSMS)
- Performance Point Dashboard Designer
- Report Builder

What tools end users utilize is determined by what they're developing and who does the development. In some cases, all tools are used, and in others cases, only an abbreviated set. Some instances may require the use of tools beyond the Microsoft stack. However, for the sake of brevity and because this book is focused on the Microsoft business intelligence, we'll discuss only tools specific to Microsoft.

Using SQL Server Data Tools

SQL Server Data Tools (SSDT) is the most comprehensive set of tools in the list. SSDT offers a full-range experience from which developers can address almost every facet of a business intelligence solution from within a single environment. Figure 1-11 displays a list of the templates available to developers from within SSDT.

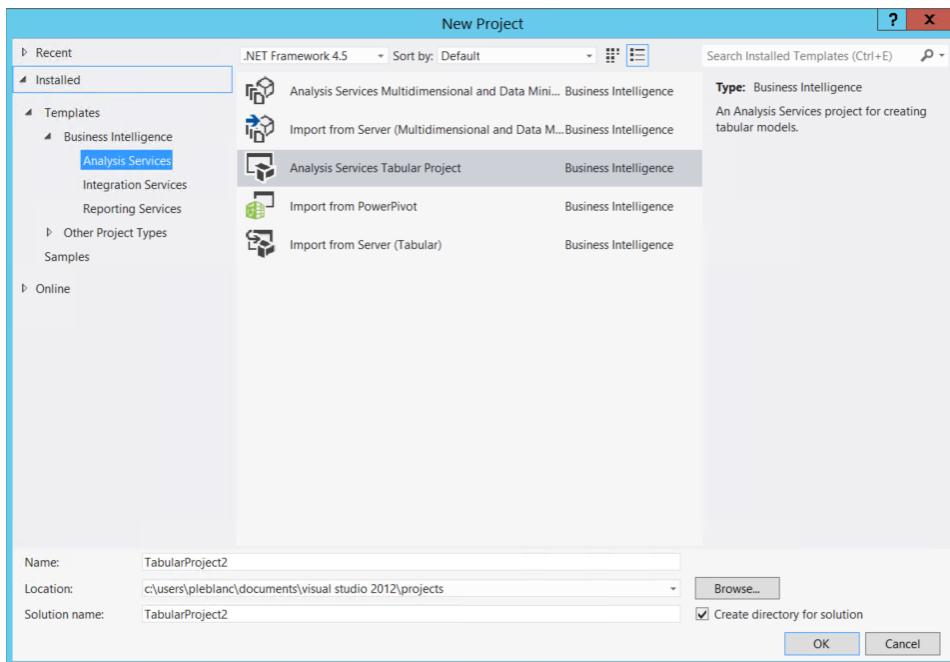


Figure 1-11: SQL Server Data Tools template list

In this tool, you can develop SQL Server Integration Services, SQL Server Reporting Services, and semantic model solutions individually or as a team. You can also develop database solutions using SSDT, which is a perfect environment for developing the data warehouse schema because SSDT provides capabilities such as refactoring code and schema compare. In addition, by leveraging SSDT as the development tool, you can version-control the entire solution using Microsoft Team Foundation Server or other third-party version-control tools like Subversion.

Using SQL Management Studio

SQL Server Management Studio (SSMS) is often considered a Database Administrators (DBA) – centric tool. However, you can also use it to develop a data warehouse schema. You could argue that it does not provide a complete development environment because it lacks certain features like version control and refactoring. Although this is true, it does provide an interface for testing and debugging Transact-SQL (TSQL) code and a diagramming feature that allows for creating and managing tables and relationships. Figure 1-12 provides a view of a typical star-schema data warehouse from the SSMS perspective.

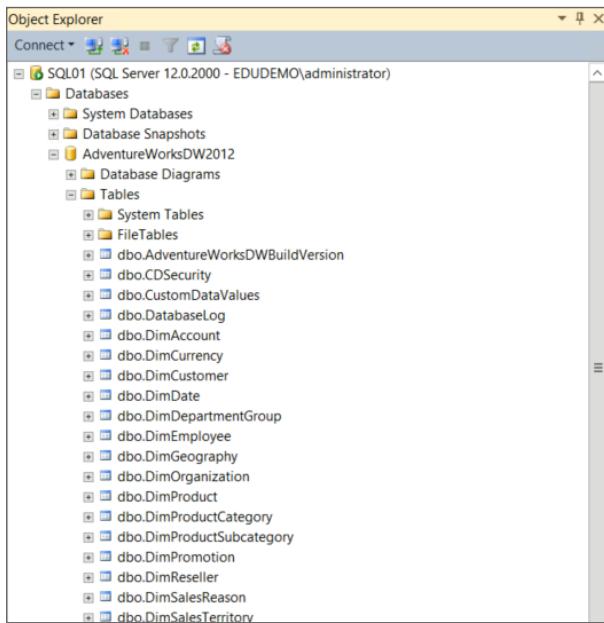


Figure 1-12: SQL Server Management Studio

Using Dashboard Designer

The starting point for authoring Performance Point dashboards, discussed earlier in the section “Working with Performance Point,” is Dashboard Designer. Dashboard Designer is a click-once application available in SharePoint that is installed on each individual machine that will author Performance Point content. It is not available by default. Once installed, you can launch Dashboard Designer to build and deploy dashboards to a SharePoint site, which is shown in Figure 1-13.

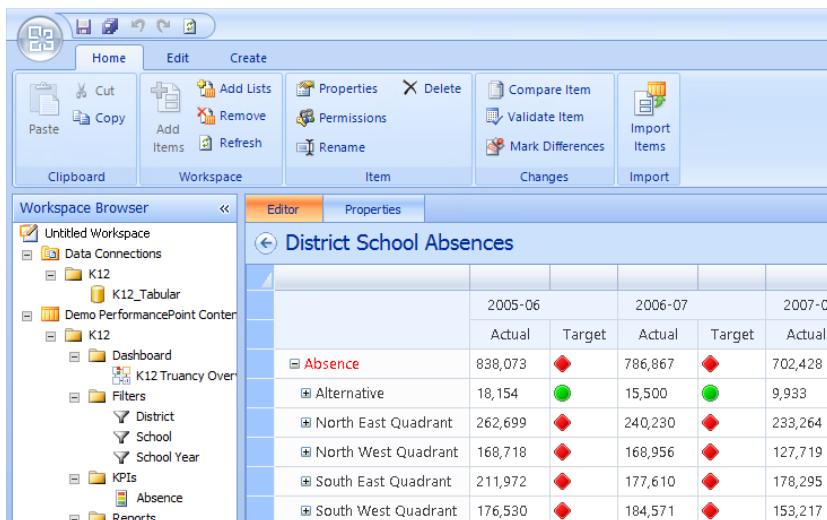


Figure 1-13: Dashboard Designer displaying scorecard

Using Dashboard Designer, you can develop each individual item that appears on the dashboard and then create a dashboard that displays the finished product. Once complete, you can then deploy the dashboard to view it.

Using Report Builder

Similar to Dashboard Designer, Report Builder is also a client tool that you must install on each individual developer's machine. You use Report Builder to develop SQL Server Reporting Services reports. When you compare it to designing reports using SSDT, the features and functionality are almost exactly the same. The difference is mainly in regard to team development and source control. A report designed in SSDT is included within a solution and a project. When designing reports with Report Builder, you can develop only a single report at a time. The concept of team collaboration is not introduced. Report Builder is a light-weight tool for people that need to quickly create and/or modify reports as needed. The application itself is installed on demand, similar to PerformancePoint.

Figure 1-14 displays a sample report being developed with Report Builder.



Figure 1-14: Sample Report Builder report

Summary

This chapter discussed the different Microsoft business intelligence tools that you can use to author, deploy, and maintain a business intelligence solution. In addition, overviews were provided for each tool, and, in some cases, comparisons among the different features and capabilities were provided.

The next chapter provides an overview and some discussions around designing an effective and efficient business intelligence architecture.

Designing an Effective Business Intelligence Architecture

Whereas the tools that you use to develop the business intelligence solution are one of the top priorities in the project, designing an effective and performant solution may be almost as important, if not more. Once the solution is developed and placed in the hands of the end users, accessibility, availability, and responsiveness now become the top priorities of the solution and of those who manage and maintain it. If either fails, then the repercussions could be detrimental or catastrophic to the entire project. As a result, prior to development and deployment, you should carefully consider identifying who the audience is and what the goals of the project are.

This chapter helps you define your audience and goals, explains the reasoning behind data sources, and gives you the information you need to determine if you need a data warehouse. You also find discussions on data governance, analytical models, and delivery solutions. Happy planning!

Identifying the Audience and Goal of the Business Intelligence Solution

Careful consideration must be taken when identifying the audience and goal of the business intelligence solution. These are key factors in most development projects, not just business intelligence projects. Although the audience often

dictates the goals, it is important to realize that in comparison they are both equally important. More specifically you consider:

- What are the data requirements?
- Is there a need for a data warehouse or a semantic model?
- What are the hardware needs?

Without adequate knowledge, these questions could result in the development of a solution that does not meet the needs or goals of an organization. Even worse, they may create a solution that meets the requirements, but cannot physically support an organization because of poorly sized hardware.

Who's the Audience?

Identifying the audience should be the starting point, because if you do not have an intended group of end-users, what is the purpose of the project in the first place? Most successful projects succeed because they have some type of buy-in or sponsorship from a larger group that is not part of the development team. For example, a common business intelligence project involves determining past, current, and future sales for a given company. The request for this may come from the CEO, Finance or Marketing department, or from a branch office. Either way, now that the project is aligned with a specific group or groups, obtaining resources to produce the solution becomes much easier, which is an advantage that any development project can benefit from. These resources, include, but are not limited to software, hardware, and people.

The newfound partnership comes with a list of items that may or may not positively assist the project. One item is goals, discussed in the next section. Another, and probably more important, is deadlines. End users often measure the success of a project based on hands-on interactivity. If end users have nothing to see, touch, and use on a given date, the project can easily lose the trust and support that existed at the inception of the project. You could categorize this as a positive aspect of the partnership in regards to success. Deadlines should ensure the on-time delivery of the solution. However, the developers working on the project may see this as negative because it could inhibit or minimize what can and will be delivered due to time constraints.

Another item that is often a result of this partnership is the amount of partner involvement (or the lack thereof). Because a large part of a business intelligence project is discovery, partners must spend time discovering data sources, data needs, goals, how to visualize, which tools to use to visualize, and so on. This requires the involvement of certain people, who are often already inundated with their existing jobs. Because their time is already at a premium, finding additional time to devote to the new project is difficult. Although developers

are often well versed in and have intimate understanding of the data to use, they often lack the knowledge of how to massage the data to meet the project's requirements. Without that knowledge, the business intelligence project is destined to fail before it begins.

What Is the Goal(s)?

Now that you've identified your project's end users, it's time to take their knowledge and convert it into goals and requirements. These goals and requirements typically equate to the scope of the project, which further assists in defining timelines, selecting tools, identifying data needs, and selecting hardware. Within the scope, you outline certain goals, such as what should be developed, how it should look, who or what should have access to it, and what to use as the delivery mechanism. Although not an exhaustive list of outlined goal types, they should assist you in quantifying and identifying a project's goals.

What Are the Data Sources?

With the users and goals identified, the time has now come to perform one of the most difficult steps in the project—identifying the data sources. Believe it or not, a person or group of people from the expected end users are the best source for this task. IT data sources often reference data that resides only in systems that IT manages with no regard to data hosted by a department, branch office, or third party. For a business intelligence project, this is not typical. You must perform an exhaustive search-and-discovery process with as much involvement from any stakeholders that are willing to assist. You may host these sources within or outside an organization.

Regardless of whether the data is internal or external, you should carefully perform discovery to ensure that you have included all relevant data in the project. This process often requires several iterations. During report development, someone may recognize that data is missing due to an oversight. As a result, you'll need to modify the ETL process to include the new source.

Using Internal Data Sources

While you source the majority of the data from traditional IT-managed relational databases, you'll always have some data managed and maintained outside the IT department. This data is likely stored in spreadsheets, Access databases, comma-delimited files, text files, or other file types, and they may reside on someone's desktop or laptop. Often vital, these datasets contain small nuggets of information that can cripple the project if they are not included.

End users may also manage other internal data sources, such as SharePoint lists or third-party applications that came with database backed during installation. In the case of the latter, the hosting department does not even realize what they have installed. In some cases these are common back ends, and others may require custom drivers to access the data.

Using External Data Sources

The data may also come from an external source via a web service, an OData feed, or even a hosted RDBMS (SQL Server or Oracle). If the source is a web service or OData feed, the data is typically accessed via a web URL. The consumption, on the other hand, may require some custom interface that parses and displays the data in a fashion meaningful to end users. Developers may overlook this data because no one on the team knows it exists; the same may apply to the hosted databases. As a result, this further heightens the need to involve end users because they may be the only people who know it's there and needed.

Using a Data Warehouse (or Not)

For the experienced business intelligence developer, developing a business intelligence solution without a data warehouse may seem absurd. However, with today's savvy end users, readily accessible data is no longer an option; it is a requirement. Therefore, nightly refreshes of data is becoming a thing of the past. And as a result, including a data warehouse as part of a business intelligence project is now optional.

Traditionally, a data warehouse is loaded at some time interval—daily, weekly, and some even monthly. Depending on the organization the time period may be longer. For example, some colleges or universities load data into a data warehouse only at the end of the semester. As data needs become more stringent, the periods of latency between live data and analytical data have become smaller and smaller, presenting challenges that are often difficult to overcome. The primary challenge is moving data from the source systems to the data warehouse, which leads to the question: Is a data warehouse required?

Think back to Chapter 1, specifically to Figure 1-1, which we're showing again in this chapter (see Figure 2-1). This figure depicts an illustration of reporting from multiple data sources. How can you create a single report to reference multiple desperate sources? Which tool would you use? A few may accomplish this task, and because this is a Microsoft-focused book, the tool that comes to mind is Power Pivot. Power Pivot is an Excel add-in that creates an in-memory

semantic model based on a plethora of data sources. Figure 2-2 displays an abbreviated list of data sources that possibly source the data warehouse.

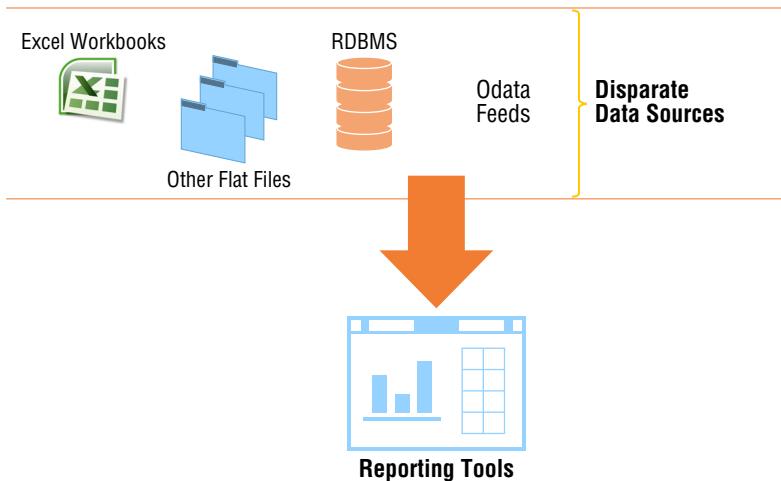


Figure 2-1: Reporting against disparate data sources

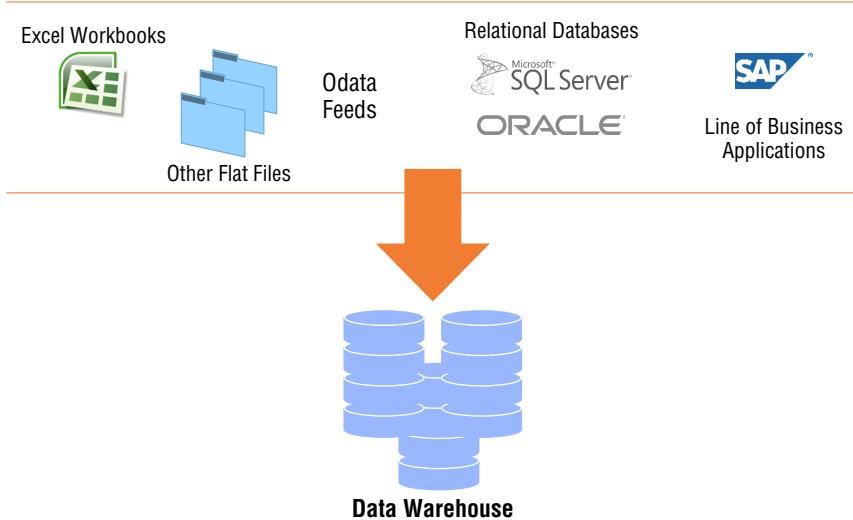


Figure 2-2: Data sources that source a data warehouse

With that in mind, and because this feature is available, you might ask yourself: Why should I develop a data warehouse? Couldn't data be accessed directly from the source, modeled, and then reported against? The answer is: absolutely. But in most cases the data has not been validated so that it could meet the needs

of every aspect of the business, or possibly two systems exist that store similar data. Which set should you use for reporting? Another possibility is that a value is calculated one way by one set of users and another way by another set. Which means of calculation is the correct one? This is where the ETL process becomes a significant part of your business intelligence solution.

NOTE Although this book does not focus on loading the data warehouse using SQL Server Integration Services (SSIS), please do not discount the importance of an ETL solution. The Microsoft Business Intelligence 24-Hour Trainer provides an SSIS section that provides a good starting point and overview.

During the ETL process, the data is extracted from the original data sources, transformed into a format or formats that meet the business requirements, and finally loaded into a central repository (data warehouse). You can use the repository as a direct source for reporting or as a semantic model. Regardless of the approach, leveraging a single source for either makes the process of obtaining and visualizing data simpler for any individual or group that needs to access the data.

Implementing and Enforcing Data Governance

The section “What Are the Data Sources,” reiterated the importance of a group of end users over and over again. You have a similar case when deciding how to implement and enforce data governance. From a technical perspective, this step is typically performed prior to loading the data in the data warehouse. So, why is it located after the data warehouse section? When planning a business intelligence solution, time is valuable. Why invest time and resources on something that you may not need? If data is accessed directly from the source, the want or need for data governance is reduced. However, because you may have multiple sources of data for a single data warehouse, you may require a data governance strategy to ensure that you deliver accurate, consistent, and trustworthy data to end users. Figure 2-3 illustrates where data governance would fit in a Microsoft business intelligence solution.

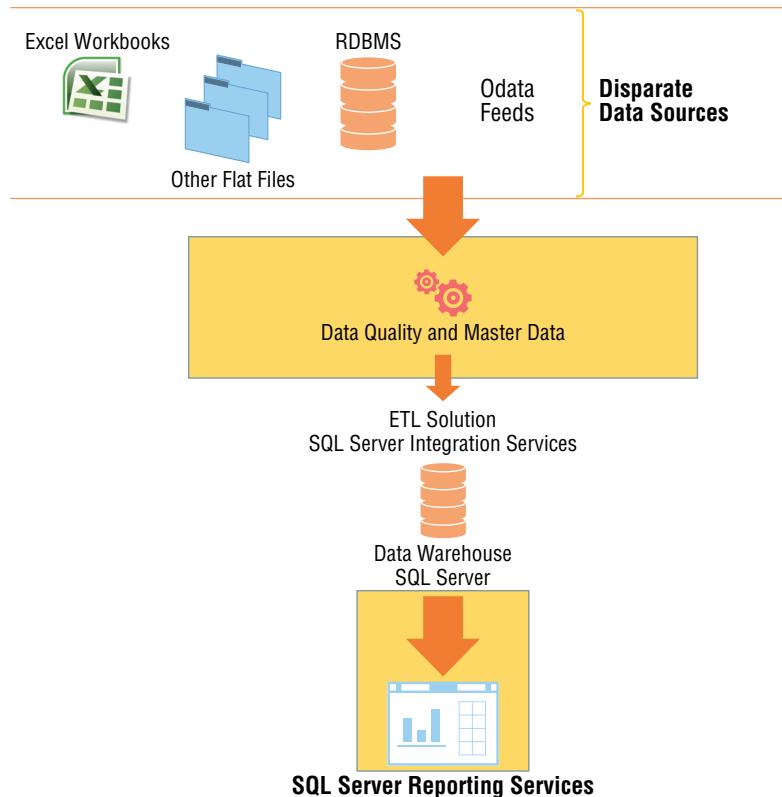


Figure 2-3: Data governance in a business intelligence project

DATA GOVERNANCE

So, what is data governance? In short, *data governance* is a process or set of processes that include:

- Data quality
- Data management
- Data monitoring
- Data maintenance
- Data security

These tasks are accomplished by putting people familiar with the data together with a technology solution or set of solutions. In the case of Microsoft, it would be a set of technologies: Data Quality Services (DQS) and Master Data Services (MDS).

NOTE A discussion about these technologies is well beyond the scope of this book; however, the exclusion of a data quality process could be detrimental to the entire business intelligence solution. The Tutorial: Enterprise Information Management using SSIS, MDS and DQS Together, teaches developers how to implement a sample Enterprise Information Management (EIM) solution. It can be downloaded from this link: <http://www.microsoft.com/en-us/download/details.aspx?id=35462>.

For example, assume that during the data discovery process you realize that two very similar datasets are obtained from two completely different sources; for the sake of brevity and simplicity, assume that both datasets reference gender data. The catch is that in one system gender is stored as Male and Female, and in the other the data is stored as M and F. Prior to importing the data into the data warehouse, a decision must be made regarding how to store gender; this is part of the data governance process. Using the knowledge of those individual stakeholders and the technology of choice (MDS and DQS in this case) you can implement a process that ensures that the data is accurate and consistent.

Planning an Analytical Model

The next logical step in architecting a business intelligence solution is deciding whether an analytical model is necessary. Prior to the release of SQL Server 2012, making that decision was a much simpler process. However, considering the added and enhanced features in that release as well as the latest release, as of this writing, making that determination is now a little more complicated. Why, and what has complicated this process?

In most cases, the driving factors behind developing analytical models is improving query performance time. Developers have come up with some work-arounds that circumvent the need for an analytical model, such as building aggregate tables, using indexed views, or adding more hardware. Although these solutions work, they usually only temporarily fix the problem. As data needs grow, the return on either of the aforementioned solutions becomes too expensive or simply just does not work. Recognizing this, Microsoft included a new column-based index within the database engine. By implementing these index types, you can increase query performance several magnitudes over, possibly eliminating the need to develop an analytical model entirely.

Although the new column-based index may improve query performance, when it comes to addressing analytical needs such as complex time-based analysis, key performance indicators, hierarchies, unary operators, and other capabilities that are beyond the scope of a relational model, the only choice is to develop an analytical model. Prior to SQL Server 2008R2, you only had a single type to choose from when developing a business intelligence solution based on the Microsoft stack. However, with later releases of SQL Server, you have three

choices: Power Pivot, tabular, and multidimensional. Chapter 5 provides a detailed explanation that will assist you in choosing the right one. However, regardless of which model you select, the planning process is very similar across all three.

Prior to building the model, you should consider the following:

- **Make a concerted effort to ensure that the data warehouse schema is almost 100 percent complete, including the data governance and load process.** The model development could actually begin without the data; however, little can be done in regards to validating the code without data.
- **Ensure that the model requirements have been scoped.** This may seem like an obvious step, but often developers start building the model without consulting any stakeholders or end users. Typically, the result is something irrelevant or wrong.
- **Decide on data latency.** In other words, can the data in the model be 15 minutes behind, 1 hour behind, 1 day behind, or 1 week behind. This decision is ultimately based on end-user needs.

If a decision was made not to develop a data warehouse and instead obtain the data directly from the source, it is possible to have almost real-time access to data by implementing either a Direct Query tabular model or Real-time Online Analytical Process (ROLAP) multidimensional model. Note that this feature is not available when developing a Power Pivot model. These methods can also be implemented if a data warehouse is implemented, but the data will be as fresh as the data in the warehouse. For example, if the data is loaded into the warehouse nightly, then you'll have a 24-hour difference between the data in the source and what you'll see in the model.

On the other hand, a period of latency may be acceptable. If that is the case, you should add a step after the data warehouse is loaded and prior to anyone accessing the data. This step will load or process the data into the model. For tabular data it would be an In-Memory model, and for multidimensional data it would be a Multidimensional Online Analytical Processing (MOLAP) model. In addition, adding this step also opens up the possibility of developing a Power Pivot model, which was not available before.

One more thing to consider is a hybrid scenario where some data can afford latency and some may not. In that case, one solution would be to use a Hybrid Online Analytical Processing (HOLAP) model, in which some objects access data real-time and some require the processing of data.

Planning the Business Intelligence Delivery Solution

With everything in progress or close to completion, the time has come to decide how to deliver everything to end users. More importantly, you must ensure that the selected topology or solution is performant, secure, and available.

Implementing a solution that encompasses all three of these characteristics can be time-consuming and costly. Which leads to the question: Are they all required? As a best practice yes, but typically no. The goal of most deployments is to improve performance and to present data in ways that resonate with consumers. Security and availability are usually an afterthought, often not occurring until it's too late. Fortunately, when building a Microsoft business intelligence delivery solution, you have several options available to economically implement a solution that addresses all three.

When implementing Microsoft business intelligence in an enterprise-wide manner, you should consider using SharePoint as the preferred method for delivery due to its collaborative nature and centralized delivery methodology. Its ability to consume data from an array of heterogeneous sources, coupled with the plethora of visualization capabilities, makes it a complete solution that addresses a variety of needs. On the other hand, architecting a solution that performs in a manner that exceeds expectations often requires a subject matter expert. Moreover, ensuring that the deliverables (dashboards, reports, spreadsheets, and so on) are always available and secure can potentially introduce other challenges and requirements.

Considering Performance

As mentioned earlier, performance is often a driving factor behind a large percentage of business intelligence projects. Most of the time this performance is related to accessing data and producing reports. While some may argue, including myself, that a business intelligence project should solve a problem, ultimately the end goal must be granting end users access to data in an effective but also efficient way. Satisfying this goal is accomplished by architecting a server topology that can scale and grow. This is where the Microsoft business intelligence stack lends itself perfectly. Figure 2-4 illustrates a sample SharePoint deployment.

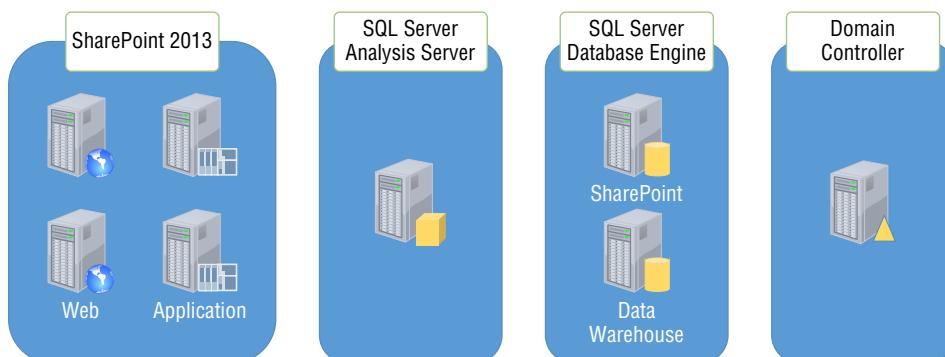


Figure 2-4: Sample Microsoft business intelligence server topology

This configuration is scaled out to support a high number of end users and a growing number of requests. You could reduce the number of servers in this configuration by consolidating the web server and application server, but this could adversely affect performance depending on the number of requests sent to the web server. Therefore, as a best practice, always separate the two in anticipation of some type of growth. In addition, as the end-user base grows, SharePoint provides the flexibility of adding more web and application servers to accommodate the increasing population. With multiple servers in place, not only has scale been introduced, but now SharePoint can use its internal load-balancing features to provide optimal performance for end users.

Considering Availability

The topology presented in Figure 2-3 does offer options to improve scale and support a high level of performance. However, it lacks a key factor that should not go overlooked when implementing the solution: availability. In most cases no one considers availability until a disaster occurs. For example, what happens if the SQL Server that hosts the SharePoint databases shuts down? How much downtime will accumulate prior to the system coming back online? As a result, in addition to installing and configuring more web and application servers for scale and high availability, additional servers should be added to ensure that a high level of availability is maintained for all parts of the configuration.

In Figure 2-4, the Domain Controller, SQL Server and Analysis Servers are single machine. This typically introduces a problem of availability. To ensure that the servers are protected in the event of a failure or disaster means implementing some type of High Availability (HA) or Disaster Recovery (DR) solution. You could use clustering for the domain controller and a mix of clustering an AlwaysOn for the SQL Server Analysis Server and the SQL Server Database Engine. Both clustering and AlwaysOn are technologies that provide HA and DR at server and database levels. Implementing this topology almost doubles the number of servers. Figure 2-5 represents a sample of an HA solution.

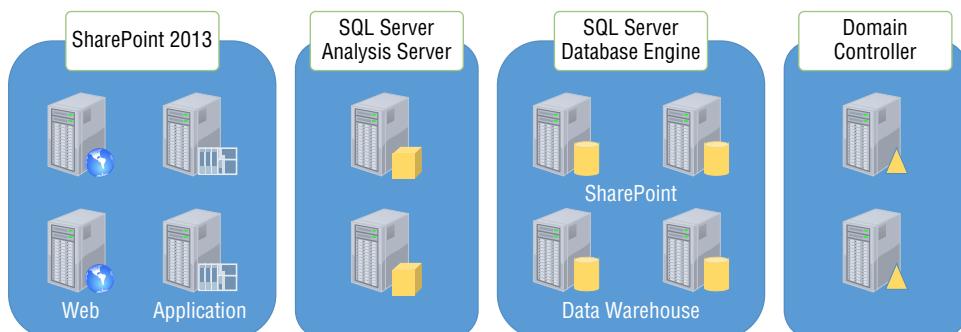


Figure 2-5: Sample Microsoft business intelligence HA server topology

The SQL Servers and domain controllers are doubled to ensure that the environment remains online in the event of a server failure. What is not represented in the figure is an additional environment that protects against the loss of the primary data center. This addition would even further extend server requirements and increase capital investments. Although costly, this added environment could provide more protection and availability in the event of a major disaster.

Summary

The chapter discussed various techniques and methods that you can use when architecting a BI environment. It also provided various approaches that can assist with identifying users, goals, and requirements. Finally, this chapter identified plans to help you define how to implement certain aspects.

Chapter 3 discusses various architecture implementation strategies. It then moves onto topics that help you identify your organizational needs and how to select the correct architecture.

Selecting the Data Architecture that Fits Your Organization

As Tim Berners-Lee famously quoted, “Data is a precious thing and will last longer than the systems themselves.” The task of choosing the correct data architecture to safeguard that data falls on data architects, developers, and database administrators. But how do you determine that ideal data architecture? You must take many factors into account to reach the final decision. Be sure to pick the right data architecture from the beginning; otherwise, you will have to reengineer your solution, wasting time and, often, your business users’ confidence in the solution.

This chapter helps you select a data architecture that fits your organization. First, you will learn the challenges of not having a data architecture, the importance of having a data architecture, and common data architectures found in organizations just like yours. Then, you will follow a process to determine the best data architecture for your organization. The process includes interviewing key stakeholders, documenting the key factors associated with your organization, and using a flow chart to find out what data architecture works best for you. Finally, you will work with your stakeholders to ensure you have picked the ideal data architecture that will provide a lasting solution for your organization.

Why Is Data Architecture Selection Important?

For organizations just getting started with their business intelligence and reporting implementations, data architecture can often be an afterthought. This is especially true when organizations are participating in a “proof of concept” or a “prototype” reporting solution that somehow ends up being promoted to the production environment for business users’ free reign. Unfortunately, postponing or ignoring the data architecture decision can be detrimental to the solution, the business, and sometimes even your role in the organization.

Data architecture could mean different things to different people, although a company’s enterprise architecture typically contains the data architecture. Although the enterprise architecture can include the specific servers and hardware needed, the data architecture is more specific. For the selection process discussed in this chapter, the data architecture includes how the data is stored, accessed, and integrated. Understanding these initial criteria ensures that you don’t run into problems down the road.

DATA ARCHITECTURE

In the context of this chapter, the term *data architecture* encompasses how the data is stored, accessed, and integrated. Data storage includes the location and scalability of the data repositories. Data access contains the models and standards to retrieve that information. Finally, data integration includes how the information moves to and from systems.

Challenges

If your organization does not have a defined data architecture, you will probably face challenges down the road. Although the challenges may not appear immediately, certain phases in the business intelligence implementation will highlight the lack of vision. Hopefully, understanding the challenges will help you recognize the necessity of determining and defining the data architecture before beginning any development.

The challenges include:

- **Slow development and maintenance time:** Without a data architecture, developers will have to reinvent the wheel each time they begin a new project, by evaluating the pros and cons of a new solution. Without the full picture of other solutions and existing systems, you increase your risk of defining a suboptimal solution for not only this solution, but for the entire data ecosystem. Additionally, by designing a new “mini” data

architecture, a developer introduces a different style to the organization, which other developers must learn. The differing styles of solutions can easily cause any new development or changes to existing systems to take much longer than expected.

- **Wasted time due to redevelopment:** If you choose the wrong data architecture, you may need to add one band-aid after another to satisfy the data needs. Business intelligence and reporting are all about the end user, but testing with the end users tends to happen at the end of the development cycle. If the end users have a concern with the amount, type, or frequency of the data they are receiving, you will be up a creek without a paddle! Having a solid data architecture up front means that you can prevent finding these issues at the end of a long development cycle.
- **Loss of confidence in the solution:** A typical symptom of not having a standard data architecture is duplication of information within the system. If a developer does not follow the existing data architecture, he could apply an undesirable band-aid fix. Sometimes this duplication does not result in the same answer, either due to different data in the system or due to user error caused by different access methods. In addition to differing answers, this duplication also causes requests for the information to take longer because end users have to decide which one is the best path. Between different answers and longer query times, the end users soon lose confidence in the system.
- **Lack of scalability:** Providing the ability to appropriately scale a data architecture is a key component of a desirable end-to-end business intelligence solution. Without this ability, the solution will not keep up with the growing business and return times, and its usefulness will slowly deteriorate.

Benefits

In addition to addressing the challenges described in the previous section, defining a data architecture provides benefits. The benefits derived from the data architecture apply to developers, end users, and the entire organization.

The benefits include that it:

- **Fits neatly into the enterprise architecture and ties business and technology together:** Creating a data architecture used by the entire organization ensures that all other architectures can take the data architecture into account. Multiple schools of thought exist surrounding enterprise architecture, but a standard architecture, such as the Zachman Framework,

typically includes the disciplines of data, function, network, people, time, and motivation. Aligning scopes, models, technologies, and functions across all these disciplines makes all areas of the business stronger than without them.

- **Better describes exact cost, time, and efforts to implementation:** By having a defined data architecture, you create a repeatable process for future development. The development process includes an implicit template for adding new data sources and providing reports. You can easily describe what steps will be undertaken and how much time each step takes, introducing both transparency and visibility into the process.

How Do You Pick the Right Data Architecture?

Now that you understand what you lose and gain from having the right data architecture in place, you are ready to begin the process to pick the right data architecture! The process includes multiple steps, which take you from understanding all your options to finalizing and sharing the process with your organization. Figure 3-1 contains the steps in the order to be completed, and the following sections describe each of the steps in the process.

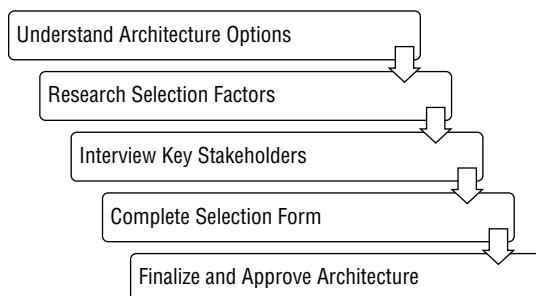


Figure 3-1: Data architecture selection process

Understanding Architecture Options

Starting with an appropriate data architecture is essential to building the rest of the data solution. Similar to building a house, you must create the blueprints before installing the foundation, framing, and utilities. In this scenario, the blueprints are the data architecture, which includes reporting, analytics, or operations.

Some of the more common data architectures that you can implement are detailed in the following sections.

Using an Enterprise Data Warehouse

An *enterprise data warehouse* (EDW) is one of the more commonly known data architectures. First introduced by Bill Inmon, the EDW brings all enterprise information into one central location. Inmon's approach is typically considered "top-down" because it brings all information together first and then disseminates it for reporting. See <http://www.inmoncif.com> for more information.

A second enterprise data warehouse approach is the dimensional data warehouse proposed by Ralph Kimball. Kimball's approach is typically considered "bottom-up" in that it starts by building subject area data marts and then combines the data marts to create a full enterprise solution. For more information on the Kimball approach, see <http://www.kimballgroup.com>.

A sample enterprise data warehouse architecture is shown in Figure 3-2.

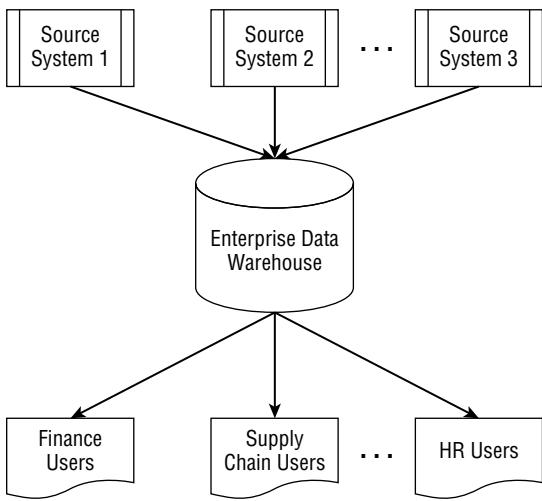


Figure 3-2: Sample enterprise data warehouse architecture

An enterprise data warehouse has the following advantages:

- **Consolidation of disparate reporting systems:** The information from separate systems is brought into a single repository.
- **One version of the truth:** Enterprise information is aligned across all departments and business units.
- **Historical change tracking:** The enterprise data warehouse stores any changes in its source records.

Using an Operational Data Store

An *operational data store* (ODS) is a data repository that combines data from different source systems with the intent of providing a real-time view of the data. Data load programs conform and cleanse the information as it enters the ODS. The source systems can then access the cleansed information for their use. In addition, a data warehouse can pull the information from the operational data store for historical tracking and reporting. A sample data architecture that includes an operational data store is shown in Figure 3-3.

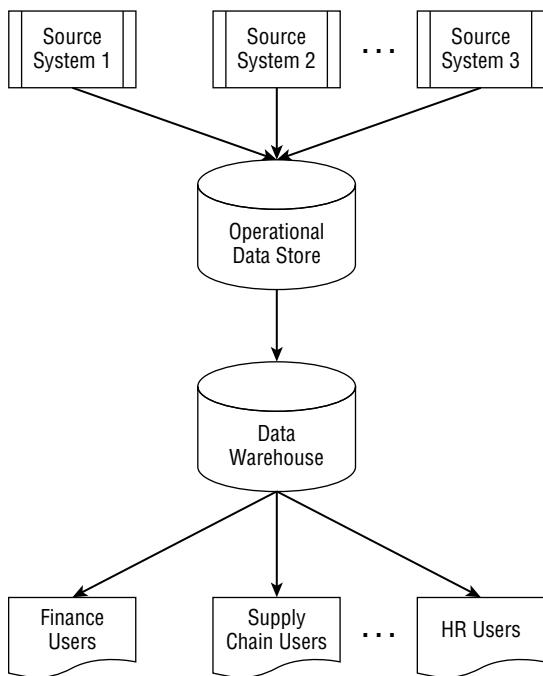


Figure 3-3: Sample operational data store architecture

An important advantage of the operational data store is its ability to provide real-time reporting for operational applications. Keep in mind that the ODS will not contain historical information and typically stores information only for a limited amount of time.

Using a Data Vault

When compared to the enterprise data warehouse and operational data store, the *data vault* architecture is fairly new to the modeling scene. Created by Dan Lindstedt, the model is a cross between a dimensional and third-normal form model. See Figure 3-4 for a sample data architecture that includes a data vault.

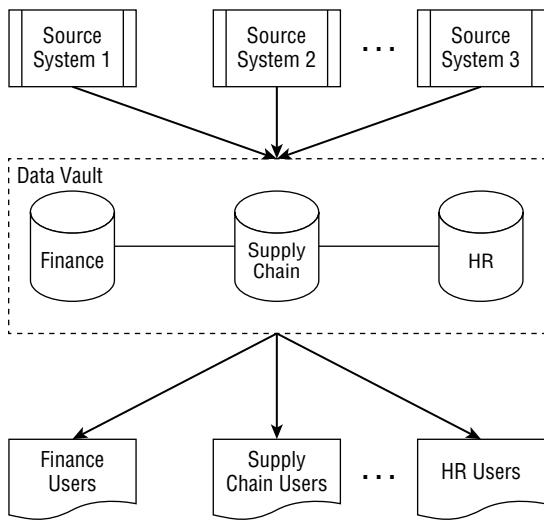


Figure 3-4: Sample data vault architecture

The data vault architecture is best used for systems where the business rules often change, which typically causes a lot of reengineering to the model.

Using Hub and Spoke Data Marts

A hybrid approach between one enterprise data warehouse and many departmental reporting silos, the *hub and spoke* data marts architecture includes one central data hub for information, which feeds separate data marts for each department. Inmon recommends this approach in his Corporate Information Factory (CIF) data architecture. Figure 3-5 shows a sample hub and spoke data model architecture.

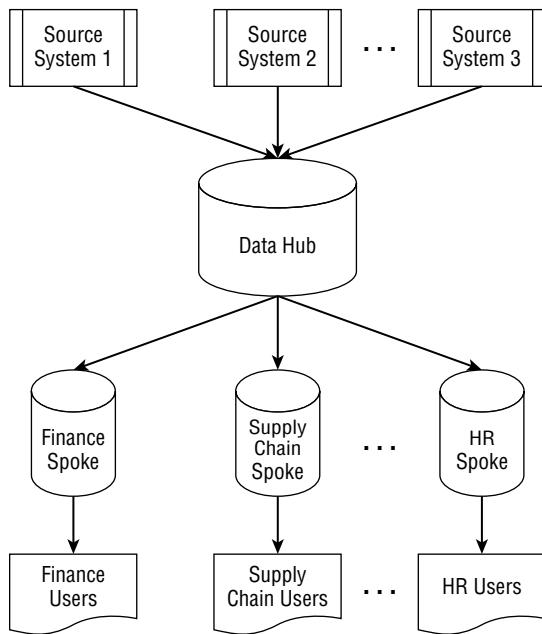


Figure 3-5: Sample hub and spoke data marts architecture

This data architecture has the following benefits:

- Information is stored in a central location.
- Business users have access to their own departmental business logic.
- It's easy to develop and maintain in parallel.

Using Big Data and Late-Binding Models

The last data architecture for you to learn about, *big data*, is typically described by the information that feeds the solution: as having one of the “3-Vs,” which include high volume, high velocity, or high variety. Some attributes often associated with big data include unstructured, NO-SQL, and non-relational. The thought process behind these models is that users may not know exactly what they want to get out of the data when the data is available for storage. So big data solutions store the information as quickly as possible, and any transformations, business logic, or binding to a schema is done at a later time or even as the information is pulled. Keep in mind that the data-access tools are still weak in this area, but will hopefully improve over time.

Microsoft has several tools to help with a big data architecture, such as Azure HDInsight, Parallel Data Warehouse’s Analytics Platform System, Azure DocumentDB, and more. For more information on Microsoft’s big data offerings, visit <http://www.microsoft.com/en-us/server-cloud/solutions/big-data.aspx>.

Additional Factors

Although not considered data architecture per se, be sure to include other factors when considering your data architecture.

Corporate versus Self-Service Analysis

The advent of in-memory applications and reporting has made *self-service analysis* easier than before. Although it could mean different things for different people, for this chapter, self-service analysis means consumers can retrieve the desired information in any fashion. Breaking that sentence down into multiple pieces makes three main phrases:

- **Desired information:** This information means metrics or properties that users want.
- **Any fashion:** This means that the information can be displayed or analyzed in different formats.
- **Consumer:** Self-service analysis is all about the end user/consumer.

In general, self-service analysis could be as simple as a report that allows the end user to filter on a few values and then displays the output of a standard formula. Or it could mean an interface that allows end users to drag and drop fields to create their own view of the data. On the extreme side, self-service analysis could allow end users to create their formulas based on any or all values, even allowing them to pull new values.

On the other hand, corporate analysis tends to be more static and conformed. This is the corporate reporting that organizations put in place to satisfy external reporting, organizational scorecards, and internal processes. Corporate analysis ensures that the data reflects one version of the truth.

Organizations use both corporate and self-service reporting and analysis quite successfully. Some organizations tend to use one over the other, whereas some use a hybrid approach of both. As you drill into the chapter more, you will be able to better determine the right type for your organization.

Cloud versus On-Premise Systems

Another factor to keep in mind is the location of your data architecture. Cloud-based servers, databases, and services allow your organization to focus on moving and integrating the data without having to manage the day-to-day hardware and software operations, such as updates and maintenance. Additionally, these systems can often provide additional processing power only when required, ensuring that you have the power you need when you need it, without having to pay for it all the time.

On-premise systems mean that you have the servers and databases in-house, which could be in your building or a separate system center. You have complete

control of your server and can upgrade or modify it when it suits you. If you have a known data size, an on-premise solution could be cheaper in the long run.

Of course, hybrid approaches exist, where only some servers, databases, or services live in the cloud and others live on-premise. You need to closely evaluate the best option for your organization.

Understanding Research Selection Factors

Once you understand the available architecture options, your next step is to understand the information that will factor into the data architecture selection process. Gaining a complete view of the existing organization, systems, and processes and any planned or proposed changes ensures that the data architecture will satisfy all existing and future needs of the organization. This section discusses the different selection factors that you must consider when making your data architecture decision.

How Organizational Structure and Size Impact Architecture

The industry, size, and risk tolerance level of your organization can greatly impact the ideal data architecture for your organization. Although a generalization, a larger organization often has more mandatory rules and standards in place to implement any changes to a system or application. These rules could be about change control, deployments, or documentation. On the other hand, smaller organizations can use more of the “cowboy-coding” paradigm, where change can happen as quickly as needed, sometimes even by changing code directly in a production environment!

Similarly, your organization’s industry can also affect the decision on the ideal data architecture. You must adhere to and closely monitor any standards or laws that are in place which affect the storage or transfer of data, such as Sarbanes-Oxley (SOX) or the Health Insurance Portability and Accountability Act (HIPAA).

How System Configuration, Maintenance, and Fragility Impact Architecture

In addition to the organizational structure factor, the configuration and maintenance of systems within the organization can also affect the data architecture decision. The system configuration includes the number of applications and databases needed to run the business, the complexity of the systems’ interactions, and the difference between the systems. For example, some organizations have one large system, such as an SAP enterprise solution, that contains multiple integrated components to handle anything from customer relations to supply-chain management processes. Other organizations may have numerous small home-grown applications, which use the proverbial duct tape fixes to make

them work together over time. Of course, many organizations run the gamut between those two extremes.

Understanding the maintenance and fragility of the systems is important to factor in the data architecture decision as well. System maintenance includes how often the existing applications break, how long they are down, how many people can fix the issue, and much more. System fragility includes the maintenance aspect as well as the people side of things. For example, if the system often changes due to new business requirements, the system is quite fragile. Also, if the system never changes and, in fact, no one knows where the original source code is, the system is also quite fragile.

Any existing technology, system, or architecture standards can directly impact your data architecture decision. Existing standards directly tie into system configuration, maintenance, and fragility because you must know if existing systems follow the standards or not. Depending on the level of compliance to existing standards and the desire to continue these standards will greatly affect your decision.

How Data Repositories, Complexity, and Volume Impact Architecture

The next selection factor centers around the data needs of the organization. Whereas applications are dependent on data, the reverse is not true, so you can consider data repositories separately. When assessing your data repositories, you need to consider the number of available data repositories, the complexity of the data and their communications, and the amount of data needed to run the business. In addition, you must consider the overall maturity level of the reporting and analysis required by the organization.

Reporting databases exist in all shapes and sizes, including relational databases, NoSQL databases, and file-based “databases,” such as Excel or Access. Understanding the structure of those repositories are as important as the objectives of choosing those repositories. Some questions to guide you in choosing repositories include:

- Do end users use the information for real-time or self-serve reporting?
- Do end users use the database to manually cleanse data?
- Do end users feel comfortable with the amount of data they have?

How Business Goals and Requirements Impact Architecture

Finally, you must understand the business’s current processes, their goals and strategy for the future, and any requirements that stem from these. In many organizations, the technical and business teams plan in different meetings and are not always aware of the “other side” of the house’s tasks. By talking to the business, you will learn of all business plans relevant to the data architecture decision.

The business's current processes do not always mimic the system processes that you learned about from the technical stakeholders. Some organizations will purchase COTS (commercial off-the-shelf) products that handle only a percentage of what the business needs. To supplement the process, the business will create additional systems and workarounds to satisfy their needs. There is a good chance that data elements captured in these "one-off" or "temporary" solutions are actually very important to know about in the data architecture.

Depending on your organization, you may also face the scenario where the business side of the organization pays for the technical implementations. This payment could be in the form of additional resources, time, or even monetary budget. Knowing how much the business will spend is an important factor in which data architecture you choose.

Finally, you have to take into account the organization's business strategy. If the business is currently a brick and mortar business with a small specialty-based product and clientele, you may choose one type of data architecture. If the business is (or is planning to become) an international Internet-only store, you would choose another data architecture entirely. Be sure to focus on current needs and the achievable goals over the next several years.

Interviewing Key Stakeholders

Once you understand the different available data architecture options, your next step is to understand the goals of the organization through interviewing key stakeholders. Although the interview step is often skipped when making the decision of what data architecture to use, it greatly factors into the final decision. By interviewing a variety of people in the organization, you'll develop a view of the organizational strategy and goals. You wouldn't want to make a cake without knowing what size oven is available or without knowing if you had a refrigerator to store ingredients. In the same way, you need to learn all potential factors by interviewing company stakeholders.

The represented stakeholders that you interview should include anyone who touches any of the data or systems in the organization. You should have enough representatives that you can get a view of the entire organization, so even planning the interview schedule can be a task within itself!

Start the interview planning with the organizational chart, which contains each of the departments and the employees within each department in the organization. From this chart, you can put together a list of each of the departments that you must cover in the interview process. Then, you can determine who in the organization has the knowledge to cover each department. Also, you should learn about each system that helps run the organization. The technical side of the organization is important in this part of the interview process. Start with the head of the information technology (IT) department and find out who is the owner of the system; it is quite possible and common for some systems

to have no owner. If so, try to talk to someone who's done some development on the system or find someone who's been with the organization for a while and can give some sort of background on the system. Not only can you gain the information you need for your decision, you can also highlight some of the potential holes in system management in your organization.

Be sure to interview the following groups of stakeholders and focus your questions on their jobs and their focus:

- Technical staff
- Business owners
- External partners

Completing the Selection Form

The purpose of interviewing the stakeholders in the organization is to obtain all information necessary to fill out the selection form. After that you will then be ready to pick the right data architecture. Keep in mind that it is not as easy as two plus two equals four. In fact, oftentimes you will find the result ends up closer to three or even five. This section shows you how to use a standard methodology to select a data architecture, but the important thing to remember about making your decision is to use the data architecture that makes the most sense to you and your organization.

The selection form shown in Figure 3-6 lists the selection factors discussed previously in this chapter. This form allows you to answer True or False to discover if the factor applies to your organization. You can use those answers to work through the selection flow chart to determine the best possible data architecture for your organization. Feel free to modify the list of factors and flow to best suit your needs; this only gives you a template from which to start.

Selection Factors	Score (True, False, N/A)
Organization Structure and Size	
Single development center	
Large organization	
Industry with many regulations or policies	
System Configuration, Maintenance, and Fragility	
High volume, velocity or variety of data	
Large number of applications or many different application vendors	
Unknown scale for application data or processing	
Data Repositories, Complexity, and Volume	
Ad-hoc and/or self-service analysis	
Record change tracking	
Real-time, operational data access	
Business Goals and Requirements	
Departmental business logic	
Limited cost and resources	
Rapidly changing business logic	

Figure 3-6: Selection form

Once you have answered the questions in the selection form, you can follow the selection flow chart, shown in Figure 3-7. This chart contains three separate chains, which focus on the data architecture type and the additional factors previously discussed. Start at the top of each diamond chain, and based on the factor in the diamond, follow the arrow to the right if the factor is true, and follow the arrow to the left if the factor is false. When you reach an oval, you can add the result to your data architecture list. The end result of the process flow provides a three-part focus for the data architecture, such as a big data, corporate, on-premise solution or an enterprise data warehouse, self-service, cloud solution.

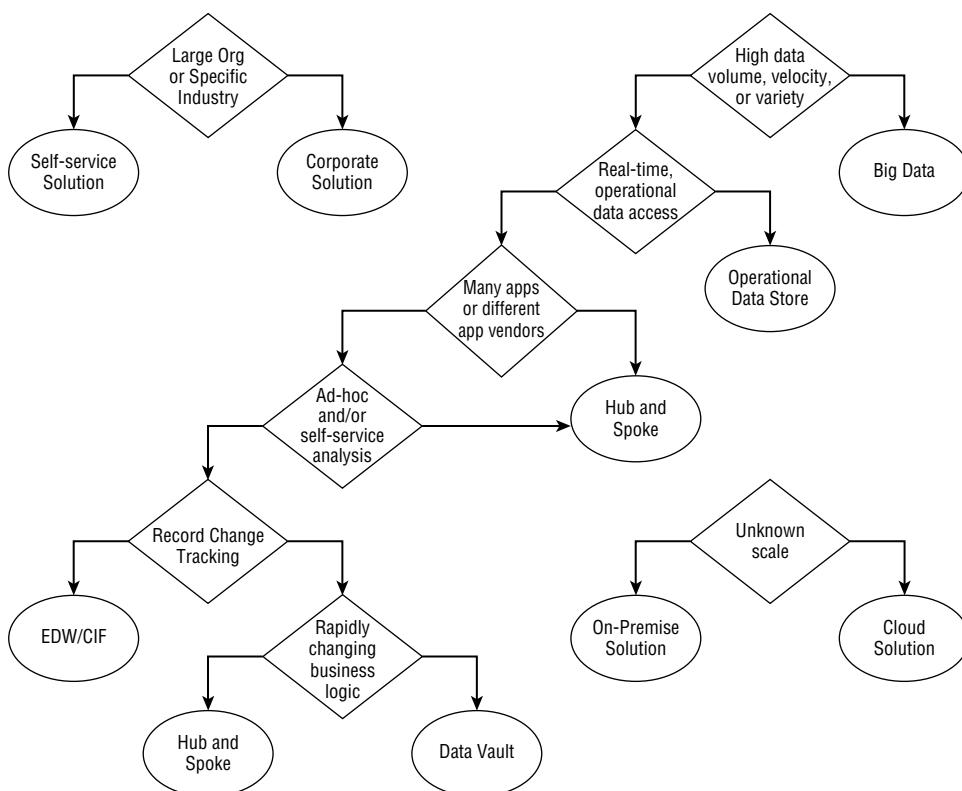


Figure 3-7: Selection flow chart

Finalizing and Approving the Architecture

Congratulations, you have finally decided on the right data architecture for your organization! Now you begin the real work: convincing everyone of the decision. Although you might be tempted, do not skip this step. Not only must

the data architecture be defined, it also needs to be followed. The best way to do that is to get everyone's buy-in to the solution.

Start by writing up the process you undertook to determine this data architecture. The process document should include all the stakeholders you interviewed and the high-level answers you received. Don't call a particular person out if you received negative information, but be sure to highlight both the pros and cons you heard during the interviews. Additionally, include an overview of the selection factors and the completed selection matrix. Seeing a summarized view of the information of the organization should resonate with your audience.

Once you have written the final document, meet with your manager or director to review the document. There are a couple reasons for this:

- For a sanity check on your decision
- To ensure you have an ally when you promote and market this to the rest of the organization

Once your manager is on board, it is time to take the document public.

You already know the people who are most interested in the data architecture because they were your original stakeholders for interviews. Because they participated in the interviews, some of the information should look familiar to them. Oftentimes, the stakeholders are surprised about what other stakeholders said—whether in adamant agreement or complete disagreement. This is the time to get everyone in the same room and let them talk it out. Once they hear everyone's point of view, they should realize that the data architecture takes the entire organization into account.

In addition to them talking through the information they provided, you will explain to them how the new data architecture meets and answers their needs. For example, if they were very concerned about real-time data, explain how the ad-hoc self-service solution you selected lets them pull information on their schedule. Or, if they were most vocal about aligning the data structures with the fast-changing business, highlight how the data vault model lets the data move with the business.

Although not for every organization, an official "Robert's Rules of Order" vote can help cement the decision in everyone's minds, reinforcing the importance of following the new data architecture. Once officially approved, store the decision document where everyone can access it, and get ready to start implementing your chosen data architecture.

Summary

In this chapter, you learned about different data architectures you can apply to your organization. You know the benefits of having and the challenges of not having a defined data architecture. Additionally, you know the steps to take to help define that organizational data architecture.

The process described in this chapter is not only for consultants. Any data architect, developer, or database administrator can follow them to determine the correct data architecture for their organization. It is up to you to follow the steps and apply the principles to make the best possible decision.

Chapter 4 discusses Power Query, which will help you load and cleanse the data going into the data architecture you just selected.



Business Intelligence for Analysis

In This Part

Chapter 4: Searching and Combing Data with Power Query

Chapter 5: Choosing the Right Semantic Model

Chapter 6: Discovering and Analyzing Data with Power Pivot

Chapter 7: Developing a Flexible and Scalable Tabular Model

Chapter 8: Developing a Flexible and Scalable Multidimensional Model

Chapter 9: Discovering Knowledge with Data Mining

Searching and Combining Data with Power Query

After making all the decisions regarding the data warehouse, end users can begin consuming the data using one or several types of technologies but some users may realize that the technology does not contain all the data they need to solve, analyze, or answer particular questions. Typically, this means the end user enters a request asking the business intelligence team to modify the existing data warehouse structure to accommodate the change. This change request enters into a queue, and after a period of time, it makes its way into both the data warehouse and the hands of the end user.

Between the time of the request and the actually delivery, the end user may have solved the problem using several different technologies and steps, for example, they may have:

1. Copied, emailed, or downloaded the data from the source
2. Shaped the data into a format that meets their needs
3. Performed a data validation step
4. Figured out how to combine new data with existing data in the data warehouse

Imagine repeating these steps for each dataset that does not exist in the data warehouse. This may not only frustrate the end users when they acquire new

data, but it may also shift the user's perception of the warehouse from one of a single centralized source to that of an incomplete dataset and project.

To mitigate this, Microsoft introduced the Power Query add-in for Excel.

DEFINITION Defined several different ways since its initial release, *Power Query* is an end-user self-service data discovery, extraction, and manipulation tool. It allows individuals to extract data from traditional sources, such as relational databases and Excel files. However, you can also import data from Odata feeds, web pages, and social media websites, to mention a few. As a result, instead of waiting on the team business intelligence to incorporate the requested data into the data warehouse, a data consumer can quickly import, model, and couple the data with existing data. All this happens via a typically available and familiar tool: Microsoft Excel.

Downloading and Installing Power Query

Before discussing how to use Power Query, this section shows how to install the add-in so that it is available inside Excel. By way of requirements, the hosting computer must be running one of the following operating systems (OS):

- Windows 7
- Windows 8/8.1
- Windows Server 2008 R2
- Windows Server 2012

In addition, Power Query is supported only by these versions of Microsoft Excel:

- Microsoft Office 2012 Professional Plus with Software Assurance
- Microsoft Office 2013 Professional Plus
- Microsoft Office 2013 Office 365 Professional Plus
- Excel 2013 Standalone

After you verify that the OS and Office/Excel version meet the minimum system requirements, you can download and install Power Query, which is available for 32- and 64-bit platforms. During the download, make sure you select the correct version.

The steps for downloading and installing Power Query are as follows:

1. **Download Power Query.** Navigate to <http://www.microsoft.com/en-us/download/details.aspx?id=39379&CorrelationId=66549b07-d2a5-4c18-bd25-6318c164dcfd>, and click the Download button (see Figure 4-1).

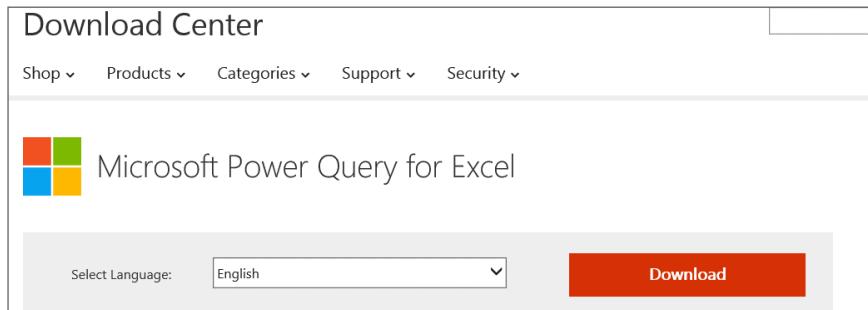


Figure 4-1: Microsoft's download center

2. **Initiate the download.** After downloading, the install should begin. Click Next on the Welcome window (see Figure 4-2).



Figure 4-2: The Power Query wizard

3. **Accept the License Agreement.** Select I accept the terms in the License Agreement checkbox on the Microsoft Software License Terms windows (see Figure 4-3).

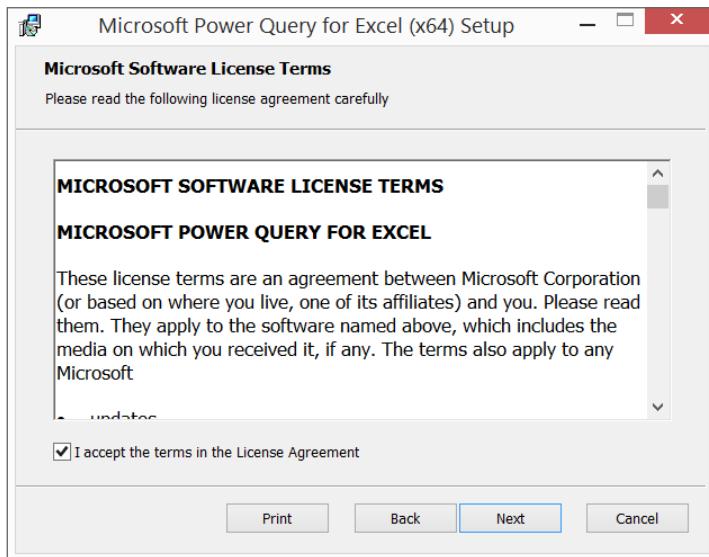


Figure 4-3: The Software License Terms window

4. **Define Power Query's location.** Specify the location where you want to install Power Query on the Destination Folder (see Figure 4-4).

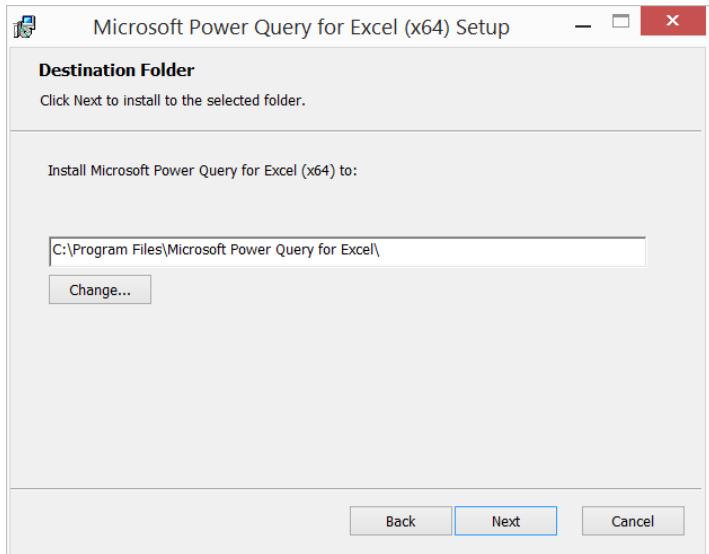


Figure 4-4: Defining the destination folder

5. **Install Power Query.** Click Install on the Ready to install window (see Figure 4-5).

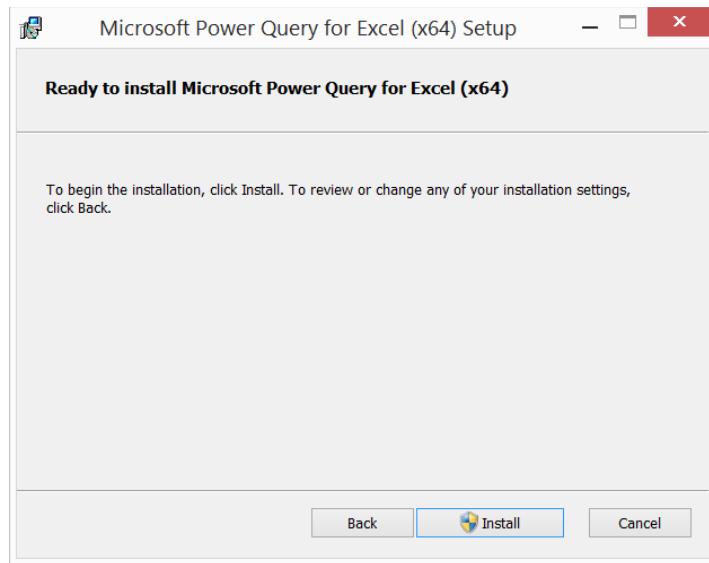


Figure 4-5: The Ready to install Microsoft Power Query for the Excel window

When the install is complete, a new option becomes available in the Excel ribbon labeled Power Query (see Figure 4-6).

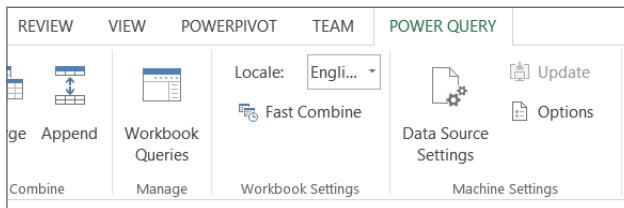


Figure 4-6: View of Power Query in the Excel ribbon

NOTE Power Query version 2.15.3722.242 was used during the writing of this book. To check the version you have currently installed, go to Power Query in the Excel ribbon and click About in the Help section.

Importing Data

As stated earlier, Power Query allows users to import data from a plethora of sources. End users are no longer at the mercy of IT when data issues or changes arise. Instead, by leveraging Excel, you can quickly access data via the Power Query add-in. Figure 4-7 shows the Power Query ribbon's Get External Data section.

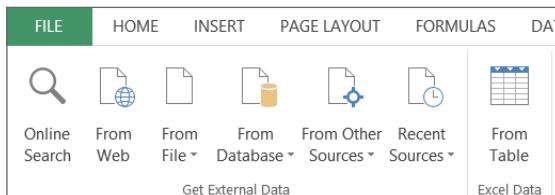


Figure 4-7: List of Data Import Source choices in the Power Query ribbon

Notice that you can import data by searching the web, local files, or databases. In addition to these more traditional sources, Power Query ventures into a new realm of data extracting by enabling users direct access to sources such as Hadoop, Active Directly, Facebook, and more. Figure 4-8 gives a complete list of all "Other" sources that you can access using Power Query.

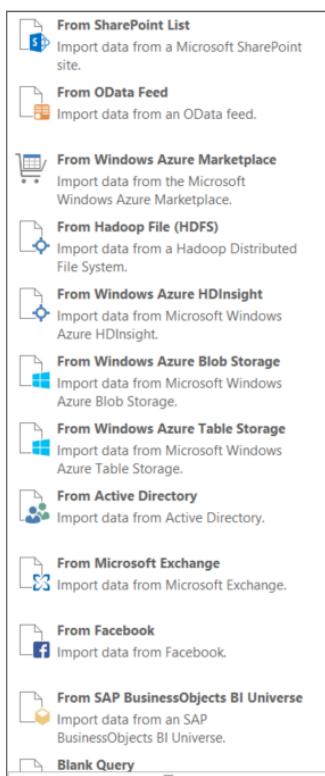


Figure 4-8: List of Other Data Source types

Importing from a Database

The steps for importing data from most of the database sources are pretty similar:

1. **Select the From Database option from the Get External Data section of the Power Query ribbon.** Refer to Figure 4-7 for this option.
2. **From the list of available choices, select the relational database source you want.** For example, if you need the data from a SQL Server database, select From SQL Server Data from the list of available choices. After you make your selection, the appropriate window appears requesting necessary information. In this example, you see the Microsoft SQL Database dialog window appear (see Figure 4-9).

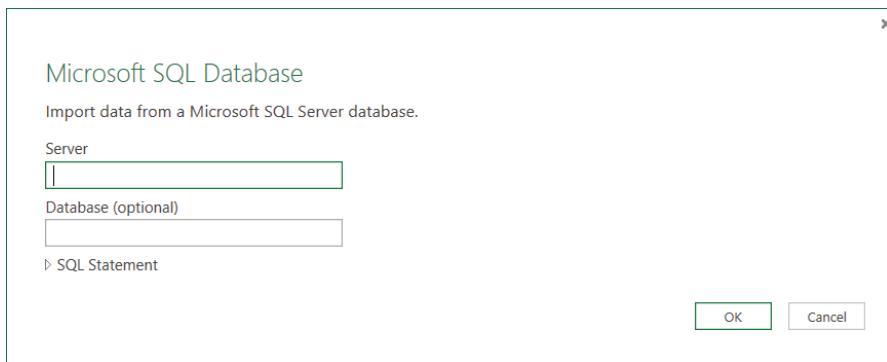


Figure 4-9: Microsoft SQL Database import dialog window

3. **Fill in the required information about the relational database source.** For this example, the Microsoft SQL screen requires the server name but the database name is optional. In addition, you may provide a query. If the database name is omitted, Power Query provides a list of databases. However, if a database and/or query is provided, this list is filtered down to a single database or query. Figure 4-10 shows the three different views, labeled accordingly.

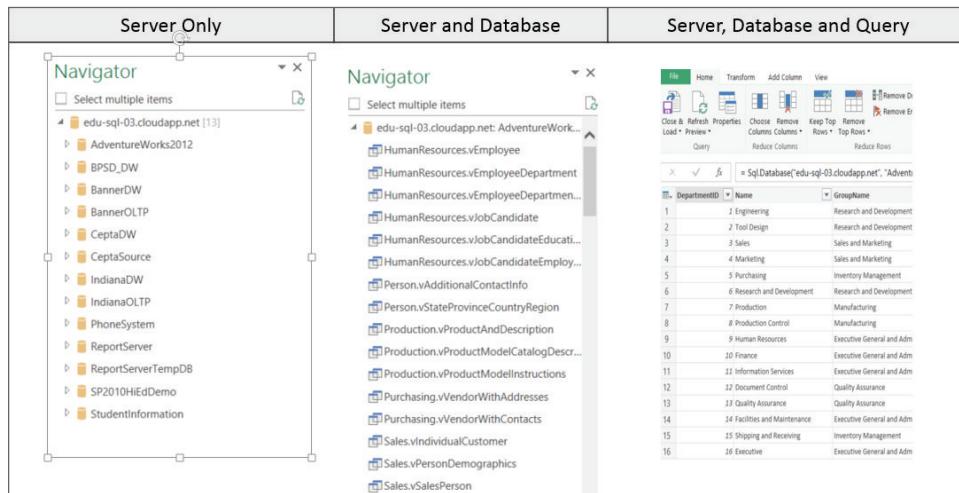


Figure 4-10: Different views when importing data from a relational data source

4. At this point, the process of importing and modeling the data is similar across all sources. The Workbook Queries window opens from the right side of Excel, and, if only the server name was provided, expands the database to show a list of available objects. On the other hand, if a database or query were provided, then only a list of objects appears. Either way, by hovering over an object, a preview of the data in the object should appear (see Figure 4-11). As seen in the image, when entering a query, the Power Query Editor window opens, displaying the data from the query.

The screenshot shows the Workbook Queries window with the following details:

- Left Panel:** A preview of the "HumanResources.Department" table. It contains 9 rows with columns: DepartmentID, Name, and GroupName. The data includes rows for Engineering, Tool Design, Sales, Marketing, Purchasing, Research and Development, Production, Production Control, and Human Resources.
- Bottom Left:** A "Columns [5]" section listing the columns: DepartmentID, Name, GroupName, ModifiedDate, and HumanResources.EmployeeDepartmentHistory.
- Bottom Center:** A "EDIT" button.
- Right Panel:** A list of available objects in the current database, including Sales.vSalesPerson, Sales.vStoreWith, AWBuildVersion, DatabaseLog, ErrorLog, and several HumanResource and Person.Address entries.
- Bottom Right:** Buttons for "Edit" and "Load".

Figure 4-11: Preview of data when hovering over an object

5. Once the preferred object has been identified, the fun begins. By right-clicking an object, the user is presented with a few options:
 - **Edit:** Opens the Power Query windows.
 - **Load:** Imports the data directly into the Excel workbook.
 - **Load To:** Opens a dialog window (Figure 4-12), presenting the users with even more choices.

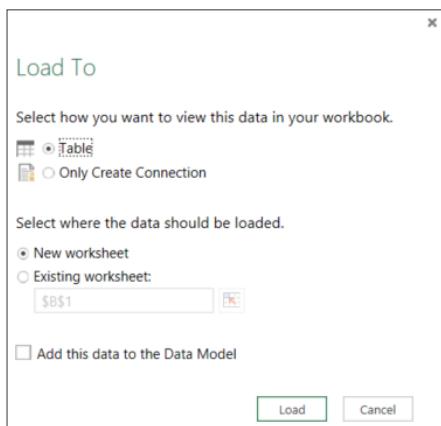


Figure 4-12: Load To Properties window

The Load To dialog window's options are:

- **Select how you want to view this data in your workbook:** The Table option simply loads data into your workbook. The Only Create Connection option does exactly what the choice says: It creates a connection in the spreadsheet.
- **Select where the data should be loaded:** If you selected the Table option, Power Query chooses where in the workbook to load the data. If you selected Only Create Connection, Power Query loads the data into a data model. If load data to data model checkbox is checked, the data is loaded into a Power Pivot model; Power Pivot will be discussed at length in Chapter 6.

For this example, ensure that you select the Only Create Connection radio button and the Add this data to the Data Model checkbox. After making these choices, the end user clicks the button labeled Load. The data will load into the Power Pivot model.

Importing from the Web

You can use Power Query to pull data from the web in several ways, including performing an online search, or directly using a URL or other sources such as

Facebook, SharePoint lists, Odata feeds, and the many Windows Azure choices. This capability presents the user with almost boundless sources of data.

For example, assume users need data that reflects the total population by state for the United States of America. Before Power Query, how would you identify and import this data? The solution could be accomplished via a number of steps, but with Power Query, solving the problem is as simple as doing a Bing search:

- 1. Click Online Search in the Power Query ribbon.** The Online Search pane appears on the left of the Excel window.
- 2. In the Search text box, type us state population (see Figure 4-13).** After the search is complete, Power Query presents a result list.
- 3. Repeat the steps outlined in the From Database section to preview, edit, or load the data at this point.** The primary difference is that Power Query presents the user with an Internet search-result list rather than a list of database objects.

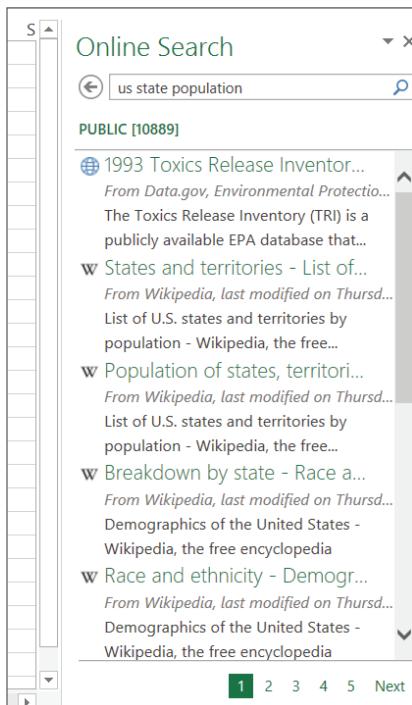


Figure 4-13: Power Query Online Search window with results

NOTE Additional features and capabilities are available if the end user has a subscription to Office 365 and a Power BI license. These are beyond the scope of this book, but you can find more information at <http://www.powerbi.com>.

Importing from a File

Importing data from Excel and comma-delimited files is a common practice among data analysts, so what would a data-manipulation tool be without this ability? Figure 4-14 shows the list of available file types that you can import using Power Query.

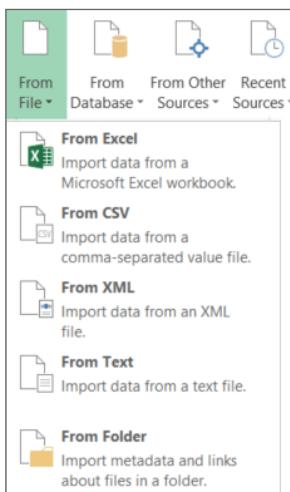


Figure 4-14: List of file types

The last item in the list, **From Folder**, is not exactly a file type. It is unique in the sense of how it imports files. In addition to importing single files, this option can combine multiple files into a single result set. The folder must contain files that have the same structure.

When importing a single file, the Power Query window automatically opens, displaying a sample of the data from the file. However, if the **From File** option is selected, instead of a view of the data in the Power Query window, the end user sees a list of the files in the folder, shown in Figure 4-15.

= Folder.Files("C:\Users\pleblanc\Documents\My Projects\EDU Power BI")					
	Content	Name	Extension	Date accessed	Date modified
1	Binary	Contoso Performance 2 (Autosaved).xlsx	.xlsx	6/11/2014 7:11:42 AM	6/11/2014 7:11:43 AM
2	Binary	Contoso Performance 2.xlsx	.xlsx	11/17/2014 9:33:40 PM	11/17/2014 9:33:40 PM
3	Binary	Contoso Performance.xlsx	.xlsx	6/4/2014 12:20:48 PM	6/4/2014 12:20:49 PM

Figure 4-15: List of files imported using the From File option

To view and combine the data, click the icon located in the Content header circled in red in Figure 4-15.

Transforming Data

With all the data now imported, you can begin to model and transform the data in a way that was difficult or impossible prior to the addition of Power Query within Excel. For example, how do you combine two disparate datasets into one using Excel alone? The first thought that comes to mind is to use VLOOKUP, which combines the two datasets based on a key value and returns a single value on a row-by-row basis.

Another common scenario is deriving multiple columns from a single column delimited by some character such as a comma or tab. By using some combination of string manipulation functions like LEFT, RIGHT, LEN, or FIND, you may be able to solve the problem.

The disadvantage to both these approaches is that they require not only knowledge of the Excel expression language, but also familiarity with the accompanying functions. Although a large population of individuals do possess this knowledge, there are just as many people who do not. Hence the need for Power Query. Certain aspects and features of Power Query are discussed in the following section. The next couple of sections detail how Power Query can simplify much of this work.

Combining Data from Multiple Sources

As mentioned earlier, a very common task in Excel involves combining data from two sources. Typically this is done using a VLOOKUP function. Now, with Power Query, combining data from two sources is a simple process that does not require any coding. Assume that some data has been imported from a single CSV file and a SQL Server Database. The CSV file contains a list of states and state names, whereas the database table contains a list of addresses. Before data analysis can begin, there is a need to merge a couple of columns to the database table from the CSV file.

You can perform this task in a few ways: directly from the Power Query ribbon, by right-clicking one of the Power Queries in the Workbook Queries pane, or from the Power Query Editor. The following steps use the Power Query ribbon:

1. Click the button labeled **Merge** in the **Combine** section of the Power Query ribbon. The Merge dialog window opens, shown in Figure 4-16.

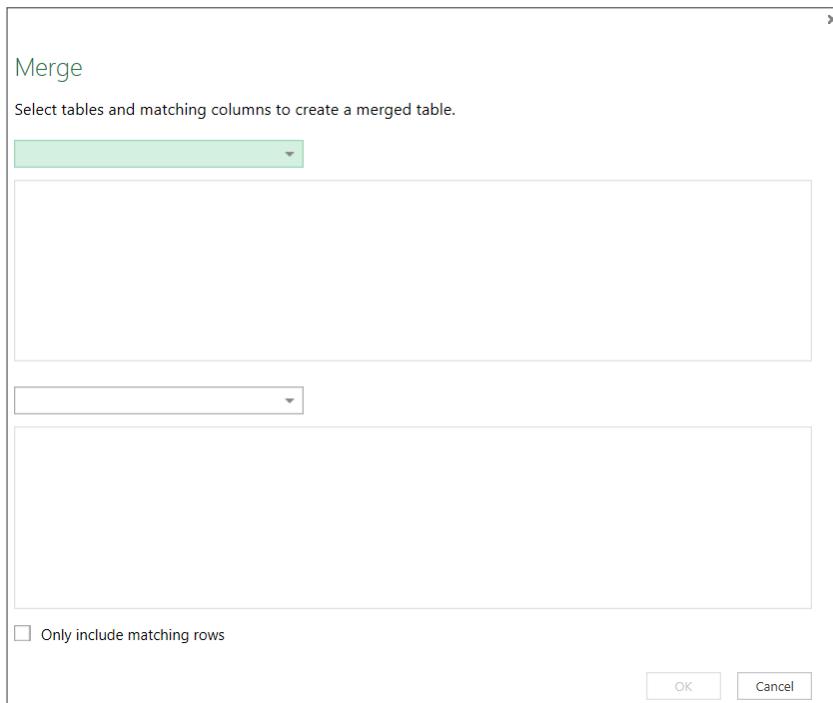


Figure 4-16: Power Query Merge window

2. **Select a Power Query in each drop-down box.** From the data preview, select a single column or multiple columns. Make sure that these are the columns that contain matching data across the datasets. These would typically be the columns selected in a foreign key relationship. After the selections are made, Power Query generates a message providing the number of matched rows.
3. **Click OK and the Query Editor opens.** The data preview shows the data from the dataset selected in the first drop-down box, and also an additional column named NewColumn. Just to the right of the name is a box containing two arrows pointing in opposite directions. The word Table is repeated as each row value. All this is shown in Figure 4-17.

	Modified Date	New Column
5C2D4EC6E9	1/4/2002 12:00:00 AM	Table
LCDE60D8C0	1/1/2003 12:00:00 AM	Table
A6F5B142A4	4/8/2007 12:00:00 AM	Table
D9DE16A880	3/7/2003 12:00:00 AM	Table
F9900E9BB0	1/20/2003 12:00:00 AM	Table
DE94F0075	3/17/2003 12:00:00 AM	Table
19CD6A0446	1/12/2004 12:00:00 AM	Table
357D21F269	1/18/2003 12:00:00 AM	Table
4FD139821A	7/1/2006 12:00:00 AM	Table
196AF34C04	1/3/2003 12:00:00 AM	Table
670785B269	4/1/2007 12:00:00 AM	Table
66B6F7F22	2/6/2007 12:00:00 AM	Table
FC67D28AE4	1/15/2008 12:00:00 AM	Table
50D39864D1	1/5/2004 12:00:00 AM	Table
D928ACAD6	12/20/2007 12:00:00 AM	Table

Figure 4-17: Merge Operations showing unexpanded table

To see or expand the data in the table, click the two arrows (Expand icon). Power Query presents a list of the columns available in the table.

4. Select the desired columns and click OK. This adds them to the result set.

Splitting Data

When working with data, you commonly encounter a value in a single cell that is separated by a delimiter such as comma, tab, or semicolon. Often the values need to be split into individual columns based on the delimiter. For example, assume you have a spreadsheet similar to the one shown in Figure 4-18.

A
1 CityStateZip
2 Houston, TX 77016
3 Boise, ID 83702
4 Baton Rouge, LA 77016
5 Alpharetta, GA 30005
6 Milton, GA 30004
7

Figure 4-18: View of comma-delimited data in a single cell

Notice that the column labeled CityStateZip contains a fully qualified value, such as Houston, TX 77016. Given its current form, analyzing the data individually for City, State, or Zip code would be impossible; you need to split the data into three separate columns. But Excel and its accompanying expression

language makes solving this problem easy, especially if you use Excel regularly. However, not everyone does, which is not to take anything away from the true power and value of Excel. This context is only setting the stage for how simple and intuitive Power Query truly is.

Splitting this data using Power Query is as simple as clicking a button (depending on what you are trying to split, of course). The previous example of splitting City, State, and Zip using Excel expressions would involve several different function combinations:

1. After you have imported data into Power Query, click the arrow under the Split Column icon on the Transform section of the Power Query Editor window. Select By Delimiter from the list of available choices. The Split column by delimiter dialog window appears.
2. Select how to split the data by choosing an option from the Select or enter delimiter drop-down box. Select from Power Query's extensive list of choices, or enter a custom value.

NOTE The Split radio button list presents the greatest problem because the available options may not seem to match every real-life scenario. Although the complexity of the data you are working with may require some coding, these options should work for most data-split situations.

3. **Split data into columns.** Returning to the CityStateZip example, shown in Figure 4-18, you want to split this data into three separate columns (City, State, and Zip). To create a separate column for City, select the Comma from the drop-down list and the At the left-most delimiter option. The combination detects the first occurrence of a comma, reading from left to right. The result should be two columns: CityStateZip.1 and CityStateZip.2, or some variation of that depending on the column names.
4. **Rename the column.** For example, to rename the column that contains only City, right-click the header and select Rename from the list of available choices. It's best not to rename any more columns until the final split is done.
5. **Split data into columns.** Click the header from the CityStateZip.2 column. Click the arrow below the Split Column icon in the Tranform section. Select By Delimiter from the list of available choices. Select Space from the drop down list and At the right-most delimiter. Click OK.
6. **Rename CityStateZip2.1 to State and CityStateZip2.2 to PostalCode.**

Aggregating Data

Power Query has so far proven to be an ally in importing and combining data, but what about adding information that does not exist, such as aggregated data. When analyzing data, additive and semi-additive values play an integral role in uncovering trends and anomalies. Actually without them, you cannot really perform much analysis. To meet this need, Power Query provides two very intuitive approaches to aggregating data. Both methods accomplish the exact same thing, and using one over the other will eventually become a matter of preference.

Aggregating Data Using the Group By Option

In this section, the Group By option is used to aggregate data down to a level that was not initially presented when the data was imported.

1. **Start by importing the Sales.SalesOrderDetail table from the AdventureWorks2012 database.** This table contains corresponding line items for each sale in the Sales.SalesOrder table.
2. **In the Group By dialog box, start grouping your columns.** During the data analysis process, you'll see that instead of detailed information, the process requires a count of each line item for each sale. To solve this problem, in Power Query click the SalesOrderID column header in the data preview area, then click Transform in the ribbon. Click the Group By icon, located on the far left, and the Group By dialog window opens (see Figure 4-19).

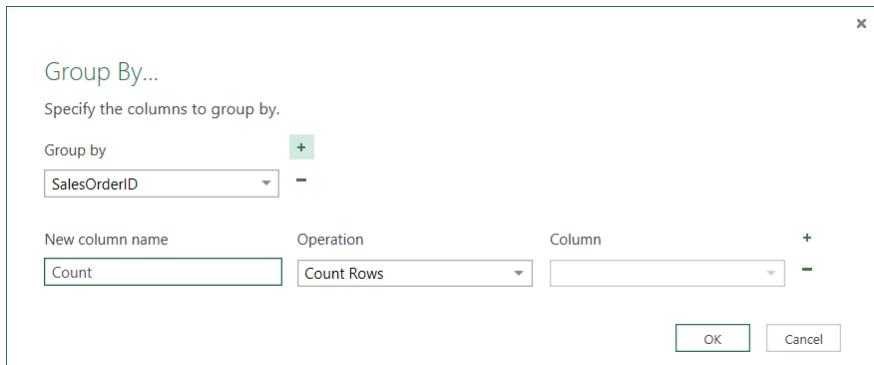


Figure 4-19: Power Query Group By dialog window

3. **Arrange the columns.** By default, the selected column appears as the specified Group By column. To change the column, simply choose a new

column from the Group drop-down list. For now, accept SalesOrderID, or make sure that SalesOrderID is selected.

4. If additional groupings are required, click the plus (+) sign located to the left of the same drop-down list. If a group must be removed, click the minus (-) sign directly below the plus sign.
5. Select how to aggregate the data. You do this by entering a column name in the New column name box, and select what type of operation will be performed in the Operation drop-down list. Enter Line Items in the text box, and select Count Rows from the drop-down list.
6. If additional aggregations are needed, click the plus (+) column located to the far left of the window, directly to the right of the Column drop-down list. If the aggregation needs to be removed, click the minus (-) located directly below the aforementioned plus (+) sign.
7. Once these steps are complete, click OK. A preview of the results appears (see Figure 4-20).

SalesOrderID	Line Items
1	12
2	2
3	15
4	22
5	1
6	8
7	10
8	6
9	4
10	29
11	1

Figure 4-20: Aggregated data using Group By

Notice that the Group By process has modified the data so that each sales order is associated with its corresponding number of line items. This is the first method.

Aggregating Data Using the Transform Ribbon

Similar to many features in Power Query, another way to perform aggregations exists. This one, although not as intuitive as the Group By approach, is

still a very effective technique. Power Query includes a ribbon that is dedicated to transforming and shaping data. Even further, there is a section, Structure Column, that is completely devoted to expanding tables and aggregating values. The following section gives steps that show how to aggregate data using this feature to obtain the exact same result set seen in Figure 4-20, follow these steps:

1. **In the Power Query edit window, import the Sales.SalesOrderHeader table that is included in the AdventureWorks2012 database.**
2. **Open the Power Query, Query Editor window.**
3. **Click Transform in the ribbon.** In the Structured Column section, locate and click the Aggregate icon. Wait, it's not enabled. This is where Power Query lacks a little of its innate intuitiveness.
4. **To enable the icon, scroll through the list of columns and locate the Sales.SalesOrderDetail column.** Notice that the expand icon, shown in Figure 4-17, is present in several of the columns. By default, when importing data from a relational source into Power Query, a column that contains the relationship also appears in the field list for each table.
5. **Click the Sales.SalesOrderDetail column.** Notice that Aggregation is now enabled in the ribbon.
6. **Click the icon and the Aggregation dialog window opens.** A list of columns appears, with Power Query suggesting how to aggregate the data.
7. **Select the Sum of SalesOrderDetailID column, then hover over the column and click the newly available drop-down arrow.** Deselect Sum and select Count (All) from the list of available choices.
8. **Click OK.** Power Query adds a new column named Count of Sales .SalesOrderDetail.SalesOrderDetailID, or something similar.
9. **Review the results.** Notice that the values for each SalesOrderID match exactly the ones in Figure 4-20.

To ensure that all possible aggregations techniques are exposed the following steps are provided. Note that the interface and the steps are similar to using the Transform ribbon. The primary difference is the method by which the interface is accessed. Using the workbook from the previous section, do the following:

1. In the Applied Steps section of the Power Query editor, delete the last step.
2. Click the expand icon located in the Sales.SalesOrderDetail column header.
3. Select the Aggregate radio button at the top of the expanded list, and select Sum of Sales.SalesOrderID.
4. Hover over the column and click the newly available drop-down arrow. Deselect Sum, and select Count (All) from the list of available choices.

Again, a new column is added named Count of Sales.SalesOrderDetail.SalesOrderDetailID, or some variation, with the results similar to those displayed in Figure 4-20.

In actuality you could consider this another method, but because the interface is almost exactly the same, most consider this approach synonymous with using the Transform ribbon.

NOTE This behavior is available only when the source table is in the one-to-many relationship. For example, a one-to-many relationship exists between Sales.SalesOrderHeader and Sales.SalesOrderDetail. On the other hand, if you select the Sales.Customer column, notice that the Aggregation icon is not enabled. Even further, if you click the expand icon, notice that neither radio button is available. This is because a many-to-one relationship exists between the two tables, with Sales.SalesOrderHeader being the “many” side to the relationship. To further validate this, open Sales.Customer in the Query Editor. If you click the Sales.SalesOrderHeader column, the Aggregate icon is enabled. In addition, if you click the expand icon, the Aggregate radio button is available.

Editing Query Steps

During the process of importing and modeling data in Power Query, at some point most users notice that each change or modification to the data is tracked in the Applied Steps section of the Query Editor window, as shown in Figure 4-21.

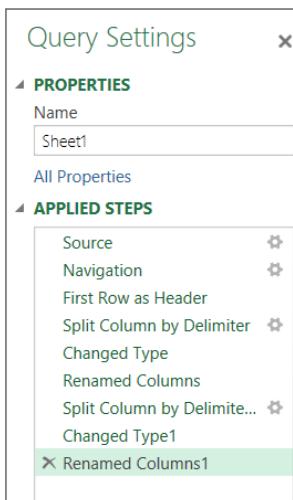


Figure 4-21: Applied Steps section displaying changes to data

Each item in the list represents some process that has been done since the data has been accessed. The icon located to the right of some of the steps indicates the starting point of some process that was performed using the Graphical User Interface (GUI). For example, clicking the icon located to the right of the first Split Column by Delimiter step opens the Split a column by delimiter dialog box.

A delete icon (X) appears whenever you hover your mouse pointer over a step name. Simply clicking this icon removes the step. There's no undo option, so be careful about removing a step.

Introducing M Programming

Although you can accomplish most tasks in Power Query through its graphical user interface, it also includes a very powerful programming language known as M.

A Glance at the M Language

This comprehensive language provides a different mechanism for importing and transforming data. To view a sample of the M programming language:

1. **Open a Power Query Editor window connected to a data source.**
2. **In the ribbon, click View and click the Advanced Editor icon in the Show section.** The Advanced Editor window opens, shown in Figure 4-22. The code seen in the text box editor typically corresponds to the steps taken while working with data in Power Query. For example, Figure 4-22 shows the M code generated during the process of importing the Sales.Customer table from the AdventureWorks2012 database. Renaming the column CustomerID to CustomerIdentification in the editor modifies the code to accommodate that change (see Figure 4-23).

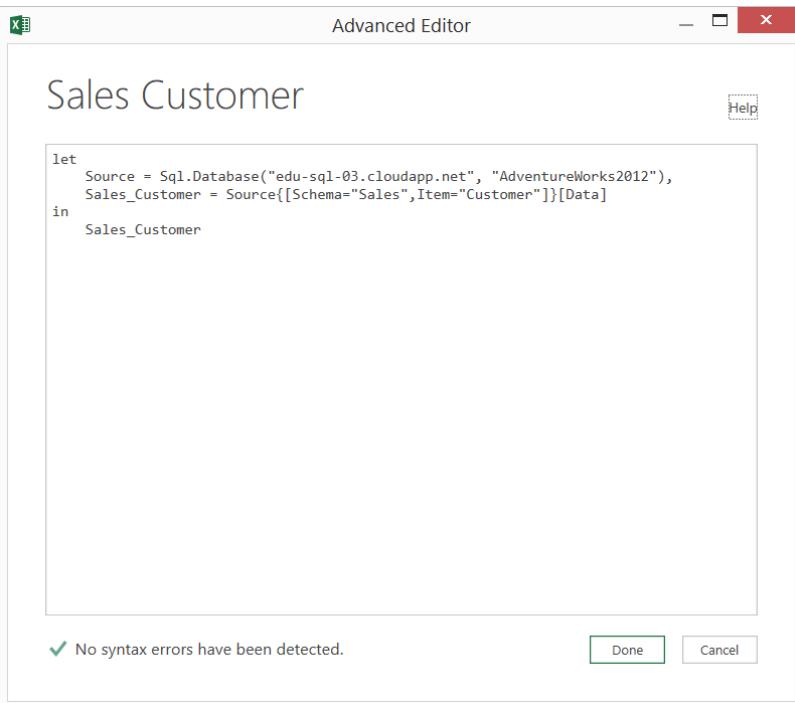


Figure 4-22: Power Query Advanced Editor window

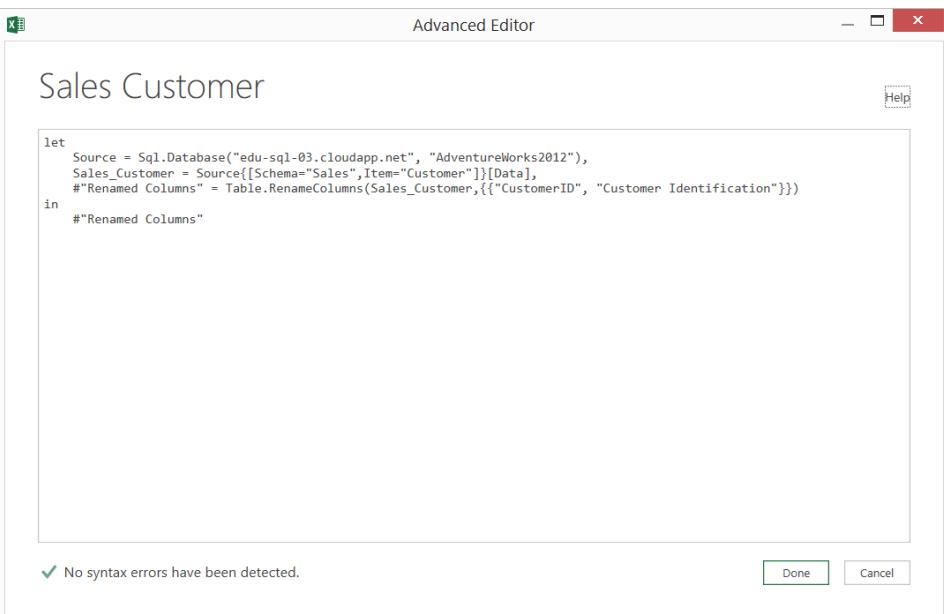


Figure 4-23: Advanced Editor window showing renamed column code

3. **Review the code located within the red rectangle.** The editor uses this code to modify the column name.

Another interesting thing about the M language: You can start a blank query in Power Query by selecting the last choice from the list of Other Sources. By leveraging this approach, you would need to be very knowledgeable of the M language. At first glance, you can easily tell that the typical Excel user, or even a power user, would not be likely to work with it. It requires a lot of time, persistence, and, honestly, devotion. Most of the techniques and concepts are well beyond the scope of this book, but there is one that does need to be discussed.

Adding and Removing Columns Using M

When using Power Query to model data, you commonly have to add one or more columns. Whereas you can get most things done in Power Query using the GUI, there is always a case that just does not fit within the scope of the tools. This is where M lends itself so well. For example, assume that you had to remove a “SO” from every value in the SalesOrderNumber column of the Sales.SalesOrderHeader table in the AdventureWorks2012 database. As of the writing of this book, you can’t accomplish this task through any existing interface. However, by leveraging M, you can do this simply by adding a column and inserting a line of code:

```
Text.Range([SalesOrderNumber], 2)
```

By leveraging the Text.Range function, you can manipulate the value so that it returns the number without “SO.” Text.Range has two required arguments and one optional. In the example, the first two are used. For a full list of these functions and their uses, visit <https://support.office.com/en-US/Article/Power-Query-formula-categories-125024ec-873c-47b9-bdfd-b437f8716819>.

Summary

This chapter provided an introduction for searching and combing data using Power Query. In some cases, it provided step-by-step demonstrations to showcase the capabilities of the tool.

In the next chapter a thorough contrast and comparison will be made between the different types of BI semantic models.

Choosing the Right Business Intelligence Semantic Model

For many releases, SQL Server has included SQL Server Analysis Services (SSAS), which provides Online Analytical Processing (OLAP) and data mining features and capabilities to business intelligence projects and solutions. In SQL Server 2012, Microsoft changed its modeling approach from Unified Dimensional Modeling (UDM) to Business Intelligence Semantic Models (BISM). The UDM included only multidimensional models, whereas BISM not only includes multidimensional but also tabular and Power Pivot models.

The multidimensional modeling approach consists of a very robust and mature set of capabilities that addresses the needs of all organization sizes. Although it is definitely a viable choice for business intelligence analytics, overcoming the learning curve is sometimes difficult for many new to the technology, and is often one of the primary deterrents to adoption. Learning to develop, deploy, and manage these models involves a very wide and intimate knowledge of the product. It typically takes someone months, if not a year, to become a competent multidimensional modeler—an investment often too timely and costly for individuals and organizations.

Just to clarify, none of this should discount the worth of SSAS multidimensional models as a business intelligence solution, but only provide context for upcoming comparisons between it and the recently released Power Pivot and tabular modeling approaches. There are various reasons why one should be implemented over the other, as well as many differences existing among the three, but in the end all provide functionality useful to any of the Microsoft

business intelligence visualization tools such as Excel, Power View, SQL Server Reporting Services (SSRS), and Power BI. This is what makes the business intelligence semantic model one of the most flexible and scalable solutions on the market today.

In short, BISM is all encompassing in regards to SSAS: It includes both multi-dimensional and tabular models. It also includes an in-memory engine, Power Pivot, which is an add-in for Excel. Figure 5-1 shows a visual representation of BISM, providing a clear illustration and definition of how it is used. The BISM provides an access layer, either dimensional or relational, to various types of visualization tools, also shown in the figure. This chapter further discusses BISM, comparing and contrasting the three different types of models, and finally offers some background as to when you should use one over another.

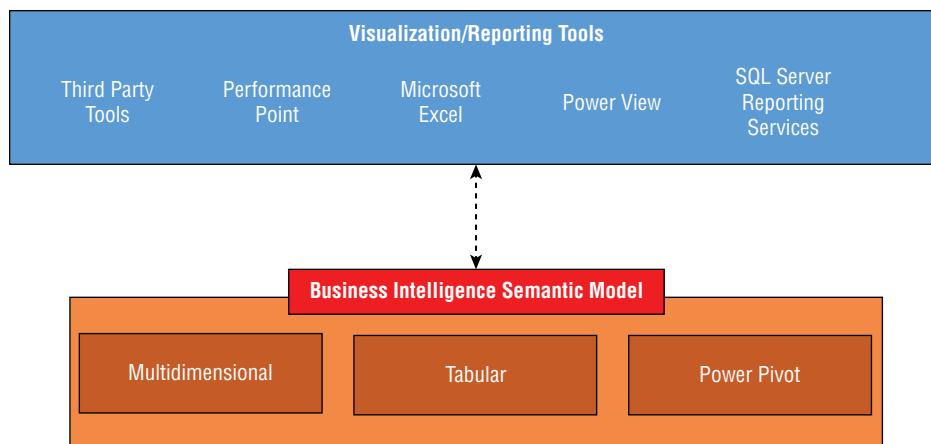


Figure 5-1: The business intelligence semantic model

Understanding the Business Intelligence Semantic Model Architecture

The business intelligence semantic model is conceptually composed of three layers:

- Data Access
- Query and Business Logic
- The Data Model

The consolidated result of each layer provides enhanced and accelerated access to data via a single model, which can then be consumed through various tools and visualizations as shown previously in Figure 5-1. The actual

authoring or design of the business intelligence semantic models takes place in either Microsoft Excel using the Power Pivot add-in, or SQL Server Data Tools (SSDT). Prior to SQL Server 2012, all business intelligence development was done in Business Intelligence Development Studio (BIDS). However, when 2012 was released, BIDS was replaced with SSDT. In addition to that change, a couple of new business intelligence project templates were added. Figure 5-2 shows the new templates in the boxed area:

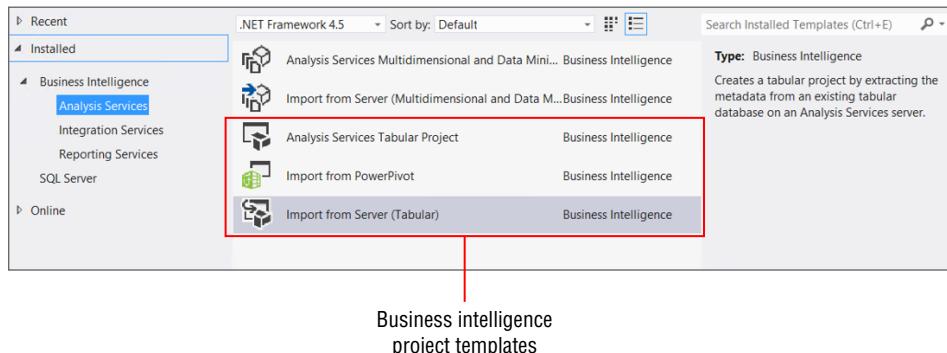


Figure 5-2: SQL Server Data Tools showing new SSAS templates

- **Analysis Services Tabular Project:** This creates a tabular model from scratch, which will be described in Chapter 7.
- **Import from Power Pivot:** This actually allows the developer to import and convert a model built in Power Pivot to a tabular model that will reside on a server.
- **Import from Server (Tabular):** This allows developers to extract the schema of a deployed tabular model into a project from the server.

The following sections of the chapter explain each layer of the business intelligence semantic model and their similarities and differences.

Understanding the Data Access Layer

Consider what data an organization would need to solve a particular problem. A manufacturing company might need to analyze sales over the past year, and a university might need to look at the past year's enrollment. Regardless of the problem, one of the first steps in either case is to identify the data sources, which may or may not reside in a centralized repository.

Typically, when performing data analytics, you obtain the source data from a data warehouse (central repository) hosted on a RDBMS like SQL Server.

However, with the changing times and end users constantly requesting access to data that either does not exist in the data warehouse or is beyond the realm of an organization's data center, analytical modeling must have the ability to combine data from heterogeneous data sources and types. As a result, the data access layer of the business intelligence semantic model can consume data from a myriad of sources. Table 5-1 provides an exhaustive list of the possible data sources that you can use across the three types of semantic models.

Table 5-1: Semantic Model Sources

POWER PIVOT	TABULAR	MULTIDIMENSIONAL
Microsoft SQL Server	Microsoft SQL Server	Microsoft SQL Server
Microsoft SQL Azure	Microsoft SQL Azure	Microsoft SQL Azure
Microsoft SQL Server PDW	Microsoft SQL Server PDW	Microsoft SQL Server PDW
Microsoft Access	Microsoft Access	Microsoft Access
Oracle	Oracle	Oracle
Teradata	Teradata	
Sybase	Sybase	
IBM DB2	IBM DB2	
Others (OLEDB\ODBC)	Others (OLEDB\ODBC)	Others (OLEDB)
Microsoft Analysis Services	Microsoft Analysis Services	
Report	Report	
Windows Azure Marketplace	Windows Azure Marketplace	
OData Feeds	OData feeds	
Excel File	Excel file	
Text File	Text file	

As the table shows, there are many data source options to consider when building a business intelligence semantic model, including traditional and some nontraditional sources such as web services, OData feeds, text files, and so on. Although Power Pivot and tabular models both eclipse multidimensional in the number of sources, all three provide varying types of data access that can accommodate organizations of all types and sizes. Figure 5-3 shows the multiple ways a business intelligence semantic model may be implemented.

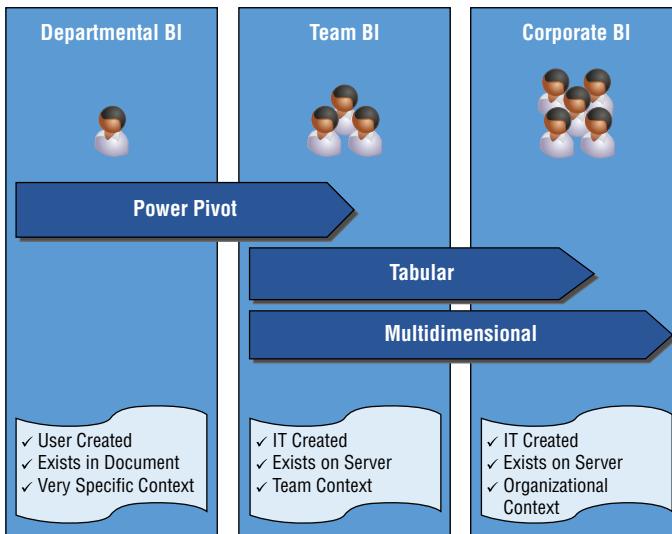


Figure 5-3: Business intelligence semantic model implementations

Both the tabular and multidimensional models include technologies that support two modes for accessing tables; Power Pivot supports only one.

Using Power Pivot

As shown, you should primarily use Power Pivot for more personalized or departmental business intelligence. However, you can sometimes use it for small teams, specifically when user-specific data security implementations are not required. The tabular and multidimensional models are meant for larger audiences with more complicated requirements and larger data needs. Additional information regarding each type will be explained later in this chapter.

In terms of the business intelligence semantic model data-access layer itself, no matter the type, the model provides different options for caching the data and allowing it to pass directly through. When Power Pivot was initially released, Microsoft introduced Vertipaq, which is an in-memory analytical engine that allows for the caching of data in-memory. Therefore, instead of loading only small amounts of data in Excel, by leveraging Power Pivot, Excel could support hundreds of millions of rows of data on desktops and laptops.

As mentioned earlier, you can cache the data or pass it directly through with the exception of Power Pivot. Remember, Power Pivot only leverages the xVelocity in-memory technology. When designing tabular or multidimensional models, you have the choice of processing the data into the model or querying the underlying data source directly (pass-through).

Using the Multidimensional Model

For multidimensional, the data-access modes would be Multidimensional Online Analytical Processing (MOLAP) or Real-time Online Analytical Processing (ROLAP). When you use the MOLAP storage mode, the model or cube must be processed. During processing, the data is imported from the underlying sources and stored in SSAS. ROLAP, on the other hand, does not require processing because it pulls the data through from the source to SSAS. It could be considered as a method for performing near real-time analytics. The multidimensional modes have been around for many releases of SSAS, and most users are already familiar with how and when to use the methods. There are a couple of variations to these modes—Hybrid Online Analytical Processing (HOLAP) for example—which is a combination of MOLAP and ROLAP. Chapter 8 provides more details about the different modes of multidimensional models.

Using the Tabular Model

The tabular model, which was introduced in SQL Server 2012, uses two modes similar to MOLAP and ROLAP:

- **In-memory mode (xVelocity):** This is similar to MOLAP, which caches the data from the sources into the model. The data is organized by column and uses advanced compression and access techniques to operate and manage the data all while in-memory. Similar to MOLAP, in-memory models must be processed to import the data.
- **Direct Query mode:** This is similar to ROLAP. Instead of caching data in-memory, the queries are passed through to the underlying source.

NOTE With the release of SQL Server 2012, Microsoft integrated the Vertipaq in-memory technology into Analysis Services and also changed the name to xVelocity. So now Excel can realize not only the performance improvements of this technology, but also SSAS via the tabular model.

Just like multidimensional models, tabular models can involve hybrid approaches; four Query mode choices are available for configuring tabular models and these are discussed in Chapter 7. There is one small caveat to using the Direct Query mode of tabular models: It works only when SQL Server is

the underlying source. Figure 5-4 provides a view of each business intelligence semantic model type and its corresponding data access method.

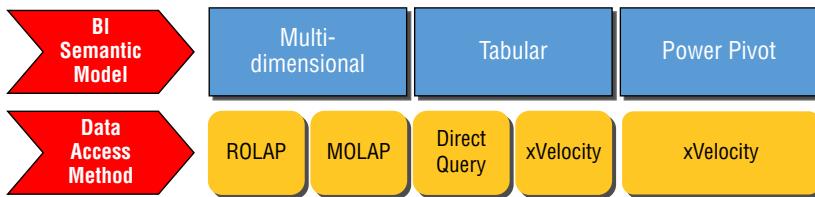


Figure 5-4: BISM models and corresponding data access methods

Implementing Query Languages and the Business Logic Layer

Now that you know what you can access and how to access it, it is time to discuss the next layer in the model: query languages, and how to implement the logic. Similar to the data-access methods, each model type has a corresponding query language. Multidimensional models use Multidimensional Expressions (MDX); tabular models and Power Pivot use Data analytic expressions (DAX). You use each language to create calculations, measures, row filters, Key Performance Indicators (KPIs), and other types of business logic.

Data Analytics Expressions (DAX)

DAX is a new expression language based on Excel formulas that was introduced with Power Pivot, but has since been integrated into SSAS tabular model projects. More specifically, you use it to build Calculated Columns and Measures in tabular and Power Pivot models, as well as Row Filters in tabular models only.

DAX is built on relational database concepts—essentially tables and relationships. Think of a simple Entity Relationship Diagram (ERD) as shown in Figure 5-5.

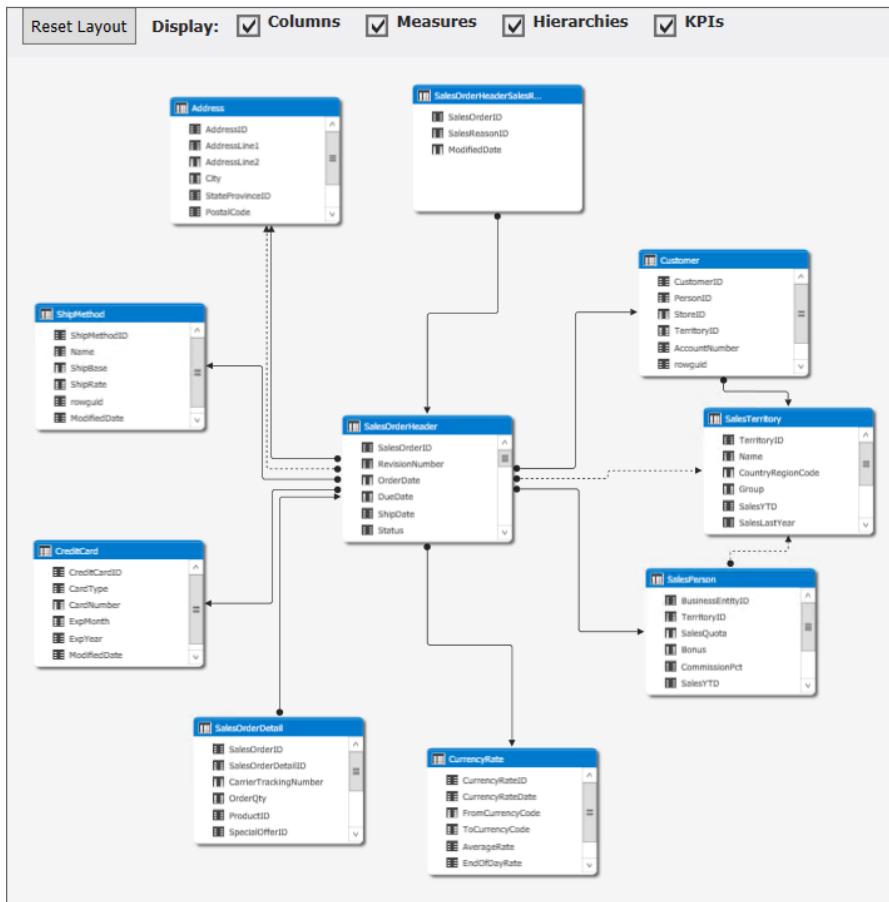


Figure 5-5: Simple Entity Relationship Diagram

Most people learned Database 101 a long time ago. For some, understanding relational databases design terminology and concepts are like getting out of bed each morning. For example, the concept of One-To-Many has the same definition when designing a tabular as compared to an ERD. Even further, remember the diagram from Figure 5-5, it was taken from a tabular model built using SSDT. Because the tabular models design is based on relational database concepts, the process of implementing complex business logic into the data models has been extremely simplified. More information and examples of using DAX is provided in Chapter 6.

The learning curve involved with the adoption of DAX has been reduced tremendously. Instead of building a model using a completely unfamiliar process, users with relational database and Excel knowledge can build a model with a minimum amount of training. This is, of course, compared to more

traditional and mature cube technologies, including multidimensional models and their corresponding programming language, MDX, which is discussed in the next section.

Multidimensional Expressions (MDX)

MDX is a very mature and robust programming language that enables building, manipulating, accessing, and querying multidimensional objects within a cube. Although its syntax includes similar keywords to Transact-SQL (TSQL), such as SELECT and FROM, it is not an extension of the TSQL programming language. TSQL is a set-based language that references two-dimensional data, typically at the intersection of columns and rows of tables within a traditional relational database. On the other hand, you use MDX, primarily to query a cube, as mentioned earlier. This cube is constructed mostly of measures and dimensions—one, two, or many dimensions, depending on how the underlying data source is built.

Consider Figure 5-5, which shows a sample ERD from a data warehouse. Querying this data with TSQL would result in a series of JOINs on columns from each table returning a subset of the data. However, when the data is loaded into a cube, the process becomes a little more complex. Although this book does not provide an introduction to MDX, understanding how to properly compose MDX queries is a time-consuming process, requiring intimate knowledge of tuples, cells, and possibly, sets. For example, instead of accessing a set of columns and rows, querying a cube results in a point on the cube that may reference N number of dimensions. These statements are not meant to devalue multidimensional models, only to provide an understanding of the knowledge needed to properly access the data, and the differences as compared to DAX.

Direct Query and ROLAP

Direct Query and ROLAP do not process data into the models, and neither is an actual expression or query language. However, they both require the use of MDX and DAX. As stated earlier, the queries are converted to TSQL and passed through to the underlying data source. Given that, the developers and administrators should ensure that the source can afford this type of activity. Often, particularly with SQL Server, leveraging features such as Compression, Partitioning, and/or Columnstore indexes lends itself well to the improved performance of this type of topology. In practice, this approach is actually easier than processing the data into the cube because of the reduction of administrative overhead (processing the data, making sure that it is processed, and so on). In addition, end users do not have to wait for the data to process; the model is as fresh as the underlying data source.

Just to sum everything up, Figure 5-6 builds upon Figure 5-4, adding the query and business logic layer for each model type.

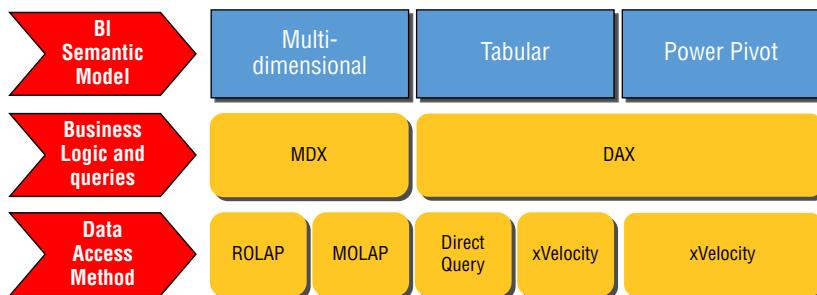


Figure 5-6: BISM models and corresponding query languages

ROLAP and Direct Query are similar in the sense of pass-through queries, but completely different technically. Direct Query has its engine for executing queries, whereas ROLAP uses the same one as MOLAP for queries. You can only use Direct Query when SQL Server is the data source; on the other hand, you can use ROLAP with other data sources such as Oracle. Finally, Direct Query is not as memory intensive because it does not cache data into memory like ROLAP.

Data Model Layer

This layer is what the developer of the model uses to design and build what visualization tools and end users consume. By using SSDT, the developer can implement the Data Access layer of choice, and also use DAX or MDX to implement complex business logic into the model. For example, when designing a business intelligence semantic model, a scenario may arise requiring year-over-year growth. By using either of the query languages and functions that each exposes, end users can create and consume this measure as part of the model.

In addition to using the query languages to add business logic in SSDT, developers can also create hierarchies, perspectives, default field sets, unique identifiers, measure groups, categorizing data, unary operators, custom rollups, partition strategies, and so on. For security purposes, both tabular and multidimensional models also allow developers to implement row-level security. You can do this using a static or dynamic approach.

Comparing the Different Types of Models

So far, this chapter has compared the three types of business intelligence semantic models. Some differences regarding the Data Layer modes of each were explicitly stated, but a direct comparison was not provided. These details are shown in Table 5-2.

Table 5-2: Business intelligence Semantic Model Comparisons

MULTIDIMENSIONAL	TABULAR	POWER PIVOT
Actions	via BIDS Helper	No
Aggregation objects	No	No
Calculated measures	Yes	Yes
Custom rollups	No	No
Distinct count	via DAX	via DAX
Drillthrough	Yes	Opens in separate worksheet
Hierarchies	Yes	Yes
KPIs	Yes	Yes
Linked objects	No	Linked tables
Many-to-many relationships	via Design Patterns	No
Parent-child hierarchies	via DAX	via DAX
Partitions	Yes	No
Perspectives	Yes	Yes
Row-level security	Yes	No
Semi-additive measures	Yes	Yes
Translations	via BIDS Helper	No
User-defined hierarchies	Yes	Yes
Writeback	No	No

The first thing you should notice is that the multidimensional model is the baseline for the comparison. In other words, it can perform most of the needed analytical requirements when compared to the other two model types. The next thing is that by using DAX or DAX patterns, it is possible to simulate some missing features. A good reference point to find these patterns and others is <http://www.daxpatterns.com>. Finally, two particular items, Actions and Translations, are not natively supported by either tabular or Power Pivot models. However,

you can create both in tabular by using BIDS Helper, which is a free plug-in to SSDT. You can download from <http://bidshelper.codeplex.com/>.

All in all each model has a robust set of capabilities. Some are native and others not so native, but each has its place within an organization, no matter the size or task that you are trying to accomplish.

Which Model Fits Your Organization?

When and how to use a specific model depends on several factors. Often it depends heavily on the amount and type of data, the complexity of the business logic, and the level of security that you must implement. Most organizations, whether large or small, private or public, profit or non-profit, are organized into three categories:

- Department
- Teams that span departments
- The entire organization

Although this is not an exhaustive list of everything that you should consider, think of them as the top three items. You might find some gray areas between and across each, but in general this should cover the basics. Each sector may have some data specific to their department or team; each may also have data that applies to the entire organization.

Departmental

Departmental business intelligence usually is focused on a very small context-specific set of data, which is often the product of a user looking for a way to access data not included in the organizational business intelligence solution. For example, the payroll department may want to analyze some salaries across the organization, or an institutional research department at a university may want to analyze some data collected during a recent project. These types of datasets are typically not loaded into the organizational data warehouse. As a result, that department will invest resources, people, money, and time to build a business intelligence solution that is self-contained and maintained within that department.

The data itself is mostly available to everyone within the department and therefore does not require security beyond document level. In other words, everyone in the department should be able to consume and view the data. Now think about those individuals in these departments who are well versed in Excel, and who know more about Excel than employees in the IT staff. It would be

easier for them to learn how to use Power Pivot as an analytical tool instead of introducing something completely unfamiliar. Using Power Pivot, end users can consume data from a mix of heterogeneous sources, and transform the data into the perfect model that meets the requirements of the department without any help from the IT staff.

However, once data is loaded into Power Pivot, anyone that has access to the workbook has access to all the data imported into the model. This is one of the most-known and discussed limitations of Power Pivot, and is one of the main reasons that Power Pivot is used primarily for personal or department business intelligence; the security requirements for those models are not so stringent. For example, in most organizations, payroll data is available only to those individuals that work in the payroll department. Therefore, building an analytical model using Power Pivot and sharing that workbook across the department is a viable solution because everyone in the department works within the context of this data on a regular basis.

Another thing to consider when working with Power Pivot is the size of the data. You can deploy Power Pivot using either 32- or 64-bit hardware. 32-bit environments have hardware limitations that directly lead to data limits inside of Power Pivot. 64-bit hardware, on the other hand, does not have these requirements. More details regarding data limitations are provided in Chapter 6; for now, as a best practice, lean more toward developing Power Pivot models for 64-bit environments. To check which version of Excel is currently installed on a machine, click File on the ribbon, then click Account from the list of choices available on the left side of the screen. On the account page, locate and click the button labeled About Excel. Figure 5-7 shows part of the About Microsoft Excel dialog window that opens.

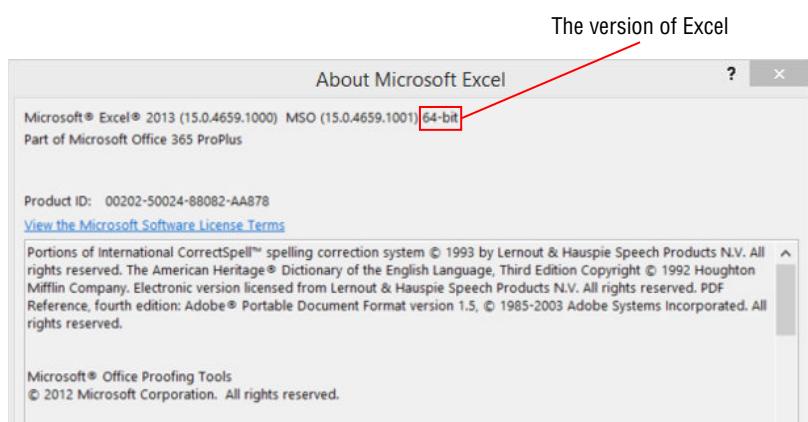


Figure 5-7: About Microsoft Excel window showing the version

The version of Excel appears toward the top of the window.

After the data is modeled, the department can use Power View, which is discussed in Chapter 12, to create interactive reports that allow end users to explore and visualize the data directly within Excel.

Team

Team business intelligence produces a lot of gray area regarding which business intelligence semantic model to use. When data sources begin to cross departmental lines, two dynamics usually help drive or determine which model will fit: the increase in both security needs and data sizes. However, this may not always be the case, as you may need more security with less data (if a 64-bit environment is available).

If, for example, you must secure data very granularly, then Power Pivot would not be a good option. Assume you have two departments working on a project—finance and sales. Those individuals analyzing the data in the finance department may need to see all the data across geographical regions, whereas individuals in sales may need to see data only in a specific region. If you use Power Pivot, the developers would need to develop and deploy workbooks for finance and each sales region within the organization. The outcome of this deployment is multiple Power Pivot models that must be updated and maintained as it grows and changes.

Instead, leveraging tabular models and their ability to intuitively support dynamic row-level security reduces administrative time and produces a more centralized and consistent data model. With this approach, instead of hosting the data and the model inside Excel, the model is designed using SSDT, and the data is hosted on a server. Now, when end users access the data, the model can dynamically display data within the context for that user. Returning back to the finance and sales example, if the manager of the east region accessed the model, it would filter the data showing only data from that region. However, if the finance department accessed the model, all the data could be seen.

Of course, security is not the only consideration. Although Power Pivot can host large amounts of data, the number of concurrent requests and connections are limited. In addition, when deploying the models to SharePoint, there must be some communication with the administrator to ensure that upload sizes are set high enough to accommodate the model needs. By default, the SharePoint maximums are much lower than the data maximums allowed in Power Pivot. If you use a Tabular model, you must place more focus on the server that hosts the model rather than on Excel and SharePoint. This is because all the data is loaded into the server memory if you select the xVelocity mode.

Organizational

The most difficult choice to make regarding which model to use lays at the organizational level, because both tabular and multidimensional can support almost all levels of models from a technical standpoint. However, as of the writing of this book, two major limitations to tabular models can quickly force organizations to use multidimensional models:

- Custom Rollups
- Writeback

If the business intelligence deployment requires either of the above then the project is limited strictly to multidimensional models.

If the project requirements do not involve these features, then the choice becomes unclear. Although some debate exists about data size affecting the overall performance of tabular models, if you developed the model with scalability and flexibility in mind, you can overcome most of these challenges. The important thing to point out is that tabular models are more memory dependent than multidimensional. Ensuring that the hardware can accommodate data growth is imperative to the performance of any model. What this means is that even though tabular models can page-out the disk if they exceed the memory limitations of the hardware, they are not as forgiving under this type of scenario. By leveraging aggregations in multidimensional models, you could possibly improve the performance but this is not an option for tabular models. The decision requires a very meticulous approach in estimating resource requirements, so the hardware you purchase not only accommodates current data needs, but also scales as the data needs increase.

A final item to consider when contemplating a server-based model applies to both team and organization business intelligence: If this is a new project, start with a tabular model. Why? Because of its simplicity. Developing a flexible and scalable multidimensional model requires a high-priced consultant, likely with years of experience. This person may or may not be on staff, or well beyond the budget constraints of the project. In that case, the tabular model is a good starting point. This is not to say you should not spend time learning and becoming versed in tabular model design. In either case, prior to starting any project, carefully gather all the requirements of the project to ensure that you choose a model that can meet all your organization's needs. Remember, tabular models are relatively new and do not support all the features that are available in multidimensional models. Over time, the tabular model will mature and likely become technically equivalent to multidimensional models. In the meantime, evaluate both carefully before making a final decision.

Summary

In this chapter, information was provided that will assist developers in selecting the correct semantic model. Similarities and distinctions between the three models, Power Pivot, Tabular, and Multidimensional, were listed and explained. Finally, this chapter provided a brief overview of where each type fits appropriately—whether that is departments, teams, or organizations.

The next chapter explains how you can change Power Pivot to create semantic models so that end-users can effectively access the data.

Discovering and Analyzing Data with Power Pivot

In this chapter another Excel add-in, Power Pivot, is introduced. Power Pivot is an in-memory analytical engine that transforms excel into a true business intelligence modeling tool. By leveraging this add-in, users can import hundreds of millions of rows into Excel because Power Pivot elegantly compresses the data down to a fraction of its original size. In addition, Power Pivot allows users to add key performance indicators (KPIs), hierarchies, calculated columns, measures, and other analytical items, which are all discussed throughout this chapter.

As stated in Chapter 5, Power Pivot models are perfect for individual or departmental business intelligence. Because most data analysts or people that work with data are typically familiar with Microsoft Excel, using Power Pivot is a good starting point. With a few exceptions, most of the knowledge that has been acquired through years of Excel use can easily be transformed and transferred directly to Power Pivot skills. The interface is similar to workbooks and tabs used in Excel today. The expression language is the same, with major additions that provide enhancements specific to Business Intelligence. Given the previous statements, and the fact that Excel is one of the most-used data analytical tools, makes Power Pivot an optimal choice.

With that said, remember that the data is contained within an Excel workbook or document, meaning the data is only as secure as the document itself. For proper sharing and collaboration, instead of sharing the workbook via email, which is the typical choice, authors of the model should consider saving the files to a Power Pivot Gallery or document library hosted in SharePoint or

SharePoint Online. This method not only provides a secure centralized way for sharing and consumption of the model, but also additional capabilities such as scheduled data refresh and the ability to leverage the model as a data source for visualization tools like SSRS and Power View. For more about deployment strategies, see Chapter 17.

Understanding Hardware and Software Requirements

This won't be the dry list of hardware and software requirements commonly included with program documentation. Instead, a more holistic approach is provided, giving an overview of certain aspects of each requirement. It is imperative that the proper version of Excel be installed and that the machine hosting the install meets more than the minimum hardware requirements.

There is not much to be said about software. Using Power Pivot in Excel 2013 requires either Office 365 Professional Plus or Microsoft Office Professional Plus. Power Pivot is also available for those machines running Excel 2010. There have been some significant improvements in Excel 2013, so you should attempt to use this version if possible. Finally, you can use Excel 2007 to view and interact with Pivot charts and tables, but the Power Pivot model itself cannot be modified.

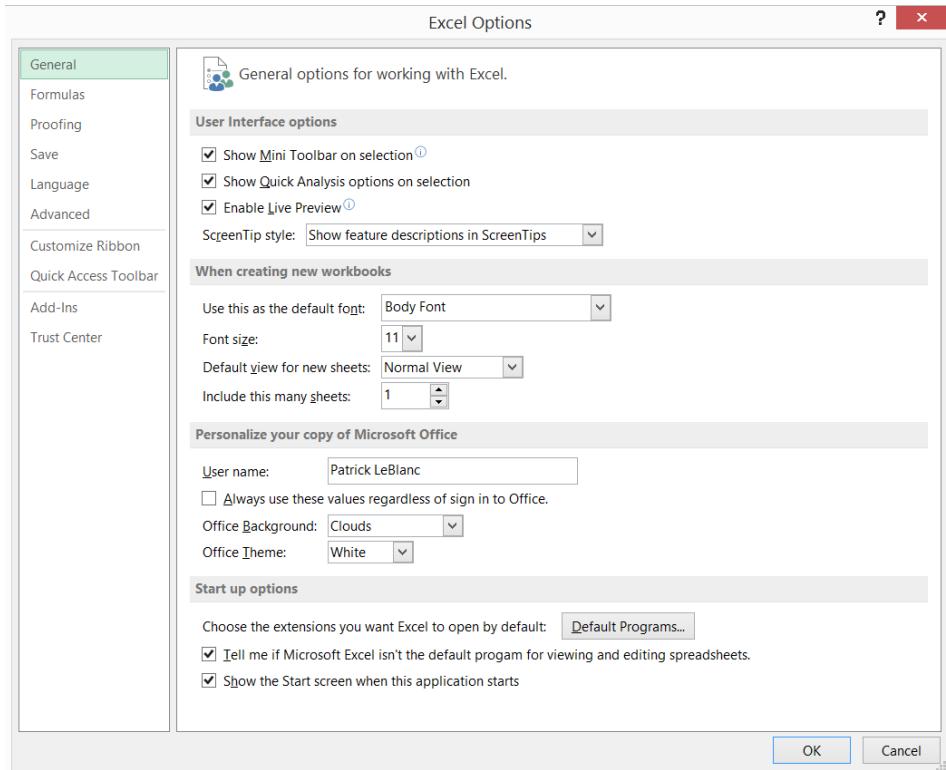
Hardware, on the other hand, is a different topic. A common trend among IT departments is to purchase and distribute 64-bit computers, which is ideal for building Power Pivot models. The unfortunate thing is that even though the machines are 64 bit, 32-bit software is installed. In many cases, specifically Microsoft Office, this is because several existing Excel add-ins are not compatible with 64-bit Excel, thus limiting the amount of data that you can load into the Power Pivot data model. When working with 32-bit Excel, you have a 2-gigabyte (GB) limit. Don't get excited; these 2GB are not dedicated to the data model. They are shared by all loaded data models and add-ins, and also open workbooks and Excel itself. Typically, the data models are limited to approximately 500–700 megabytes (MB). If the models are authored in a 64-bit environment, the 2GB limit does not exist. The data models are limited only by the available machine resources.

Enabling Power Pivot

By default, Power Pivot is disabled. If it does not appear in the ribbon simply follow these steps to enable it:

1. Open Excel.
2. Create a blank workbook.

3. Verify that Power Pivot is not available in the ribbon.
4. In the ribbon, click File ➔ Options from the list of available choices and the Excel Options window will open (see Figure 6-1).



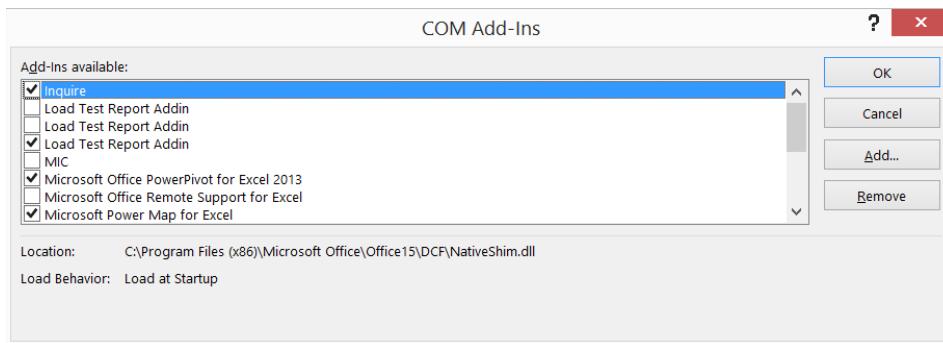


Figure 6-2: The COM Add-ins window

Once these steps are complete, Power Pivot will appear in the ribbon.

Designing an Optimal Power Pivot Model

Designing an optimal Power Pivot model should be centered on usability and performance. While the in-memory models created by Power Pivot employ a very powerful compression algorithm that often compresses the data between 7 and 10 times, it is imperative that the author understands how varying data and data types affect the compression ratio and performance. In addition, the developer must consider the consumers of the model and ensure that their interaction is fluid and intuitive.

All models start with a dataset or several datasets. Regardless of the type of data you're working with, carefully selecting the data and constructing the model is critical to how it performs. You don't often think of this until someone complains about poor performance. The next sections give a few tips explaining how to optimize the model so that it performs acceptably to end users.

Importing Only What You Need

Using Excel 2013, you can author and perform data analysis against Power Pivot models that contain millions or even hundreds of millions of rows of data and a corresponding large number of columns. Although you can possibly build these very large models, there are several reasons you should carefully consider before just loading any or all data into a model:

- Over- or misuse of memory
- Excel file size limitations when shared via SharePoint or Office 365
- Overcomplicated models and long field lists

Is it possible to design a model that contains all the data without regard to data size? Sure. However, as consumption increases and data grows, the host hardware will likely reach a tipping point, or the end users will become frustrated with the overly complicated structure of the model. As a result, taking a more proactive approach in data selection and model design will help create a performant and intuitive model for end users.

The best approach to ensuring proper use of memory is by omitting unnecessary columns. When importing data into the Power Pivot model, think carefully about the problem or the questions that may arise. If a particular column of data does not add any value to either, consider omitting it. On the other hand, some columns may provide value that does not relate directly to answering questions such as in the key columns. Power Pivot provides a very user-friendly interface when accessing and importing data that allows you to easily omit rows or columns. Figure 6-3 shows the Table Import Wizard displaying a list of tables from the AdventureWorksDW2012.

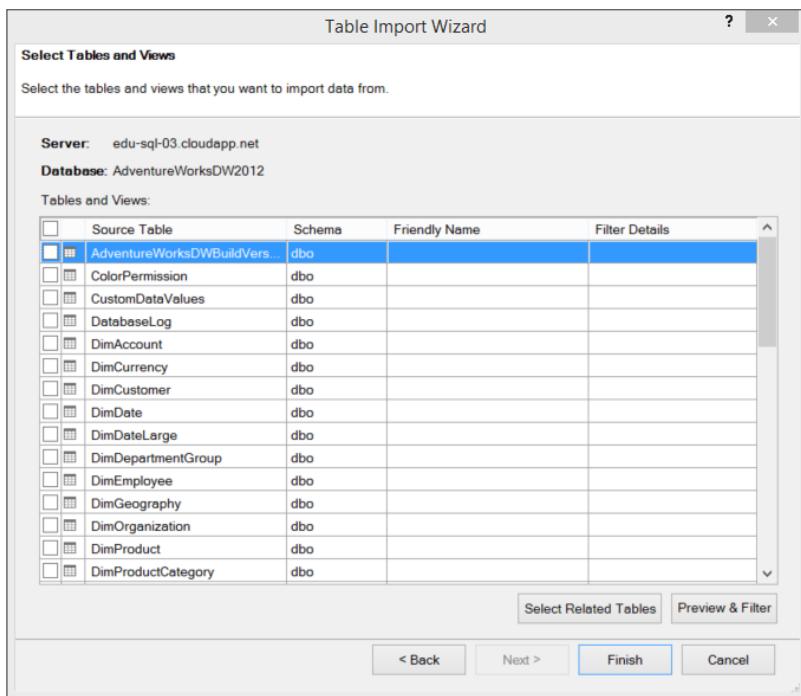


Figure 6-3: Power Pivot Table Import Wizard

Using this wizard, you can select a list of tables/views from the database to import by checking the box to the left of each list item. After selecting the tables, you must then decide if every row and every column from a table should be imported. If not, click the Preview & Filter button, located in the bottom right

corner of the wizard. This opens the Preview Selected Table window of the Table Import Wizard, shown in Figure 6-4, which you can use to limit the number of import rows or columns.

The screenshot shows the 'Table Import Wizard' dialog box titled 'Preview Selected Table'. It displays a preview of the 'FactInternetSales' table with several columns: ProductID, OrderDate, DueDate, ShipDate, CustomerID, PromotionID, CurrencyID, and SalesTerritoryID. Each column has a dropdown arrow indicating filter options. The table contains approximately 20 rows of sample data. At the bottom of the dialog are 'OK' and 'Cancel' buttons, with 'OK' being the active button.

Figure 6-4: Table Import Wizard Preview Selected Table window

To remove a column from the import, check the box next to the column name in the header. Rows can be filtered from the table by clicking the drop-down arrow located to the right of each column header. After the list of data appears, simply uncheck any of the values that need to be omitted. When you're done, click OK.

A data warehouse, which is a common data source for Power Pivot, typically contains four basic types of data:

- Key columns (surrogate or alternate)
- Additive columns
- Meta-data columns
- Dimension attribute or descriptive columns

The first three are mostly necessary. However, the fourth should be omitted in most cases. These are usually columns such as load date or ETL run date. Not only do they not add value, but their inclusion could misuse memory because of their data type, which will be discussed in the next section.

Consider the data shown in Figure 6-5.

	ProductK...	OrderDateK...	CustomerK...	SalesAmo...	LoadDate
1	310	20050701	21768	3578.27	2005-07-01 00:21:00.000
2	346	20050701	28389	3399.99	2005-07-01 06:28:00.000
3	346	20050701	25863	3399.99	2005-07-01 06:25:00.000
4	336	20050701	14501	699.0982	2005-07-01 06:14:00.000
5	346	20050701	11003	3399.99	2005-07-01 06:11:00.000
6	311	20050702	27645	3578.27	2005-07-02 01:27:00.000
7	310	20050702	16624	3578.27	2005-07-02 00:16:00.000
8	351	20050702	11005	3374.99	2005-07-02 01:11:00.000
9	344	20050702	11011	3399.99	2005-07-02 04:11:00.000
10	312	20050703	27621	3578.27	2005-07-03 02:27:00.000
11	312	20050703	27616	3578.27	2005-07-03 02:27:00.000
12	330	20050703	20042	699.0982	2005-07-03 00:20:00.000

Figure 6-5: Abbreviated FactInternetSales dataset

This is an abbreviated dataset obtained from the AdventureWorks sample data warehouse FactInternetSales table. Of the four displayed columns, you can conclude that only the SalesAmount column is needed. Additive values such as sales amounts are data points, and are typically included when doing data analysis. Key values (ProductKey, OrderDateKey and CustomerKey), such as those seen in Figure 6-5, may be inadvertently omitted because at first glance they appear to be data only needed for data warehouse manageability and maintainability. However, they are actually critical values when creating relationships, and directly assist in providing the functionality needed to perform aggregations and filtering across facts and dimensions. Omitting these key values limits the capabilities of the Power Pivot model. The final column, LoadDate, is strictly used by the data warehouse load process, and is a data value usually used by the ETL developer to perform more specific things relating to the data warehouse load process. These columns do not add any value to the model and therefore should not be loaded. Figure 6-6 shows the diagram of the Power Pivot model including relationships between the key columns.

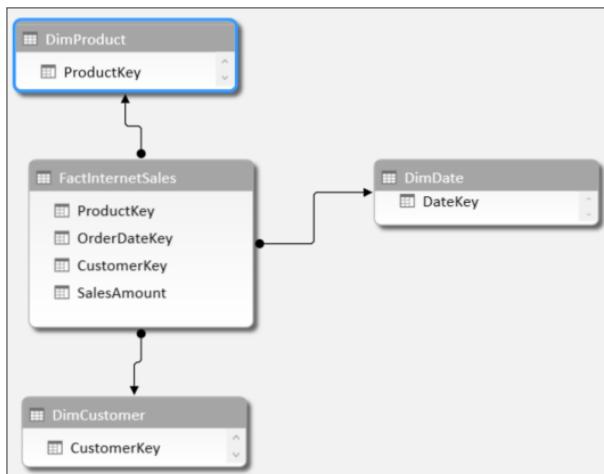


Figure 6-6: Diagram view of Power Pivot model

Each of the arrows represent a single relationship from the fact table to each dimension. Aggregations on attributes from the dimension attributes could not be performed without creating these relationships, thus the need for the key values.

Finally, the dimension attributes or descriptive data values are columns used to provide context around the additive values. They are typically included in the dimensions, but not always. When designing reports, they are used to filter the data and/or used to aggregate the data, and are absolute necessities in the model. Without them, the model would display only meaningless aggregated data.

With that understanding, it is time to build a model that will be used throughout the rest of this chapter. Follow these steps:

1. **Open Microsoft Excel.**
2. **Make sure that Power Pivot is enabled.** If Power Pivot does not appear in the ribbon follow the steps outlined in the “Enabling Power Pivot” section earlier in this chapter.
3. **Click Power Pivot in the ribbon.**
4. **Click the Manage icon located on the extreme left of the ribbon.** This opens the Power Pivot window.
5. **Click the drop-down arrow below the From Database icon.** Select From SQL Server from the list of available choices (see Figure 6-7).

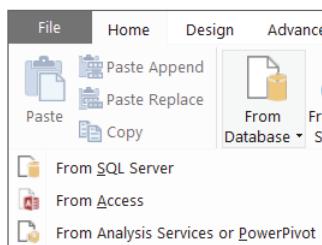


Figure 6-7: Select SQL Server

6. **Enter the name of the SQL Server that is hosting the AdventureWorksDW2012 databases.**
7. **Select the method of authentication that is used to log in.** This is either Windows or SQL Authentication (see Figure 6-8)

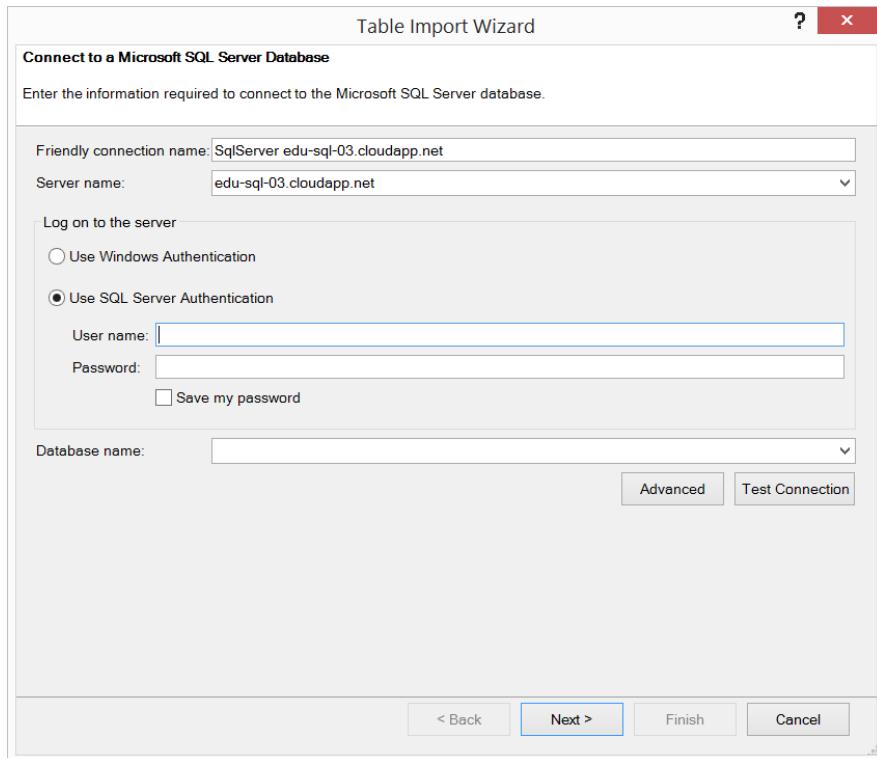


Figure 6-8: Select an authentication method

8. **Select AdventureWorksDW2012 from the Database name drop-down list.** Click Next.
9. **Chose how to import the data.** For this example, select the “Select from a list of tables and views to choose the data import” option (see Figure 6-9). Click Next.

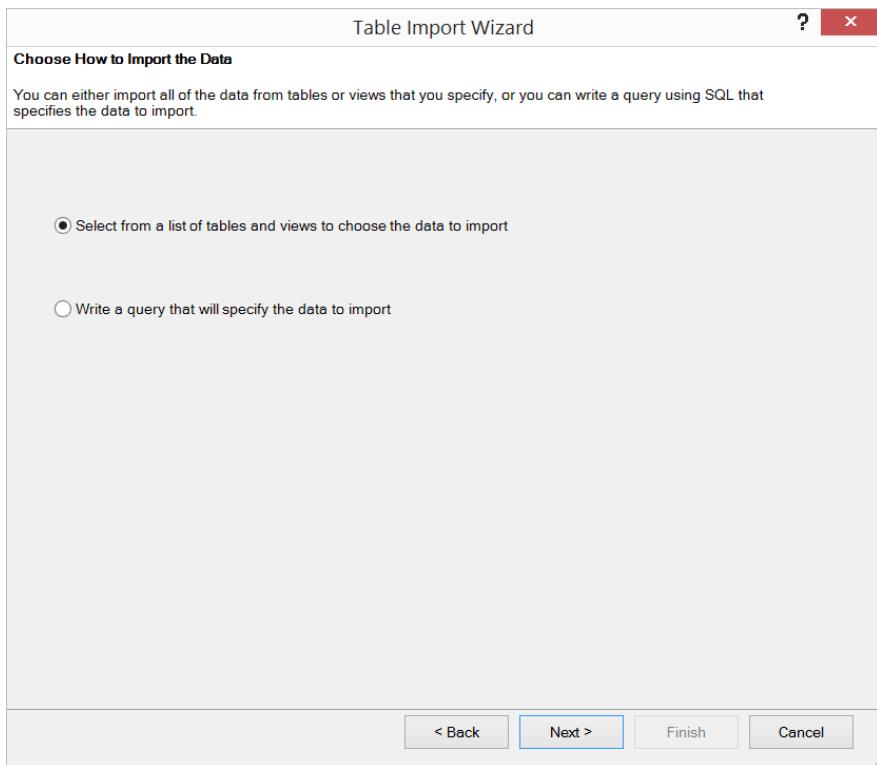


Figure 6-9: Select how to import data

10. **Select what to import.** Select the DimProduct, DimProductCategory, DimProductSubCategory, DimDate, DimSalesTerritory, and FactInternetSales checkboxes on the Select Tables and Views screen of the Table Import Wizard. Refer back to Figure 6-3 for a view of this screen.
11. **With FactInternetSales highlighted, click the button labeled Preview and Filter.** This will be on the Preview Selected Table screen of the Table Import Wizard. Refer back to Figure 6-4 for a view of this screen.
12. **Deselect the first check box in the header row of the preview.** This unselects all the columns.
13. **Select the ProductKey, OrderDateKey, CustomerKey, SalesTerritoryKey, ShipDateKey, DueDateKey, and SalesAmount columns checkboxes.**
14. **Repeat steps 11 to 13.** Do this for DimProduct, selecting the ProductKey, ProductSubcategoryKey, EnglishProductName, and LargePhoto columns.
15. **Repeat steps 11 to 13.** This time do this for DimProductCategory, selecting the ProductCategoryKey and EnglishProductCategoryName columns.

16. **Repeat steps 11 to 13.** This time do this for DimProductsubCategory, selecting ProductSubcategoryKey, EnglishProductSubcategoryName, and ProductCategoryKey columns.
17. **Repeat steps 11 to 13.** Finally, do this for DimDate, selecting DateKey, FullDateAlternateKey, DayNumberOfWeek, EnglishDayNameOfWeek, EnglishMonthName, MonthNumberOfYear, CalendarQuarter, and CalendarYear.
18. **Finish up.** Click OK. Click Finish. Close the Power Pivot window and save the Excel workbook with a name of **ABI Power Pivot Ch6**.

Remember where the workbook is stored because you will use it later in this chapter.

Understanding Why Data Types Matter

As mentioned at the beginning of this chapter, the Power Pivot storage engine compresses the data between 7 and 10 times the original size. For example, an original data size of 700MB will be only about 100MB after the model is loaded into Power Pivot. The actual amount of compression depends on the cardinality or uniqueness of the data. It is often better to have several columns with a large amount of repeating values rather than a single column with mostly distinct data. The columns with a high degree of repeating data provide a better compression ratio as compared to those with a large number of distinct values. Because of this, it is imperative to have a thorough and detailed knowledge of the data. Importing unnecessary data can adversely affect the model, resulting in poor performance.

With that context, think about a traditional data warehouse. Fact tables traditionally contain key columns and additive values, whereas dimension tables contain key columns and descriptive attributes. Most of the data types in the fact tables are numeric. They could be whole numbers, decimals, dollar amounts, or some other variation of a number. Note that for those numbers that include decimals, if the decimal place is not important, you modify the query to not include the decimal. The more decimal places a number has, the smaller amount of compression that can be applied. This is because the numbers become more and more unique as the number of decimal places increase. For example, assume that Sales Amount is a value that must be imported from the fact table. In its current state, the value is stored with two decimal places, similar to 147.35. The likelihood of that exact number appearing over and over in the fact table is remote. On the other hand, if the number was only 147, that likelihood increases, thus increasing the amount of compression that may occur. As a result, more data can fit in memory.

Choosing a Data Access Type

Also note that during the import process, you can access data from a relational data source in one of two ways: from a table/view or using a query. Using the table or view option has the advantage of enabling you to select multiple items at once, whereas the query option allows only one query to be imported at a time. Remember back to the decimal problem, your initial response would be to write a query using a CAST or CONVERT to remove the decimal. That is partially correct. However, instead of writing a query, follow these simple steps.

- 1. Open the ABI Power Pivot Ch6 Excel file.** For the sake of demonstration, assume that data imported from the FactInternetSales table in the AdventureWorks2012DW database does not require decimal places. The SalesAmount is stored in the data warehouse so that it includes decimal places.
- 2. Click the FactInternetSales table.** Notice that the values shown in the SalesAmount column currently contain values in the two decimal places.
- 3. Select Design in the Power Pivot ribbon and click the icon labeled Table Properties.** The Edit Table Properties window opens.
- 4. On the right-hand side of the window, click the drop-down list labeled Switch to.** Then select Query Editor from the list of available choices, shown in Figure 6-10.

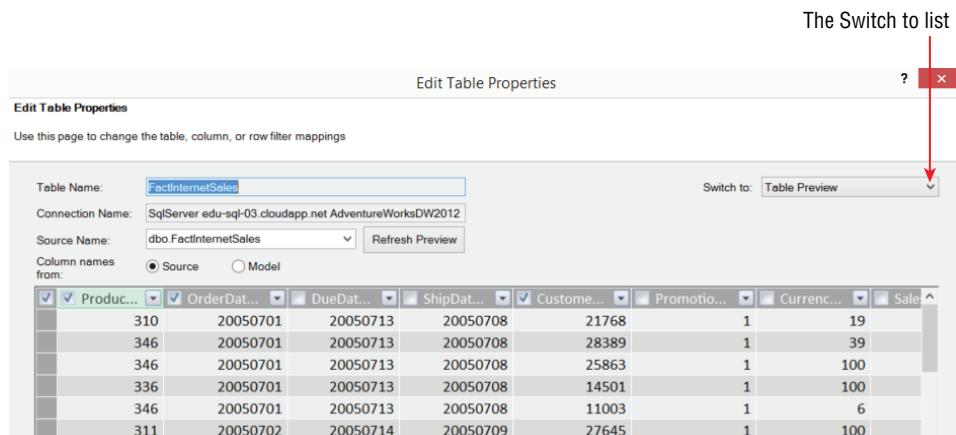


Figure 6-10: Power Pivot Edit Table Properties window

5. Enter the following TSQL syntax in the Sql statement window:

```

SELECT
    [dbo] . [FactInternetSales] . [ProductKey] ,
    [dbo] . [FactInternetSales] . [OrderDateKey] ,
    [dbo] . [FactInternetSales] . [DueDateKey] ,
    [dbo] . [FactInternetSales] . [ShipDateKey] ,
    [dbo] . [FactInternetSales] . [SalesTerritoryKey] ,
    [dbo] . [FactInternetSales] . [CustomerKey] ,
    CAST([dbo] . [FactInternetSales] . [SalesAmount]
AS INT) SalesAmount
FROM [dbo] . [FactInternetSales]

```

In the previous script, the SalesAmount has been changed from the original source data type to an integer using the CAST scalar function. You could have also done this using functions like CONVERT or ROUND. Click the Save button to update the data. Revisit the SalesAmount column and notice that the decimal places are still there, but now the values are all rounded up, instead of showing a value like 4.99 Power Pivot shows 5.00. This is because of the format that Power Pivot has automatically selected. To remove the decimal places completely, click Home in the Power Pivot ribbon and reduce the number of decimal places by clicking that option in the Formatting section. The main point here is that the data is now less unique and Power Pivot can compress it at a higher level. Save the file before proceeding.

Working with Date and Time Data Types

Now let's move away from the numeric data types and onto another type that could be troublesome and cause over- or misuse of memory. Date and time data types, depending on how they are imported, can be very granular and unique. For example, take a look at Figure 6-11.

	DateAndTime
1	1/1/2005 12:00:01
2	1/2/2005 13:01:02
3	1/3/2005 14:02:03
4	1/4/2005 15:03:04
5	1/5/2005 16:04:05
6	1/6/2005 17:05:06
7	1/7/2005 18:06:07
8	1/8/2005 19:07:08
9	1/9/2005 20:08:09

Figure 6-11: List of dates and times

Date and/or time columns are common when working with data warehouses. In many cases, several date values are associated with a single fact table. If data warehouse best practices are applied, the actual data has been converted to an integer, and a relationship exists between the fact table and some type of date dimension. In addition to the key values, other values like actual dates, similar to what's seen in Figure 6-11, are stored in the Date dimension table. Because of the high cardinality of the data, the compression ratio may be low. Deciding to import a date and time value involves the following considerations:

- Requirements on having both date and time values
- If time is stored, store each value separately.
- If a calculation is performed on two dates but the dates are not needed, include only the result of the calculation and not the data.

Implementing these techniques can help minimize the amount of memory the Date and Time dimension requires. The following code sample illustrates a few methods that can be leveraged:

```
SELECT  
    CAST([DateAndTime] AS DATE) DateOnly,  
    DATEPART(Hour, [DateAndTime]) HourOnly,  
    DATEPART(Minute, [DateAndTime]) MinuteOnly,  
    DATEPART(Second, [DateAndTime]) SecondOnly,  
    DATEDIFF(Day, [DateAndTime], GETDATE()) DateCalculataion  
FROM  
(  
    SELECT '1/1/2005 12:00:01' [DateAndTime]  
    UNION ALL  
    SELECT '1/2/2005 13:01:02' [DateAndTime]  
    UNION ALL  
    SELECT '1/3/2005 14:02:03' [DateAndTime]  
    UNION ALL  
    SELECT '1/4/2005 15:03:04' [DateAndTime]  
    UNION ALL  
    SELECT '1/5/2005 16:04:05' [DateAndTime]  
    UNION ALL  
    SELECT '1/6/2005 17:05:06' [DateAndTime]  
    UNION ALL  
    SELECT '1/7/2005 18:06:07' [DateAndTime]  
    UNION ALL  
    SELECT '1/8/2005 19:07:08' [DateAndTime]  
    UNION ALL  
    SELECT '1/9/2005 20:08:09' [DateAndTime]  
) DatesAndTimes
```

You can do this by using the approach outlined in the discussion on decimals. Each method reduces the high degree of uniqueness in each column, resulting in a higher compression ratio.

Working with Columns or DAX Calculated Measures

The final topic on optimizing Power Pivot models focuses on something that most users overlook. Fact tables typically contain most of the additive values in the data warehouse. During the load process, some massaging is performed to ensure the validity and accuracy of the data. In most cases, the developer includes more columns in the fact table than are necessary. For example, in a sales scenario, the table may include a SubTotal, TaxAmt, Freight, and Total columns. Figure 6-12 shows a sample dataset.

SubTotal	TaxAmt	Freight	TotalDue
\$2,369.97	\$189.60	\$59.25	\$2,618.82
\$2,380.47	\$190.44	\$59.51	\$2,630.42
\$1,228.83	\$98.31	\$30.72	\$1,357.86
\$1,194.97	\$95.60	\$29.87	\$1,320.44
\$1,183.47	\$94.68	\$29.59	\$1,307.73
\$578.46	\$46.28	\$14.46	\$639.20
\$565.47	\$45.24	\$14.14	\$624.84
\$1,183.47	\$94.68	\$29.59	\$1,307.73
\$2,414.45	\$193.16	\$60.36	\$2,667.97
\$120.47	\$9.64	\$3.01	\$133.12
\$88.98	\$7.12	\$2.22	\$98.32
\$62.98	\$5.04	\$1.57	\$69.59
\$2,638.94	\$211.12	\$65.97	\$2,916.03
\$2,465.28	\$197.22	\$61.63	\$2,724.13

Figure 6-12: Sample dataset

During data review, you should notice that a summation of the SubTotal, TaxAmt, and Freight columns equals the TotalDue column. So, is the Total column really needed? Definitely not when building the Power Pivot model. Instead, create a Calculated Measure using DAX. The syntax could be similar to either of the following:

```
Total Due:=SUMX(SalesOrderHeader, ([SubTotal]+[TaxAmt]+[Freight]))
```

```
Total Due:=SUM([SubTotal])+SUM([TaxAmt])+SUM([Freight])
```

This saves space in memory because a column is completely omitted from the import process. Demonstrations of how to create calculated columns and DAX measures are provided later in this chapter.

Optimizing the Power Pivot Model for Reporting

When interacting with the model, a clean and consistent Field List should be provided. There are a few things that you can do to ensure the intuitiveness of the model, such as hiding unnecessary columns, naming columns so that they resonate with end users, categorizing and formatting columns appropriately, and sorting data properly. The following sections provide tips to assist you in building a proper model for reporting.

Understanding Power Pivot Model Basics

The basic steps you should take immediately after importing data are as follows:

1. Hide columns that are not needed for reporting.
2. Rename columns so that they resonate with end users.
3. Format columns.
4. Specify how data should aggregate.

Following these few steps helps ensure that the model is easy to use and minimizes what appears in the field list.

Hiding and Renaming Columns

Hiding and renaming columns are done when working with Excel workbooks. Simply right-click a column header or table in the model to get the context menu shown in Figure 6-13, and select either Hide from Client Tools to hide the column or table, or Rename Column to change the name of a column or table.

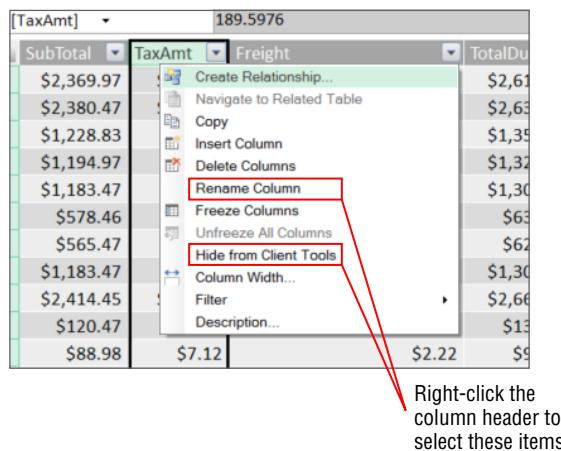


Figure 6-13: Rename Column and Hide from Client Tools context menu

Although seemingly trivial, this step could determine whether someone will use the model for analysis. Overpopulated field lists or improperly named columns often become deterrents to usage and consumption from an end-user perspective. Tables 6-1, 6-2, and 6-3 demonstrate how to hide and rename columns in a model. Table 6-1 lists tables and the corresponding columns that you should hide, and Table 6-2 lists the tables and the corresponding names that the table will be renamed to.

Table 6-1: Columns to Hide

TABLE NAME	COLUMNS TO HIDE
DimProduct	ProductKey and ProductSubcategoryKey
DimProductCategory	ProductCategoryKey
DimProductSubCategory	ProductSubcategoryKey and ProductCategoryKey
DimDate	DateKey, DayNumberOfWeek, and MonthNumberOfYear
DimSalesTerritory	SalesTerritoryKey and SalesTerritoryAlternateKey
FactInternetSales	ProductKey, OrderDateKey, DueDateKey, ShipDateKey, CustomerKey, and SalesTerritoryKey

Table 6-2: Tables to Rename

TABLE NAME	NEW TABLE NAME
DimProduct	Product
DimProductCategory	Product Category
DimProductSubcategory	Product Subcategory
DimDate	Date
DimSalesTerritory	Sales Territory
FactInternetSales	Internet Sales

Table 6-3: Columns to Rename

TABLE NAME	COLUMN NAME	NEW COLUMN NAME
Product	EnglishProductName	Product Name
Product Category	LargePhoto	Product Image
Product Category	EnglishProductCategoryName	Product Category Name
Product Subcategory	EnglishProductSubcategoryName	Product Subcategory Name

Continues

Table 6-3 (continued)

TABLE NAME	COLUMN NAME	NEW COLUMN NAME
Date	FullDateAlternateKey	Actual Date
Date	EnglishDayNameOfWeek	Day
Date	EnglishMonthName	Month
Date	CalendarQuarter	Quarter
Sales Territory	SalesTerritoryRegion	Region
Sales Territory	SalesTerritoryCountry	Country
Sales Territory	SalesTerritoryGroup	Group
Sales Territory	SalesTerritoryImage	Image
Date	CalendarYear	Year
Internet Sales	SalesAmount	Sales Amount

Formatting Columns

The next step is formatting the columns in the model properly. For example, if the value is money, a percentage, text, and so on, format it as such by selecting Home from the Power Pivot ribbon and locating the Formatting section. Several options appear in that section, as shown in Figure 6-14.

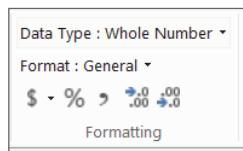


Figure 6-14: Power Pivot ribbon Formatting section

The Data Type drop-down list determines the options available in the Format drop-down list. In addition, a few standard items appear such as currency type, commas, and decimal place. To change the currency, simply click the drop-down arrow located next to the currently selected currency symbol. The formatting options specified here will surface when building reports in Power View.

Specifying Aggregation Rules

The final step is to set the aggregation rules. Power Pivot automatically aggregates—more specifically, sums—numeric data. In some cases, you may expect this. However, aggregation is not required, and just does not make sense when working with years (2012, 2013, 2015) or day numbers (1, 2, 3, 4). Power Pivot

provides a Summarize By option to stop this behavior. Click the Advanced tab in the Power Pivot window and click the drop-down box located in the Summarize By section, shown in Figure 6-15.

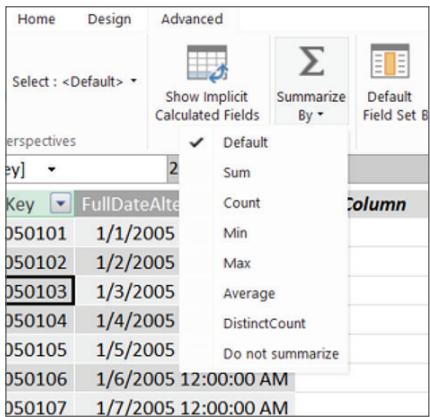


Figure 6-15: Summarize By context menu display options

Similar to the formatting you did previously, specifying aggregation rules also affects how Power View displays reporting options. With a column selected, simply select Do not summarize from the list of available options to stop a particular column from aggregating when surfaced in a Power View report. On the other hand, if you want a column to aggregate in a specific way, select the corresponding aggregation level. Then, when that column is added to a Power View report, the selected value is automatically aggregated.

To demonstrate this, select the date table and then click the Advanced tab in the Power Pivot ribbon. Quarter and Year are both numeric data types; however, aggregating this type of data makes no sense. Click the Quarter column and select Do not summarize from the Summarize By drop-down menu. Repeat these steps for Year and any other columns that do not require summarization.

Adding All Necessary Relationships

With the model nicely formatted, the next step is to ensure that additive values aggregate properly when dimension attributes are selected. To do this, you must create relationships between the table that contains the additive values—which is typically the fact table—and the dimension tables. Power Pivot provides a few ways to do this. Note that Power Pivot automatically preserves any preexisting relationships in a relational data source when it imports the tables. However, if you add tables from the same source after the fact, or import them from other relational sources or other data source types, you will need to create the relationships manually.

Assume that FactInternetSales and DimProduct have been imported into a tabular model. A relationship between the ProductKey column on both tables is automatically created during import because that relationship exists in the underlying data source. Now suppose that the DimCustomer table is added later to the model. Looking at the diagram view shown in Figure 6-16, a relationship clearly does not exist.

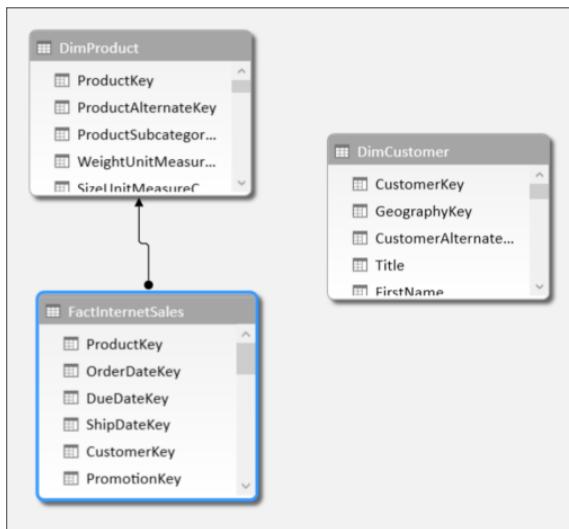


Figure 6-16: Power Pivot diagram view showing missing relationship

As mentioned earlier, there are several ways to create relationships. These are outlined in the following sections.

Using the Table View Method

One method is to drag one column onto the corresponding column in another table using the diagram view. If you have experience with ERD diagramming tools, this may be the most intuitive approach. If not, then creating relationships in the table view may be the best approach.

To create a relationship using the table view, follow these steps:

1. **Open the Power Pivot model.** If you don't already have this opened, this is the model you created earlier in this chapter.
2. **In the Power Pivot ribbon, click the Get External Data icon.** Then choose Existing Connections.
3. **Click the connection associated with the AdventureWorks data warehouse.** Then click Open.
4. **Ensure that the first radio button is selected.** Click Next.

5. Click the check box on the same row as the DimCustomer table. Enter Customer in the Friendly Name column.
6. Click the Preview and filter button. Import only the CustomerKey, FirstName, and LastName columns.
7. Click the Finish then Close button.
8. Open the Internet Sales table and right-click the header for the column that needs the relationship (CustomerKey). Now select Create Relationship from the context menu that appears (see Figure 6-17). The Create Relationship window opens with the current table and active column selected in the Table and Column drop-down list.

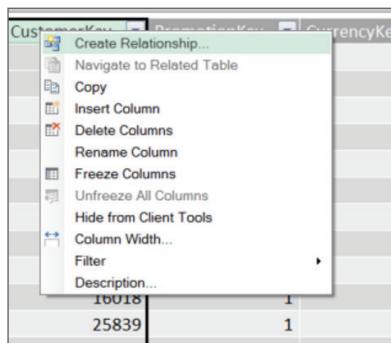


Figure 6-17: Context menu displaying Create Menu option

9. Select the Related Lookup Table from that drop-down list. If the related table contains a column with the same name selected in the Column drop-down list, that column will automatically be selected in the Related Lookup Column drop-down list. If not, manually select it from the list. Figure 6-18 illustrates a relationship between two tables.

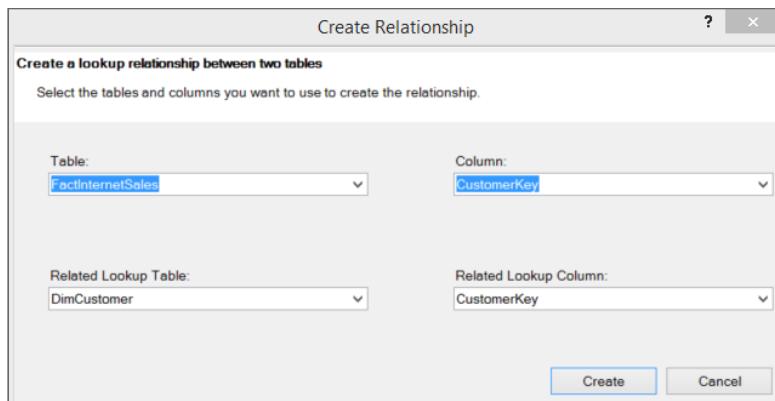


Figure 6-18: Create Relationship window

Using the Create Relationships Window

In addition to the table view method, you can also open the Create Relationship window by selecting the Design option in the Power Pivot window, then clicking the Create Relationship icon. It's also possible to view all existing relationships by clicking the Manage Relationships icon.

Before closing this section, a few things should be mentioned about relationships. First, many-to-many and self-referencing (parent-child) relationships are not supported out of the box. Even so, both types can be implemented using DAX patterns. Also, multiple relationships can exist between two tables, but only one can be active—for example, when importing the FactInternetSales table and the DimDate table into Power Pivot. Change the model to diagram view, shown in Figure 6-19.

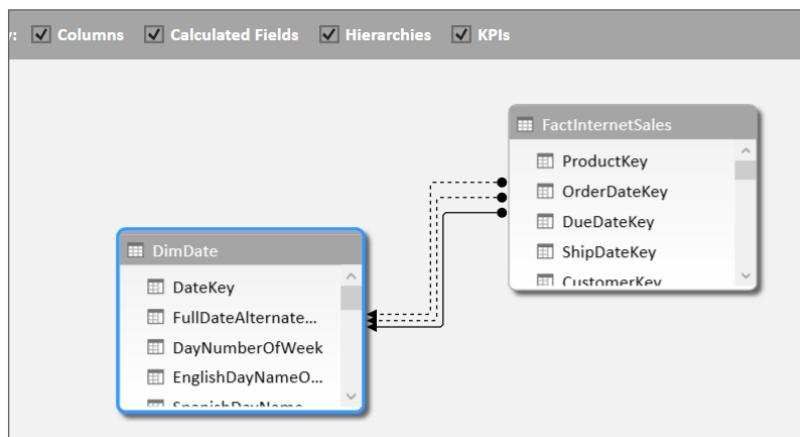


Figure 6-19: Power Pivot inactive relationships

Notice that several relationships exist between the two tables, but only one has a solid line and the others are all dashed. The dashes denote that the relationships exist but are not used by default in query or filter context. For example, in the Power Pivot window three relationships exist, but two are inactive.

1. **Click Home in the Power Pivot window.** Then click the drop-down arrow below the Pivot Table icon.
2. **Select PivotTable from the list of available choices.**
3. **Click OK on the Create Pivot Table window.**
4. **Drag Year to the Rows section from the date table and drag Sales Amount to the Values section.** By default, the Sum of the sales amount is aggregated by year using the relationship established between the OrderDateKey and FactInternetSales.
5. **If necessary, aggregate by the inactive key columns.** You can do this by creating custom DAX measures or by using role-playing dimensions (discussed later in this section).

Adding Report Capabilities with Power Pivot

Using Power Pivot, you can also add additional reporting capabilities visibly and interactively to the Power View field list to assist in creating a more user-friendly reporting environment. Chapter 12 provides some examples of how these properties are used in reporting. To view and configure these options, click the Advanced tab. Under the Reporting Properties section, you can see that these three options can be configured:

- Data Category
- Default Field Set
- Table Behavior

The Data Category is probably the least advertised of the three. To assign a category to a column header, click the drop-down list labeled Data Category and select More categories from the list of available choices. Figure 6-20 shows a complete listing of the available categories.

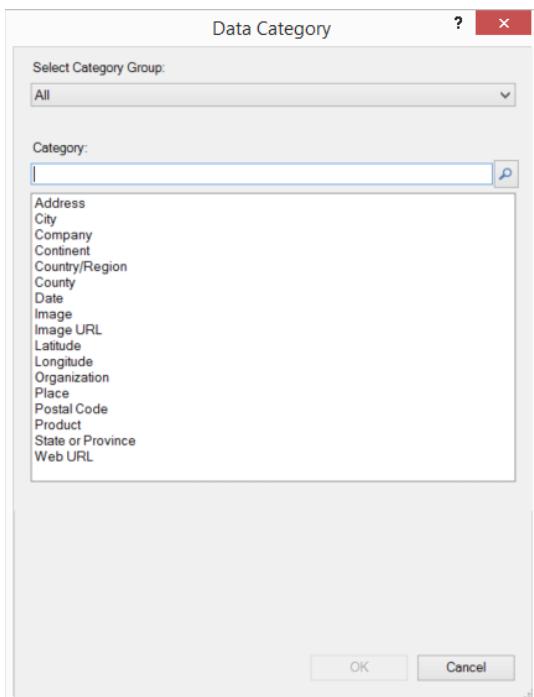


Figure 6-20: Power Pivot Data Category window

Select the category and click OK.

The category of a column modifies the behavior of the presented data. For example, if a database includes images, they must be categorized as Image; but

if you access the image via a web address (URL), it should be set to Image URL. Similar to the Image URL category is the Web URL category. To add a hyperlink to a report, that column must be categorized as such. You can add hyperlinks in other ways, but you should take this approach for data stored in the model. In addition, if you import geospatial data (City, State, Zip Code, and so on), each column should be set to the corresponding category. This adds a visualization to the Power View field list.

To demonstrate, follow these steps.

1. **Select the date table.** Then click the header for the Actual Date column.
2. **Choose Advanced in the Power Pivot ribbon and expand the Data Category drop-down list.** This list is located in the Reporting Properties section. Select Date from the list of available choices. Repeat these steps, this time selecting the Sales Territory table and specifying Image as the category for the Image column and Country/Region for the Country column.
3. **Select the reporting property is the Default Field Set.** Select the Product table and then click the Default Field Set icon in the ribbon to configure this on the Advanced tab of the Power Pivot window. This opens the Default Field Set window shown in Figure 6-21.

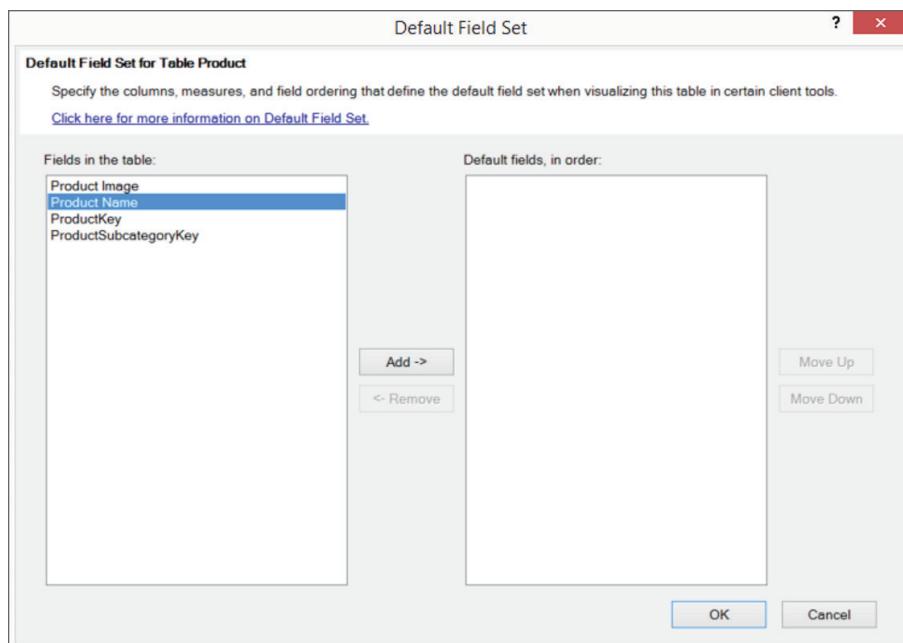


Figure 6-21: Default Field Set window

4. **Configure this property.** Select a column and then click Add. The field moves from the Fields in the table list box to the Default fields list box. Perform this action for both Product Image and Product Name.
5. **Change the column order.** Remember that column order is important. When all the columns are added, click OK and the property is configured. With the property set, when an end user builds reports with Power View, double-clicking the table in the field list automatically adds the columns to the report in a table in the order specified.
6. **Set the final reporting property which is Table Behavior.** During the configuring process, you can set Table Behavior properties for not only changing the behavior, but also the visualization types and default grouping behavior when you access them via Power View reports. To demonstrate this, open the Power Pivot model you are using for all demonstrations in this chapter. Open the Product table, and then open the Table Behavior window shown in Figure 6-22.

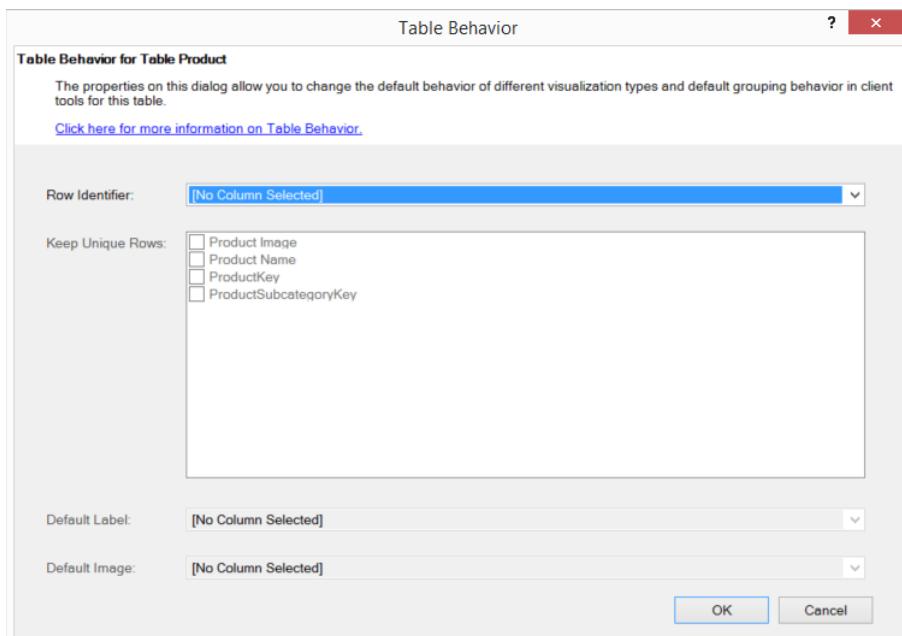
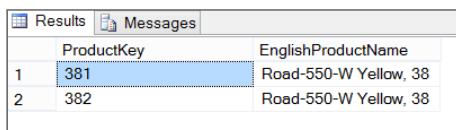


Figure 6-22: Power Pivot Table Behavior window

7. **Configure the grouping option by specifying two properties.** The first one is the Row Identifier, which uniquely identifies each row. This is similar to a primary key on a table. In this example, select the ProductKey column. The second property is Keep Unique Rows. This option is used to show values as unique even when they are duplicates. For example, in the

Product table, certain Product Names are duplicates, but the ProductKey is different, as shown in Figure 6-23.



The screenshot shows a 'Results' view from a Power BI model. It has two columns: 'ProductKey' and 'EnglishProductName'. Row 1 has ProductKey 381 and EnglishProductName 'Road-550-W Yellow, 38'. Row 2 has ProductKey 382 and EnglishProductName 'Road-550-W Yellow, 38'. The 'Messages' tab is also visible at the top.

	ProductKey	EnglishProductName
1	381	Road-550-W Yellow, 38
2	382	Road-550-W Yellow, 38

Figure 6-23: Multiple rows for the same product name

Notice, how Road-550-W Yellow, 38 appears multiple times, but with different product keys. When reporting this, you may need to show each row individually. Specifying these options ensures that additive values do not aggregate into one product name, but instead show individual rows based on the key value and the name value without actually including the key value in the report.

8. **Configure the final two properties: Default Label and Default Image.** These properties actually work together when reporting with Power View. Set the Default Label to Product Name and Default Image to Product Image. Click OK to configure the property.

Now when you create a Power View report, and when you add certain visualization types to the report, the image and name will appear together in that display. These options and all that have appeared in this section are explained and shown in more detail in Chapter 12.

Adding Calculated Columns and DAX Measures

So far, most of the discussion has focused on adding properties to ease reporting. The focus will now shift slightly to adding items to the model that enhance data analytics. Two aspects of Power Pivot lend themselves effectively to this approach. Those are calculated columns and DAX measures; this section explains both and how to create them.

Using Calculated Columns for Row-by-Row Values

What are calculated columns? So basically, a calculated column is a column that performs row-by-row calculations based on DAX expressions. The result of the following DAX expression is a concatenation of the Last Name and First Name columns separated by a column as illustrated in the following steps:

1. **Open the Customer table in the Power Pivot model that has been created in this chapter and combine columns.** Notice that there is a FirstName and LastName column. Neither one may have true significance when

reporting, but they both will likely be added to the model. So instead of providing the end user with two separate columns, combine them into a single column. To do this, double-click Add Column, which is located directly to the right of LastName, and enter the value **Full Name**. Press Enter when complete.

2. Click in the cell directly below the Full Name Header. Enter the following and then press enter to complete:

```
= [LastName] & ", "& [FirstName]
```

3. **Hide both First and Last Name columns because they probably don't serve a true purpose.** You could have also used TSQL to perform this concatenation and import only the Full Name itself. That is actually the best approach because it minimizes the amount of memory dedicated to storing the Customer table.

NOTE Remember, the omission of a column is the best technique for truly maximizing the use of memory.

To further demonstrate calculated columns, follow these steps:

1. **Open the Product table.** The Product table has a relationship to the Product Subcategory table, which has a relationship to the Product Category table. What the end user is requesting from this is a hierarchy that rolls up from category, to subcategory, and finally product.
2. **Add a couple of calculated columns.** Creating hierarchies is discussed at length in the next section. However, to prepare for this hierarchy, you must add some columns. With the Product table open, add a new column name **Subcategory**. Add the following code as the DAX expression:

```
=RELATED('Product Subcategory'[Product Subcategory Name])
```

3. **Repeat steps 1 and 2, but instead name the column Category and use the following code as the DAX expression:**

```
=RELATED('Product Category'[Product Category Name])
```

4. **After the two columns are added, click the drop-down arrows located to the right of each.** A distinct list of values appears.

Using DAX Measures

Both examples of calculated columns in the previous section demonstrate row-by-row values, but what about totals, sums, and averages? How are they added to the model?

You can actually use two methods to include aggregations. As stated earlier, Power Pivot automatically aggregates data based on the data type. In most cases, this approach alone does not meet the project requirements. As a result, Power Pivot includes a section that allows you to create custom DAX measures, which is available in each table. The result of either method is a DAX measure that you can use when analyzing the data. For example, when using the data for reporting, you can use these measures with attributes such as Product Names and Product Categories. To demonstrate this, open the Internet Sales table and look directly below the data to see the DAX measures section. This is shown in Figure 6-24. These grayed-out columns are used to create measures with the DAX expression language.

477	20070728	15155	\$5	20070803	20070807	
477	20070731	15548	\$5	20070812	20070807	4
477	20070801	16982	\$5	20070813	20070808	4
477	20070801	14329	\$5	20070813	20070808	4

Figure 6-24: DAX measures section

DEFINITION Measures are additive values used for analysis. For example, Total Sales, Number of Customers, Previous Year Sales, and so on. These types of values are generally aggregated across the different dimensions instead of row by row.

Next, open the Internet Sales table and click in the first measure column directly below the CustomerKey column. Select Home in the Power Pivot ribbon and locate the AutoSum drop-down value in the Calculations section. Click the drop-down arrow and select Distinct Count from the list of available choices.

Power Pivot automatically generates a DAX expression that results in a distinct count of the CustomerKey values in the above column. Figure 6-25 provides an explanation of the syntax.

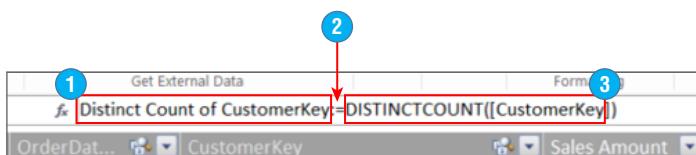


Figure 6-25: DAX measure syntax

1. The name of the column. Instead of Distinct Count of CustomerKey, which makes no sense at all, change that value to **Customer Count**.

2. Separates the column name from the actual expression. Leave that unchanged because it is required.
3. The DAX expression used to calculate the value.

Item three would be the easy way, but unfortunately the AutoSum option cannot create more complex calculations. This is when you must create measures manually. To do so, follow these steps:

1. **Ensure that the Internet Sales table is open.** Then hide the Sales Amount column.
2. **Add an AutoSum function.** Click in the cell directly below the Sales Amount column located in the measures section. Use the AutoSum function to add a Sum of Sales Amount and change the name to **Total Sales**.
3. **Compare current year sales to previous year sales.** This is one of the most common scenarios in a sales organization. Before you can create the calculation, a slight digression must be taken.
4. **Assign the calculation as a Mark as Date Table.** Open the date table and click Design in the Power Pivot ribbon. Locate the icon labeled Mark as Date Table and click the drop-down arrow. Select Mark as Date Table from the list of available choices; the Mark as Date Table window opens. You must configure this to perform temporal-based data analysis.
5. **First make sure that the date table contains a unique date value on each row.** This has already been done for the imported date table. Therefore, select Actual Date in the drop-down list if it is not already selected and click OK.
6. **After the trip, return to the Internet Sales table, click in the column directly below the Total Sales column located in the measures section.** Enter the following DAX Expression:

```
Preview Period Total Sales:=  
CALCULATE([Total Sales], PREVIOUSYEAR('Date'[Actual Date]))
```

It will appear as (blank) because it is dependent upon the selection of a Year from the date table.

7. **Watch the calculation in action.** To do so, create a pivot table and add Year to Columns, Country to Rows, and Total Sales and Previous Period Total Sales to Values. The result is shown in Figure 6-26.

The screenshot shows a Pivot table with 'Column Labels' and 'Row Labels'. The columns are grouped by year: 2005, 2006, and 2007. Each year group contains 'Total Sales' and 'Preview Period Total Sales'. The rows list countries: Australia, Canada, France, Germany, United Kingdom, United States, and Grand Total. The data shows sales figures for each country in each year, with the 2005 sales being zero.

	Column Labels		2005				2006				2007			
Row Labels	Total Sales	Preview Period Total Sales	Total Sales	Preview Period Total Sales	Total Sales	Preview Period Total Sales	Total Sales	Preview Period Total Sales	Total Sales	Preview Period Total Sales	Total Sales	Preview Period Total Sales	Total Sales	
Australia	\$1,308,971		\$2,154,210		\$1,308,971.00	\$3,033,443	\$2,154,210.00		\$1,308,971		\$2,154,210.00		\$2,154,210.00	
Canada	\$146,820		\$621,568		\$146,820.00	\$535,683	\$621,568.00		\$146,820		\$621,568.00		\$621,568.00	
France	\$180,560		\$514,924		\$180,560.00	\$1,026,217	\$514,924.00		\$180,560		\$514,924.00		\$514,924.00	
Germany	\$237,770		\$521,209		\$237,770.00	\$1,058,293	\$521,209.00		\$237,770		\$521,209.00		\$521,209.00	
United Kingdom	\$291,572		\$591,566		\$291,572.00	\$1,298,097	\$591,566.00		\$291,572		\$591,566.00		\$591,566.00	
United States	\$1,100,475		\$2,126,583		\$1,100,475.00	\$2,838,164	\$2,126,583.00		\$1,100,475		\$2,126,583.00		\$2,126,583.00	
Grand Total	\$3,266,168		\$6,530,060		\$3,266,168.00	\$9,789,897	\$6,530,060.00		\$3,266,168		\$6,530,060.00		\$6,530,060.00	

Figure 6-26: Pivot table showing previous year total sales

Notice that there are no sales in the year 2005, but for 2006, the previous year sales correspond to the 2005 sales. This pattern continues throughout all the years.

Although this is just an introduction to calculated columns and DAX measures, you should realize that DAX expressions can extend and enhance the model, enabling you to perform most types of data analysis by leveraging this expression language. More examples are provided throughout this section and the rest of the book.

Creating Hierarchies and Key Performance Indicators (KPIs)

With all the properties set for tables and columns, it is time to add some additional functionality to the model. Two features that add ease of use to the model from both a visual and data perspective are hierarchies and KPIs.

Using Hierarchies

Let's start the discussion with hierarchies. The hierarchies remove guesswork involved with figuring out how data aggregates up and down dimension attributes when the consumer of the model is unfamiliar with the data. A perfect example is the Date. How the data rolls up is pretty obvious: Year, Quarter, Month Date, and Day. However, what about attributes in the Product or Department dimension? How do they aggregate? To see how, follow these steps:

1. Open the Power Pivot model you have used so far.
2. Change from Table view to Diagram view.
3. Locate and select the date table. Two icons appear in the upper right corner, shown in Figure 6-27.

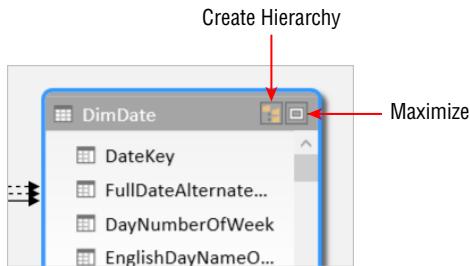


Figure 6-27: Diagram view showing Create Hierarchy and Maximize icons

4. Click the Maximize icon to create a large surface space to work with, then click the Create Hierarchy icon. This automatically adds a hierarchy named Hierarchy1 to the end of the items in the date table; rename it to Date Hierarchy.
5. Right-click the Year column and select Add to Hierarchy and then Date Hierarchy from the context menus.
6. Repeat these steps for Quarter, Month, and Actual Date in that order. That's it; the hierarchy is now created and you can use it in both pivot tables and Power View reports.

NOTE You can perform these steps using a drag-and-drop approach instead. For example, after creating the hierarchy, drag the columns onto it in the order that you expect the date to aggregate. As a challenge, open the Product table and create a hierarchy that aggregates the date in this order: Category, Subcategory, and then Product Name.

Using KPIs

The next topic, KPIs, are commonplace when performing data analysis. They are used on scorecards, dashboards, and any other type of reporting tools. KPIs offer quick insight and understanding of data, without actually knowing the value. For example, when comparing current year sales to last year sales, the primary concern is whether or not sales have gone up, down, or remained constant. Does the value itself really matter? At some level, yes, but at a high level, not really. Creating KPIs in Power Pivot requires very little coding, with the exception of the actual values that will be measured.

To demonstrate, open the Internet Sales table and add two new DAX measures using the following code sample:

```

Preview Period Total Sales:=
CALCULATE([Total Sales], PREVIOUSYEAR('Date'[Actual Date])) 

YOY Growth:=
DIVIDE(([Total Sales] - [Preview Period Total Sales]), [Total Sales])

```

The first calculates previous year sales, and the second calculates a value that represents year-to-year growth. Right-click YOY Growth and select Create KPI from the context menu. The target value for the KPI can be based on a measure that resides in the model or an absolute value. This example uses an absolute value of 1 because the goal is to identify growth based on a percentage. Therefore, select the radio button labeled Absolute value and enter 1 as the value.

The next step is to define the thresholds. First, replace 0.4 with 0; this means that any value less than zero is represented by a red icon. Then replace 80 with .10. This does two things: It specifies that anything between 0 percent and 5 percent is yellow and anything greater than 10 percent is green. Figure 6-28 illustrates how the screen should look at this point.

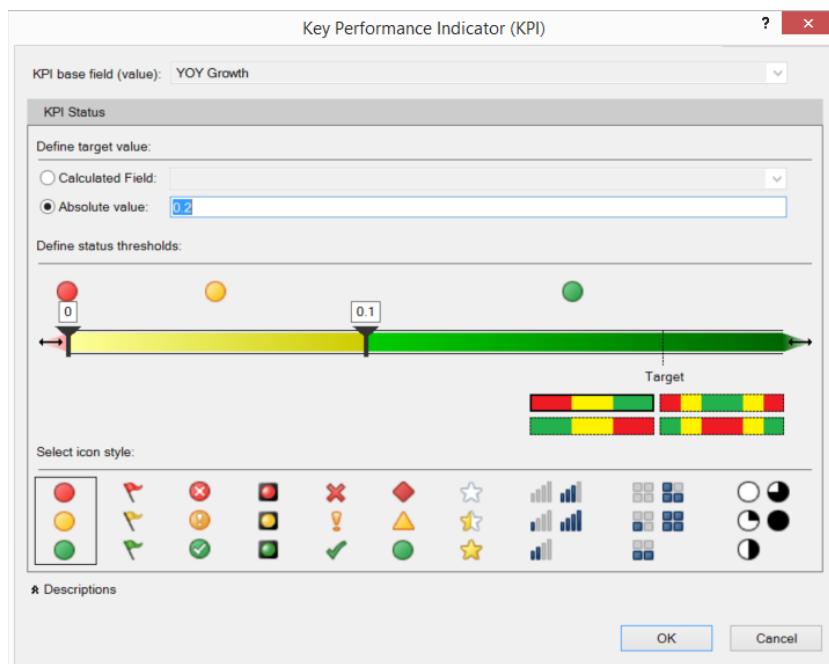


Figure 6-28: Key Performance Indicator Window

You can also change the style of the icons and the direction of the thresholds. For now, accept the defaults and click OK. To test this, create a pivot table and add Year to Columns, Country to Rows, and the KPI Status for the measure YOY Growth to Values.

Sorting Your Data to Meet End-User Needs

Before this chapter is wrapped up, a few more things should be pointed out. The first is very subtle, but when it comes to reporting, it makes a big difference when analyzing data. Consider the chart shown in Figure 6-29.



Figure 6-29: Line graph with Month sorted by Month Name alphabetically

Notice how the months on the x-axis are not in the correct order. By default they are sorted alphabetically, which does not lend itself at all to data analytics. To fix it, Power Pivot provides the ability to sort by a different column. To demonstrate, open the date table and select the Month column. In the ribbon, ensure that the Home tab is active and locate the Sort and Filter section. Click the drop-down arrow associated to the Sort by Column icon; the Sort By Column window opens. Select MonthNumberOfYear from the Column drop-down list in the By section and click OK. A chart similar to the one shown in Figure 6-30 now appears when you create a report based on the same data.



Figure 6-30: Line graph with Month sorted by month number

One thing to note here is that one small requirement must be in place for this to work. Each value being sorted must be associated to a single distinct value in the Sort By column. In other words, there must be a one-to-one relationship. In the previous example, each month must be associated to a single value in the MonthNumberOfYear column: January = 1, February = 2, March = 3, and so on.

Implementing Role-Playing Dimensions

To end this chapter, let's discuss something not often thought of until later in the product when an end user realizes a certain feature does or does not exist, and they don't quite understand how to use it. These are role-playing dimensions. *Role-playing dimensions* are dimensions used for multiple purposes. For example, Date is the most common: The Date dimension could be a Sales Date, Order Date, Due Date, and so on. Out of the box, Power Pivot models do not support role-playing dimensions. As a result, you could use a couple of patterns to implement this. This section explains and demonstrates each pattern.

As stated, two approaches exist, one involves calculated columns and the other involves importing the date table multiple times. The approach you use depends on a few factors, among the main ones of which is machine or server resources. Obviously, importing the date table multiple times requires more data, which may not be an option due to hardware limitations. On the other hand, this approach does provide a very intuitive approach for end users and is relatively easy to develop. The other approach, calculated columns, does not require the same amount of resources, but does require creating, managing, and maintaining multiple calculated columns to accommodate the various roles that a dimension would play.

Regardless of which method is chosen, you should provide some type of instruction or documentation so that end users can understand how to properly model and consume the date. Without that information, someone may accidentally construe the model as incorrect or inaccurate. It is imperative that you provide a clear definition either of the calculated values or of the different date tables.

Using Calculated Columns

Remember back to the relationships column. Think about the inactive relationships that exist. This pattern actually leverages those relationships. Although they are not active as it relates to data analytics, you can use them when creating measures. To demonstrate, start with the calculated column pattern.

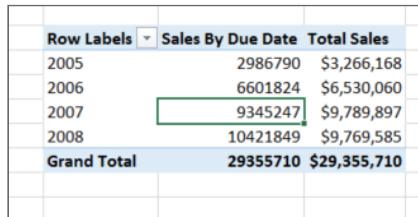
1. **Add the DAX expression.** Open the Power Pivot model you have used throughout this chapter. Open the Internet Sales table and click the cell

directly below the DueDateKey column in the measures section. Enter this code as the DAX expression:

```
Sales By Due Date:=  
CALCULATE(  
    SUM([Sales Amount]),  
    USERELATIONSHIP('Date'[DateKey],  
    'Internet Sales'[DueDateKey])  
)
```

This expression performs a calculation on Sales Amount that aggregates date attributes from the date table, such as year, quarter, and month, based on the Due Date.

2. **Observe the expression in action.** To do this, create a pivot table, and add Year to Rows and Total Sales, and add Sales By Due Date to Values. Figure 6-31 illustrates this.



The screenshot shows a Microsoft Excel pivot table with the following data:

Row Labels	Sales By Due Date	Total Sales
2005	2986790	\$3,266,168
2006	6601824	\$6,530,060
2007	9345247	\$9,789,897
2008	10421849	\$9,769,585
Grand Total	29355710	\$29,355,710

Figure 6-31: Pivot table showing role-playing calculated column

Notice the difference between the two values over time.

3. **Make the calculation easier to understand from an end-user perspective.** Rename Total Sales to Sales By Order Date. By doing this, that value and any subsequent value added create a more user-friendly model. In addition to these benefits is the fact that a single filter from a centralized date table is supported.

The challenge with this solution is the context in which the data is shown. For example, referring back to Figure 6-26, the data shown on each row is specific to the corresponding year. However, no correlation exists between the actual sale year and due date year. The values in the pivot table are specific to the date of the active relationship or the user relationship specific in the calculation.

Using Multiple Date Table Imports

The second approach is somewhat simpler from a model design perspective; however, from an end-user or consumption perspective, the results in the field list may not be as obvious. In addition, this approach may require more

resources—specifically, on the machine or server that hosts the model. To demonstrate, follow these steps:

1. Open the model you've been using throughout this chapter.
2. Import the date table again, but name it Ship Date.
3. Delete the existing relationship between ShipDateKey and the date table.
4. Create a relationship between Internet Sales and Ship Date using ShipDateKey and DateKey, respectively.
5. Create a new pivot table and add two Timeline filters. Connect one to the Actual Date from the date table and the other to FullDateAlternateKey from the ShipDate table.

The pivot table should resemble Figure 6-32.

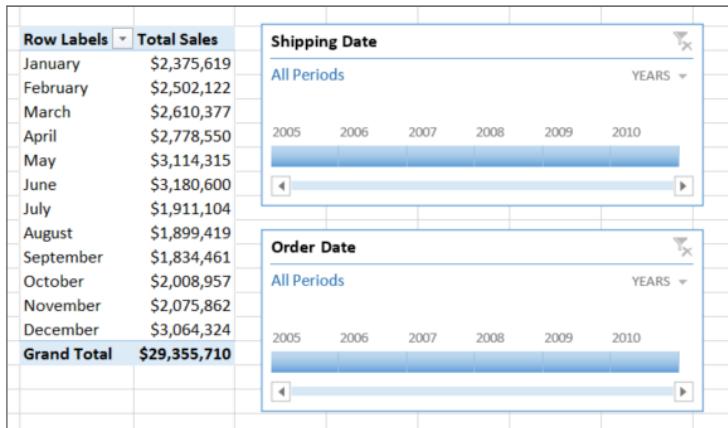


Figure 6-32: Pivot table with two date dimensions

Notice that the two different date tables initially could be a little confusing. However, this provides the flexibility not present in the first pattern. Now, using this method, an end user can select a period for the Order Date and a separate time period for Shipping Date, with both applying a single additive value. For example, the query could be answered to show all orders in 2007 due in June 2007.

Summary

In this chapter you learned how to use Power Pivot to build analytical models that you can use as sources for varying reporting systems. Examples and discussions are provided that explain and illustrate how to accomplish many tasks using Power Pivot.

The next chapter discusses a modeling technique that was introduced in SQL Server 2012, tabular modeling. It also explains and demonstrates concepts and features that will assist you in developing a flexible and scalable model.

Developing a Flexible and Scalable Tabular Model

The tabular model in Analysis Services provides an easy-to-use and handy way for business users to access their data. Determining the best way to make the model as friendly and optimized as possible can often be a tricky process. You must understand the correct way to build the model from the ground up, modify it to add needed enhancements, and adjust the model if it performs erratically. Only in this way can you create a flexible and scalable tabular model that will fit the needs of all your business users.

This chapter describes the best way to build an Analysis Services tabular model, with a focus on scalability and flexibility. You first learn the origination, definition, and benefits of using the tabular model. Next, you design your model using best optimization practices and then enhance it to make it more flexible. Finally, you discover how to performance-tune the model in case of issues in either querying or processing. By the end of this chapter, you should have an optimal tabular model for your business users.

Why Use a Tabular Model?

The tabular model is the younger sibling in the Analysis Services family, but may quickly turn out to be your favorite child. In this section, you learn about the origination of the tabular model, its benefits, and the differences between it and the multidimensional model.

As discussed in Chapter 5, Microsoft recently added the tabular model to the Analysis Services product. The tabular model took the idea of semantic modeling to the next level, by using the same business intelligence semantic model (BISM) as a multidimensional model and also storing the information in a column store format all in-memory! Microsoft also introduced a new querying language to retrieve information from the tabular model, called DAX (data analysis expressions), which has Excel-like syntax.

Understanding the Tabular Model

Now that you know the origins of the tabular model, you can better understand its definition. As a columnar, in-memory database, the tabular model provides a semantic layer for end users to easily access its information. Built on the Analysis Services platform, the tabular model acts as a “cube,” so your business user can “slice and dice” aggregated values.

TABULAR MODEL

A *tabular model* is a semantic model that stores its information in an in-memory, columnar structure to provide easy development and fast querying.

Because the tabular model was originally exposed through Excel, you can most easily think of it in Excel terms. A table equals a tab, a column is also a column, and a measure matches a cell. For a visual depiction of the relationship, see Figure 7-1.

The magic of the tabular model comes after creation, when a business user actually starts working with it. Using the tabular model in Excel or an analysis client tool such as Power View highlights just how much easier it is for business users to find the information they need in a quick and direct fashion.

Using the Tabular Model

Although creating a tabular model is a fairly straightforward process, it involves a number of high-level steps that include data load, data storage, and data consumption, as shown in Figure 7-2.

Column

Sum of SalesA...

Table/Tab

Measure/Cell

Figure 7-1: Comparison to Excel**Figure 7-2:** Tabular model usage process

In the data load phase, a developer populates the tabular model with information from a variety of sources. The developer then assigns relationships and creates additional calculations in the model, and the model gets published and processed on an Analysis Services server. Business users can then use and analyze that information through a client tool that can consume the tabular model. Later

on, this chapter explains how to populate the model so that the end user can consume it.

Comparing the Tabular and Multidimensional Models

The tabular model provides a lot of power, but how does it match its older sibling, the multidimensional model? This section outlines some of the reasons you would want to use a tabular model over the multidimensional model; but to get a more comprehensive view of the differences, and why you should use one model type over another, see Chapter 5.

As semantic layers, both models are valuable when you use a client tool built for querying them. They provide fast querying, the ability to slice and dice the information, and a business-friendly abstraction layer between the front- and back-end database. However, the tabular model offers some additional benefits that aren't found in the multidimensional model:

- Because the tabular model stores data in-memory, the query engine can access it more quickly.
- The columnar format that the tabular model uses to compress and store data can reduce the size of that data, which also increases querying speeds.
- People find the Excel-like query language that accesses the tabular model, DAX, easier to pick up than the multidimensional querying tool, MDX.

Chapter 8 discusses the multidimensional model, which also provides some benefits of its own.

Understanding the Tabular Development Process

The first step in having a scalable and flexible tabular model is building it that way in the first place! Understanding some key points while developing your model prevents you from having a lot of rework in the future. The development model contains eight steps, as shown in Figure 7-3.

These eight steps can be loosely grouped into three general phases:

1. **Designing the model:** In this phase you import data, design relationships and create calculations.
2. **Enhancing the model:** This phase involves adding hierarchies, designing perspectives and then adding partitions.
3. **Tuning the model:** Here, you spend time optimizing both the processing and the querying parts of the model.

The next three sections discuss these three phases in greater detail.

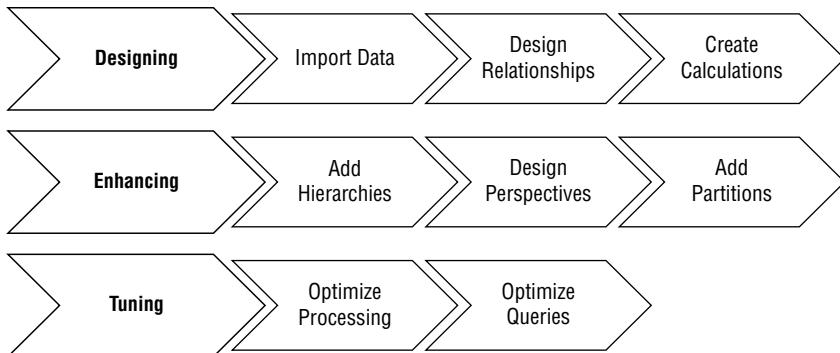


Figure 7-3: Tabular model development process

How Do You Design the Model?

You should follow the steps in this chapter only if you've followed the process to select the right semantic model, as discussed in Chapter 5. After you've made your decision to use a tabular model, you can continue on with this section. Note that the examples shown in this section are based on the AdventureWorks data warehouse database.

Importing Data

To build the tabular model, you start by importing data, which includes defining the correct data source, deciding how to pull the information, and correctly naming the objects to relate to the business. Ensure that you select data that supplements the model and relates to each other.

NOTE A great way to start developing the tabular model is to create a PowerPivot model in Excel (or have your business users create their own model!). This allows you to know exactly the data that the business needs, to learn the desired calculations, and to speed up your development time. You can even import the Power Pivot model directly into a tabular model, although that is outside the scope of this chapter.

To start importing data, follow these steps:

1. **Go into SQL Server Data Tools (SSDT).** Create a new project.
2. **In the New Project dialog box, select the Business Intelligence \Rightarrow Tabular Model \Rightarrow New Tabular project option.**

3. As shown in Figure 7-4, select a name, such as AdventureWorks. Click OK.

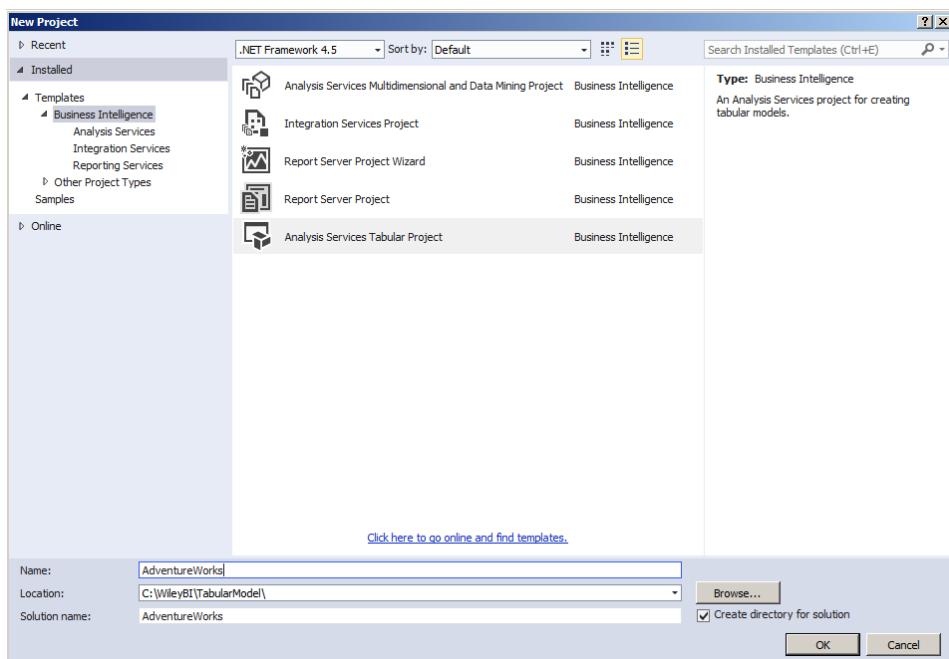


Figure 7-4: Create a new tabular model solution

4. After you've created the model, bring in the data you need. You can do this by going to the Model menu and selecting the Import from Data Source option. The data sources available to you include:
- Relational databases, such as SQL Server, Azure, Oracle, and any OLEDB or ODBC connection
 - Multidimensional sources, such as Analysis Services
 - Data feeds, such as a Reporting Services Report or a Windows Azure Marketplace feed
 - Text files, such as Excel or text

NOTE You can copy data directly into a pane. Analysis Services treats this pasted data as a table, as though you imported the data through a relational database. Be careful about using this method too often because it causes the data to be static. If the data could potentially change, store it in a table where you can bring it into the model on a reoccurring basis. A typical reason for using this method is a Yes-No dimension.

5. Decide how to pull the data into your model, either using a query or tables/views:

- Using a query to limit the data helps the model performance because it only has to store the columns needed for analysis.
- If you do not use a query, you can filter the columns the table pulls in as well, by selecting the Filter button and deselecting the columns not needed.

Either way, reducing the number of columns pulled into the model reduces the size of the model and decreases the query times for the business users.

6. For this scenario, choose the FactInternetSales table and click the Select Related Tables button. The model adds an additional six tables. You could continue to click the same button to add more tables, but for this example, just add two more tables: DimProductSubcategory and DimProductCategory. The completed Table Import Wizard looks like Figure 7-5.

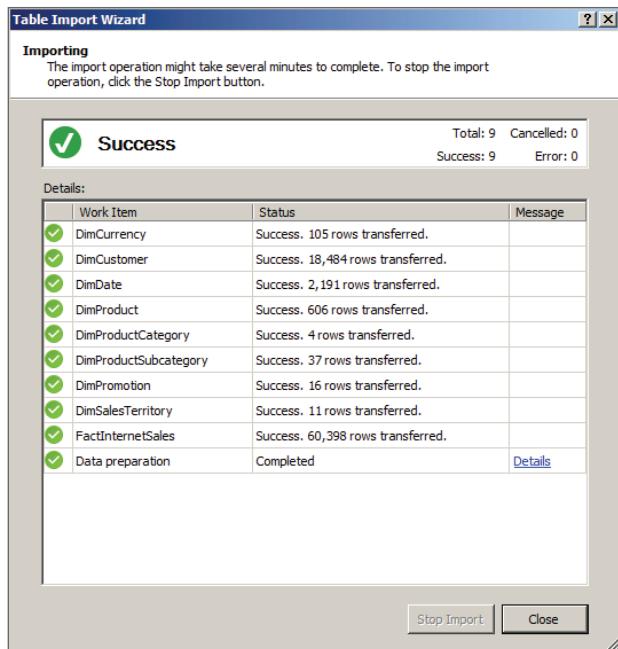


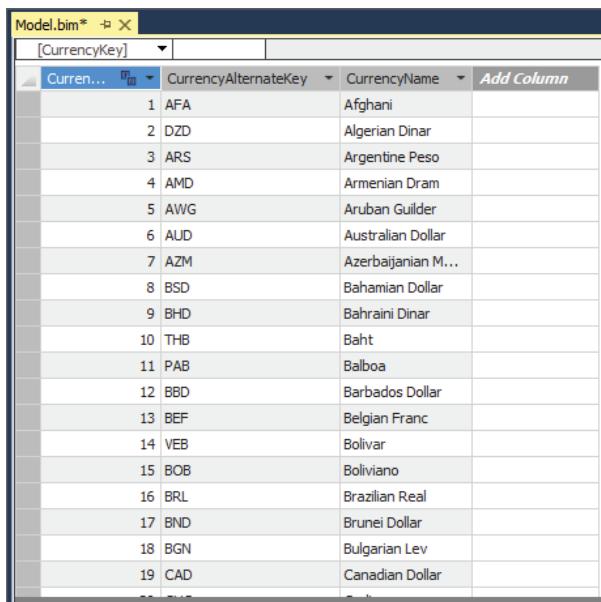
Figure 7-5: Completed Table Import Wizard

NOTE You can also rename your tables and columns to business-friendly names during this step. It is much easier to do it sooner rather than later, so take the time to do it right! Keep in mind that you can use multiple data sources, following the same steps as you did for the first data source.

Designing Relationships

Next, you must create the relationships between your data sources. If the data comes from a relational data source, the import tool brings in the foreign keys to create the relationships that already exist. If the relationship doesn't already exist, you need to create it, either within one data source or between data sources. An important aspect of your model, relationships tell the model how to relate the data elements to each other.

If you need to create a relationship that does not yet exist, you can do this by switching the model view that your designer uses. Initially, the view is in Data View mode, as shown in Figure 7-6. To switch the view, go to the Model menu, select the Model View option, and select the Diagram View option (the result is shown in Figure 7-7).



The screenshot shows a data grid titled 'Model.bim*' with a table of currency data. The columns are labeled 'CurrencyKey' (dropdown), 'Current...', 'CurrencyAlternateKey' (dropdown), 'CurrencyName' (dropdown), and 'Add Column'. The data rows are numbered 1 through 19, listing various currencies with their names in parentheses:

CurrencyKey	Current...	CurrencyAlternateKey	CurrencyName	Add Column
1	AFA		Afghani	
2	DZD		Algerian Dinar	
3	ARS		Argentine Peso	
4	AMD		Armenian Dram	
5	AWG		Aruban Guilder	
6	AUD		Australian Dollar	
7	AZN		Azerbaijanian M...	
8	BSD		Bahamian Dollar	
9	BHD		Bahraini Dinar	
10	THB		Baht	
11	PAB		Balboa	
12	BBD		Barbados Dollar	
13	BEF		Belgian Franc	
14	VEB		Bolivar	
15	BOB		Boliviano	
16	BRL		Brazilian Real	
17	BND		Brunei Dollar	
18	BGN		Bulgarian Lev	
19	CAD		Canadian Dollar	

Figure 7-6: Data View

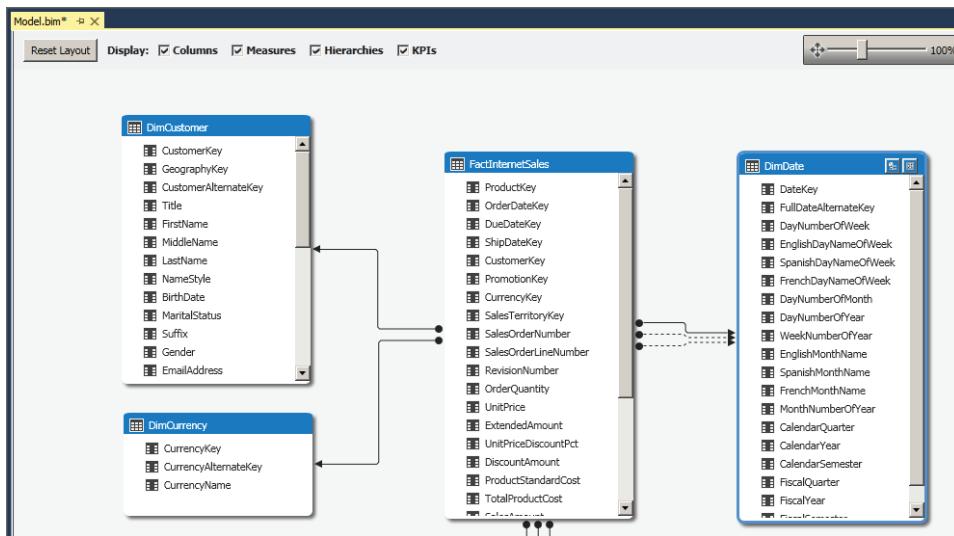


Figure 7-7: Diagram View

The Diagram View contains the tables available in the model and any existing relationships. Clicking a line between two tables highlights the columns included in the relationship. Solid lines are “Active” relationships, which mean calculations use them by default, and dashed lines are “Inactive,” so that look-ups can use them, although the inactive relationship is not used by default. To create a new relationship, click a column and drag it to the column that makes up the other side of the relationship. Luckily, the smart designer prevents you from creating a relationship that won’t work, and makes sure that one column contains only unique values.

Calculated Columns and Measures

At this stage, you’ve created a series of tables and connected them together using relationships. You now need to add any additional columns and values that help your analysis. Additional columns are called *calculated columns* and values are called *measures*. Calculated columns run for every row in the table, and measures run for the queried subset of data.

Creating Calculated Columns

You create a calculated column in your existing solution by going to the DimDate tab and clicking the far right column where it says Add column. For this example, use the column name DaysInMonth. In the formula box at the top of the screen, you

can enter the DAX formula needed to populate the column. Use the formula shown in the following line of code:

```
=DAY(EOMONTH([FullDateAlternateKey], 0))
```

The final column should look like Figure 7-8.

Year	WeekNumberOfYear	EnglishMonthName	SpanishMonthName	FrenchMonthName	MonthNumberOfYear	DaysInMonth	CalendarQuarter
182	27	July	Julio	Juillet	7	31	3
183	27	July	Julio	Juillet	7	31	3
184	28	July	Julio	Juillet	7	31	3
185	28	July	Julio	Juillet	7	31	3
186	28	July	Julio	Juillet	7	31	3
187	28	July	Julio	Juillet	7	31	3
188	28	July	Julio	Juillet	7	31	3
189	28	July	Julio	Juillet	7	31	3
190	28	July	Julio	Juillet	7	31	3
191	29	July	Julio	Juillet	7	31	3
192	29	July	Julio	Juillet	7	31	3
193	29	July	Julio	Juillet	7	31	3
194	29	July	Julio	Juillet	7	31	3
195	29	July	Julio	Juillet	7	31	3
196	29	July	Julio	Juillet	7	31	3
197	29	July	Julio	Juillet	7	31	3
198	30	July	Julio	Juillet	7	31	3
199	30	July	Julio	Juillet	7	31	3
200	30	July	Julio	Juillet	7	31	3
201	30	July	Julio	Juillet	7	31	3
202	30	July	Julio	Juillet	7	31	3
203	30	July	Julio	Juillet	7	31	3
204	30	July	Julio	Juillet	7	31	3
205	31	July	Julio	Juillet	7	31	3
206	31	July	Julio	Juillet	7	31	3
207	31	July	Julio	Juillet	7	31	3
208	31	July	Julio	Juillet	7	31	3
209	31	July	Julio	Juillet	7	31	3

Figure 7-8: Addition of a calculated column

Creating Measures

Instead of creating a calculated column, you can create a measure on a table. A measure is a value that you want to aggregate, sum, or evaluate in some fashion. Because you can do this in multiple ways, this example shows you the method that has the most flexibility.

Start by going to the Measure grid, located at the bottom of the screen. You can enter the name of the measure and the aggregation or calculation you want.

For this example, go to the FactInternetSales tab and enter the following formula to create a measure named Distinct Order Count that will provide the distinct number of orders within the tables. The formula and measure grid after entering the values should like Figure 7-9.

```
Distinct Order Count:=DISTINCTCOUNT([SalesOrderNumber])
```

NOTE Certain functions are more process-intensive than others, so use them appropriately.

fx Distinct Order Count:=DISTINCTCOUNT([SalesOrderNumber])			
SalesOrderNumber	SalesOrderLineNumber	RevisionNumber	
SO51900		1	1
SO51948		1	1
SO52043		1	1
SO52045		1	1
SO52094		1	1
SO52175		1	1
SO52190		1	1
SO52232		1	1
SO52234		1	1
SO52245		1	1
SO52301		1	1
SO52314		1	1
SO52342		1	1
SO52387		1	1
SO52499		1	1
SO52500		1	1
SO52545		1	1
SO52593		1	1
SO52627		1	1
SO52637		1	1
Distinct Order Count: 27659			

Figure 7-9: Formula and measure grid

At this stage, you have created your initial model. You've paid attention to the size of your model, brought in the desired information, and linked it together in a usable fashion. You could deploy the model at this time and let people use it. Or, you could add some additional enhancements to make it even easier for the end user, as you will read about next.

How Do You Enhance the Model?

You can enhance the tabular model in many ways, including making it easier for end users to consume data, adding elements that allow queries to perform well, and employing methods to allow the model to scale as needed for both the underlying data and the number of users accessing the information.

Adding Hierarchies

To analyze data and make the analysis easier for business users, you can group it in a way that allows you to drill up and drill down while reducing the number of total elements you need to look through. A good example of this includes food items, as shown in Figure 7-10. The top level of the hierarchy contains all food;

the next level has the food groups (fruits, vegetables, grains, proteins, and dairy); and the lowest level (leaf level) contains each item within each food group. For example, if you want to find all the apples, you can look under the Fruits item and search through only 4 items, instead of having to search through all 17 leaf-level items if no hierarchy were to exist.

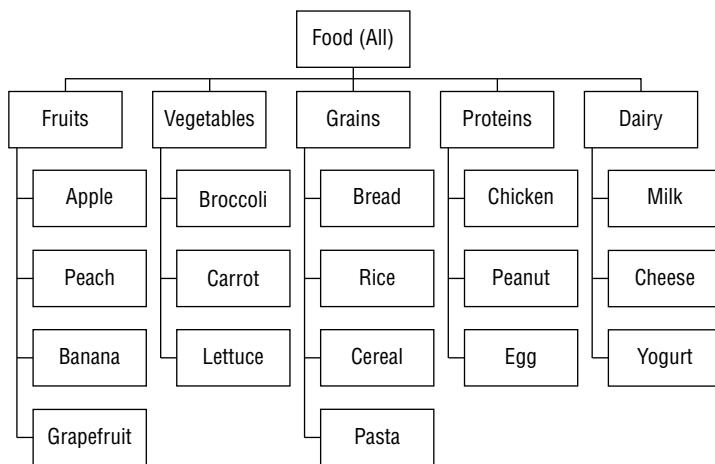


Figure 7-10: Example hierarchy

Using the same method, you can set up hierarchies in the tabular model. For this example, you will create a hierarchy on the Product table for Product Subcategory and Product Category. To start, you must include the desired columns in one table. Do this:

- 1. Add columns to the table.** Add the EnglishProductSubcategoryName and EnglishProductCategoryName to the DimProduct table.
- 2. Using Data View, add two new calculated columns.** Use the following code (line 1 is the ProductSubcategory column and line 2 is the ProductCategory column):


```
=RELATED( (DimProductSubcategory [EnglishProductSubcategoryName] ) )
=RELATED( (DimProductCategory [EnglishProductCategoryName] ) )
```
- 3. Create your hierarchy.** You do this by going to Diagram View option from the Model ▾ Model View menu.
- 4. Right-click the DimProduct table, and select the Create Hierarchy option.** This is shown in Figure 7-11.
- 5. Rename the hierarchy (in this case to Category Hierarchy).** Then you're ready to create the hierarchy levels.

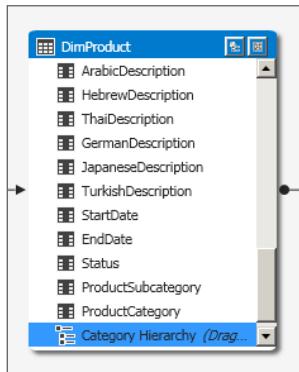


Figure 7-11: Hierarchy creation in Diagram View

6. **Drag each of the columns that should participate in the hierarchy from within the DimProduct table to the top of the hierarchy you just created.** Start with ProductCategory, then ProductSubcategory, and finally EnglishProductName.
7. **Remove those columns (see step 6) from client tools.** You want business users to look in the hierarchy rather than spend too much time searching through the regular fields. Right-click each of the columns that you had previously pulled into the hierarchy, and select the Hide from Client Tools option. This option turns the columns gray. The final hierarchy and hidden columns are shown in Figure 7-12.

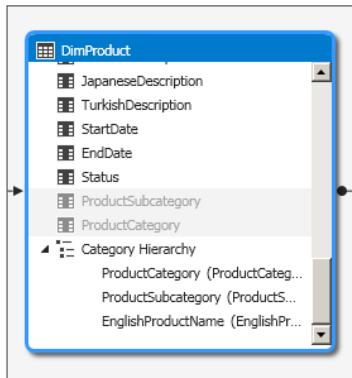


Figure 7-12: Product Category hierarchy

The model gives no indication if you have created a natural hierarchy (where the top-level item encapsulates the most descendant items, and all the way down) or an unnatural hierarchy (where the levels could have no relation to one another). You will see some performance benefit to using a natural hierarchy, but depending on your scenario, you could also see some usability benefit to using an unnatural hierarchy because the user can drill through each level as a form of filtering the data. Just be aware of which one you use and the benefit you get from that type.

Designing Perspectives

Upon initiation of a Business Intelligence project, users (and developers!) often get excited about the amount of data available to include in the model for analysis. Unfortunately, they can also be overwhelmed by that same amount of data, between data for different departments and data elements not relevant to the users' typical analyses. To help combat the overwhelming data problem and increase the model's usability for business users, you can limit what the end user sees, using a perspective.

A *perspective*, an object in a tabular model, limits the information shown through the client tool. Essentially, you connect to a perspective as though connecting to a cube, and you only see the filtered elements. When using the perspective, you don't even realize the full model contained additional columns. The objects you can filter include both the fields direct from the data source and any calculated columns you created.

In this example, you create a perspective in the AdventureWorks model:

1. **Navigate to the Model menu.** Select the Perspectives option, and select the Create and Manage option.
2. **Add a new column to the screen.** You do this by clicking the New Perspective button to add a new column to the screen, and name it Marketing.
3. **Check the boxes next to each field you want shown, or check the top box to select all and uncheck the boxes you do not want shown.** In this scenario, select all, and then deselect DimCurrency, DimCustomer, and DimSalesTerritory. After creating the new perspective, the Perspectives window should look like Figure 7-13.

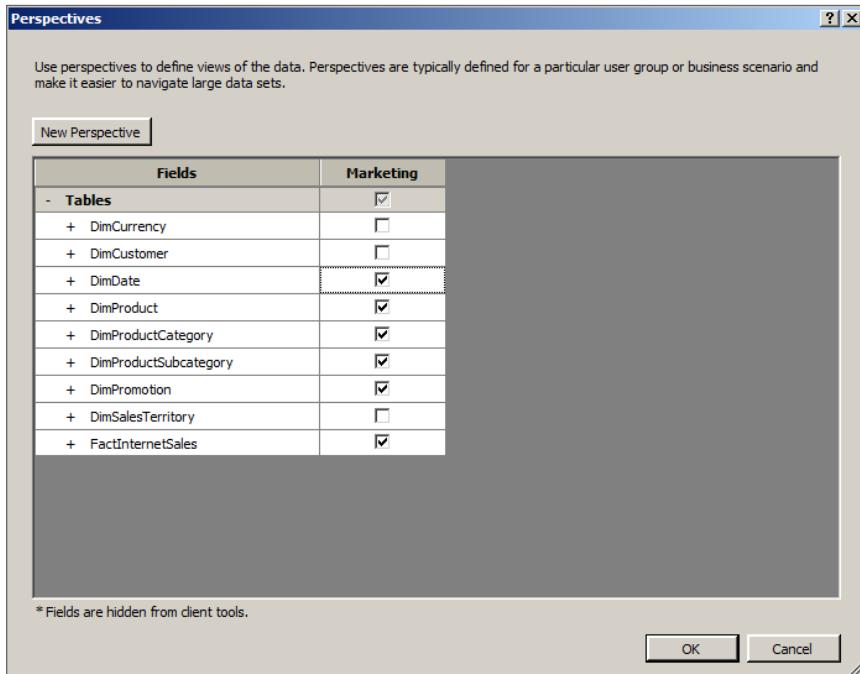


Figure 7-13: Perspectives window

Keep in mind that perspectives do not provide security, so use them only for their intended purpose: better usability for end users.

Adding Partitions

Depending on your organization's situation, you may also need your model to load its data quickly. Some reasons for fast processing times include the need for close-to-real-time data and data source connectivity issues. If any of these are the case, you may want to consider partitioning your model to speed up the processing time.

Table partitions split a table into logical chunks to process separately. To set a partition, you filter the rows it contains. A typical partitioning paradigm is to use the date column, so that you will need to reprocess only the most recent partitions. An added benefit of the tabular model is that you can also partition the descriptor (or dimension) tables; this comes in handy when you have a very large dimension, such as products or customers.

For the AdventureWorks model, you will create a partition on the Internet Sales table:

1. **Navigate to the Table menu.** Then select the Partitions option.
2. **Change the Table drop-down list to Internet Sales.** At this stage, the Partition Manager window should like Figure 7-14.

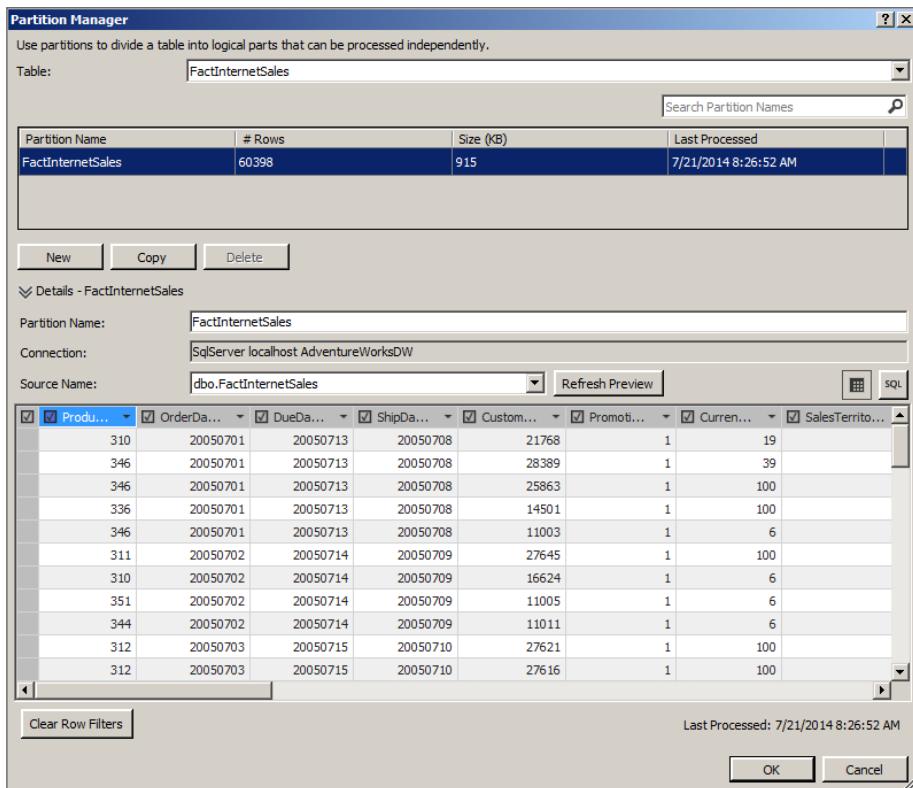


Figure 7-14: Partition Manager with one partition

3. **Create a new partition.** Click the New button below the partition section and rename the partition FactInternetSales 2005 in the Details section.
4. **With this partition selected, click the SQL button on the top right of the table area in the Details section.**
5. **Change the SQL statement to the following T-SQL statement:**

```
SELECT [dbo].[FactInternetSales].*
FROM [dbo].[FactInternetSales]
WHERE OrderDateKey BETWEEN 20050101 AND 20051231
```

6. **Be sure to change the original partition to contain the rest of the data.** You do this by changing the name to FactInternetSales Current, and using the SQL statement for the partition as shown in the following code:

```
SELECT [dbo].[FactInternetSales].*
FROM [dbo].[FactInternetSales]
WHERE OrderDateKey >= 20060101
```

7. Click OK.
8. **Process the partitions.** Go to the Model menu, select the Process option, and then select the Process Partitions option. Check the boxes next to both of the partitions you just created, and click OK. Once the processing has completed, you will see the screen shown in Figure 7-15.

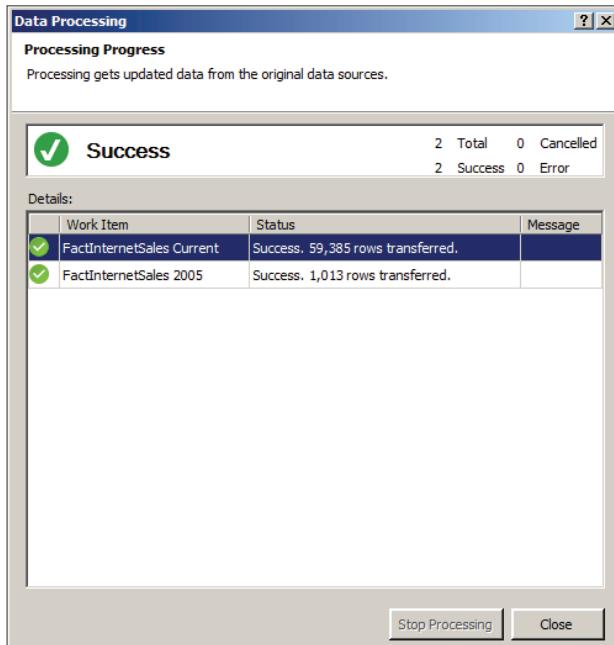


Figure 7-15: Process partitions window

9. **Once you see the Success message on the Data Processing window, go back to the partitions menu.** Here you'll see the number of processed rows and size based on your queries. This should look like Figure 7-16.

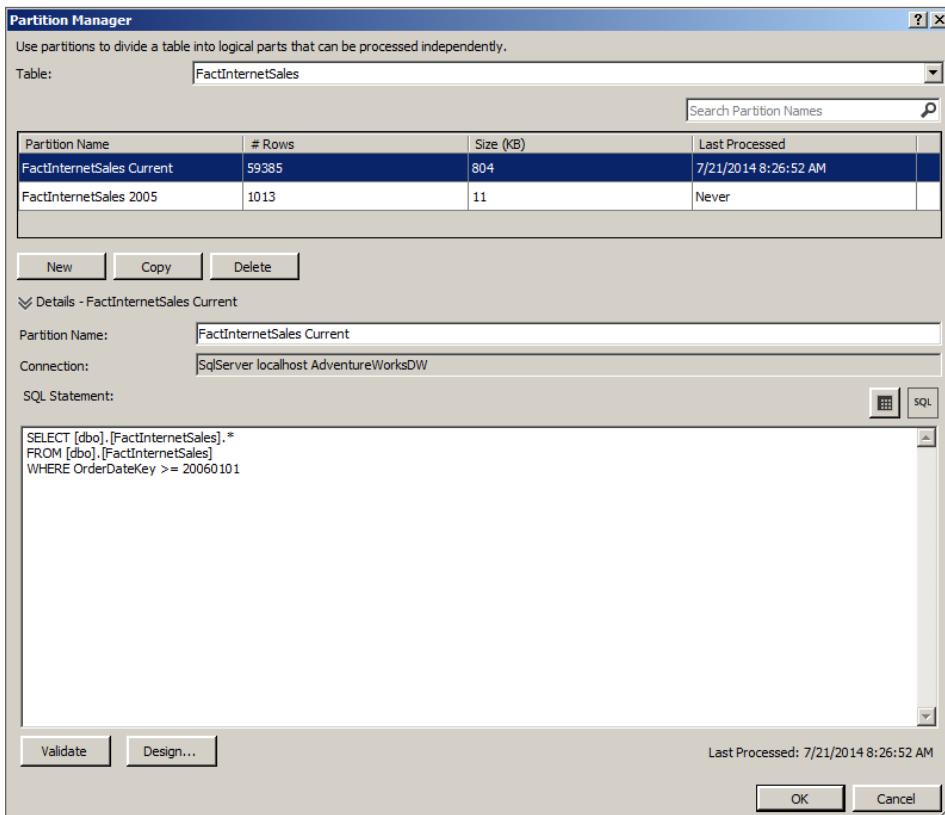


Figure 7-16: Partition Manager with two processed partitions

How Do You Tune the Model?

So far in this chapter, you have designed an optimal model and enhanced it to add additional flexibility and scalability. But what do you do if it still isn't performing the way you would expect? The two main times when a performance issue could occur are in the processing of the cube and the querying, either through a client tool or directly against the cube. This section looks at both issues and some possible solutions for each.

Optimizing Processing

Your tabular model's first potential issue is slow processing. As previously discussed, processing occurs when the tabular model connects to the data source and pulls information into a database. The model also takes this time to validate relationships and perform any calculations on calculated columns.

Processing Modes

You should understand the different processing modes and query types to fully understand what could be causing problems. You can process each object (either a partition, a table, or the entire database) in a variety of modes, which will load and unload the data in different ways. Some of the more common processing modes and their purposes include:

- **Process Default:** Loads data for any unprocessed or incomplete partitions and performs any associated calculations
- **Process Full:** Reloads data and performs associated calculations whether the object is processed or not
- **Process Data:** Reloads data into the object without touching calculations
- **Process Clear:** Wipes the data and calculations in the object
- **Process Add:** Keeps existing data within an object and includes additional data into the object

Keep in mind that if you change the model structure, you will need to run a Process Full. However, if you always run a Process Full, you may realize some performance benefit by modifying your processing mode.

For a great reference on processing in tabular models, see Microsoft ‘employee’ Cathy Dumas’s blog post here: <http://blogs.msdn.com/b/cathyk/archive/2011/09/26/processing-tabular-models-101.aspx>.

Storage Modes

As previously discussed, tabular models provide the ability to store all the data in-memory, resulting in fast querying. However, in certain scenarios, it may make more sense to not store all data, and directly query the underlying data source instead. You can still do this through the tabular interface, with a few caveats. The list below describes each storage mode available to you:

- **DirectQuery:** Any DAX queries are converted to T-SQL on-the-fly and executed directly against the SQL Server data source instead of using stored data. This works only for SQL Server data sources, and prevents the usage of calculated columns, in addition to other restrictions.
- **InMemory:** All data is stored in-memory and DAX queries retrieve information from that location. Data is as fresh as the last time the data was processed.
- **InMemoryWithDirectQuery:** Hybrid option, where queries should act like the data is stored in-memory, unless the client sends in a request to use the DirectQuery option. The data model obeys DirectQuery restrictions.

- **DirectQueryWithInMemory:** Hybrid option, where queries should act like the data is accessible only through the SQL Server database, unless the client sends in a request to use the InMemory option. The data model obeys DirectQuery restrictions.

Optimization

To optimize your model, you must understand the processing process, as shown in Figure 7-17. The process starts at the left of the diagram and moves its way to the right. When a client tool sends a process request to the Analysis Services database, the database uses the processing and storage modes to determine its next steps. The next steps could either be connecting to the database to pull a full set of data, handling the process internally in the database to update the calculations, or both. This encoding step converts the information provided by the source or engine into an index for the database to use. Finally, the engine compresses and stores the information in the columnar format for later querying.

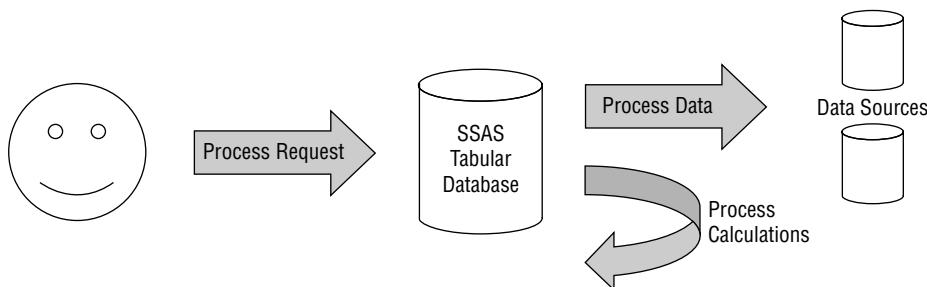


Figure 7-17: Processing process

Look at how long each step takes by using Profiler and separating out each of the pieces that could cause a slowdown. Then you can make a decision on how to best fix the issue. Some of the possible options include:

- **Determine the storage mode of the partition.** If the storage mode is a variation of InMemory, would DirectQuery be a better option?
- **Determine the processing mode of the request.** If the storage mode includes processing the data, would a Process Default or Process Add be a better option?
- **Run the query that pulls data from the partition against the data.** Can you optimize the query to retrieve data from the source more quickly?
- **Evaluate the data types used to store information in the model.** Change decimals with less than four decimal places to “Currency” and use numeric data types whenever possible.

- **Assess the time needed for the calculations in the model.** Can you rewrite the DAX formula for faster querying or move some of the calculation logic to the data movement prior to processing?
- **Consider the amount of data imported into the model.** If you modify the processing to run in parallel, will that decrease the overall processing time? Would partitioning the data also assist in decreasing the time? Is all data being used or can you remove unused columns or archive historical data?

The above listing just scratches the surface of performance problems that could occur when processing the tabular model. For more information, see Books Online or Microsoft's performance tuning white paper.

NOTE Microsoft authored an excellent article on performance tuning of tabular models in Analysis Services 2012. Although you may be using a more recent version of the product, the content is most likely still applicable. You can download the white paper here: <http://msdn.microsoft.com/en-us/library/dn393915.aspx>.

Optimizing Querying

Query performance tuning often comes to the developer's attention as a problem that must be fixed immediately, because business users come knocking when they can't get their data quickly! You must understand the query request and response in the Analysis Services tabular model to understand the location of the issue and how to fix it. Figure 7-18 shows the architecture used by the query engine.

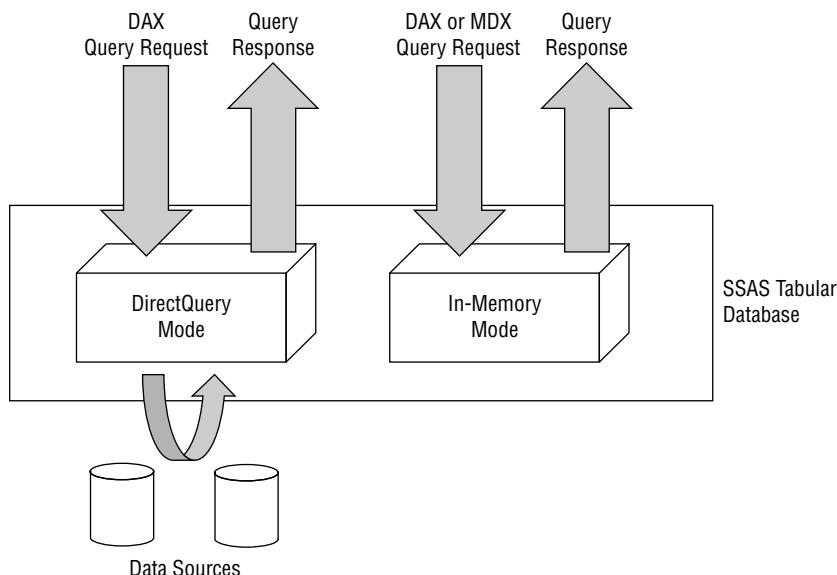


Figure 7-18: Query engine architecture

Note that the tabular engine can take requests in both the MDX and DAX languages. The formula engine will return information that only requires a calculation, whereas the storage engine will return results for queries that require further data exploration.

To troubleshoot any performance issues in a query, you need to investigate where the slowdown occurs. You can do so by:

- 1. Running the query.** This ensures that you see the same performance issue.
- 2. If you do see the same issue, drill into each piece of the query, calculation by calculation, to see the cause of the problem.** If you remove the calculations, does the problem still occur?
- 3. Once you have simplified the query as much as possible, if you still see the issue, you can determine what part of the engine is returning the information.** You can do this using SQL Server Profiler to run a trace against the tabular instance. An example of the types of messages returned is shown in Figure 7-19. An EventClass that starts with “Vertipaq SE Query” and spends more than 50 percent of the total query time in this stage has an issue with the storage engine. Otherwise, the issue is in the formula engine.

EventClass	EventSubclass	TextData	ConnectionID
Calculation Evaluation	7 - RunEvalNode Start	<EvaluationNode> <NodeIndex>1</Node...	50
Vertipaq SE Query Begin	0 - Vertipaq Scan	SET DC_KIND="AUTO"; SELECT [DimProd...	50
Vertipaq SE Query Begin	10 - Vertipaq Scan internal	SET DC_KIND="DENSE"; SELECT [DimPro...	50
Vertipaq SE Query End	10 - Vertipaq Scan internal	SET DC_KIND="DENSE"; SELECT [DimPro...	50
Vertipaq SE Query End	0 - Vertipaq Scan	SET DC_KIND="AUTO"; SELECT [DimProd...	50
Query Dimension	2 - Non-cache data	00000000000000000000000000000000...	50
Calculation Evaluation Detailed Information	107 - RunEvalNode Finished Calculating Item	<EvaluationNodeItem> <NodeIndex>1<...	50
Calculation Evaluation	8 - RunEvalNode End	<EvaluationNode> <NodeIndex>1</Node...	50
calculation Evaluation	7 - RunEvalNode Start	<EvaluationNode> <NodeIndex>3</Node...	50
Vertipaq SE Query Begin	0 - Vertipaq Scan	SET DC_KIND="AUTO"; SELECT [DimProd...	50
Vertipaq SE Query Cache Match	0 - Vertipaq Cache exact match	SET DC_KIND="AUTO"; SELECT [DimPr...	50
Vertipaq SE Query End	0 - Vertipaq Scan	SET DC_KIND="AUTO"; SELECT [DimProd...	50
Query Dimension	2 - Non-cache data	00000000000000000000000000000010	50
Calculation Evaluation Detailed Information	107 - RunEvalNode Finished Calculating Item	<EvaluationNodeItem> <NodeIndex>3<...	50
Calculation Evaluation	8 - RunEvalNode End	<EvaluationNode> <NodeIndex>3</Node...	50
Calculate Non Empty Current	1 - Get Data		50
Get Data From Cache	2 - Get data from flat cache	Global Scope	50
Calculate Non Empty Current	1 - Get Data		50
Vertipaq SE Query Begin	0 - Vertipaq Scan	SET DC_KIND="AUTO"; SELECT [DimProd...	50
Vertipaq SE Query Cache Match	0 - Vertipaq Cache exact match	SET DC_KIND="AUTO"; SELECT [DimPr...	50
Vertipaq SE Query End	0 - Vertipaq Scan	SET DC_KIND="AUTO"; SELECT [DimProd...	50
Query Dimension	2 - Non-cache data	00000000000000000000000000000000...	50
Calculate Non Emty Current	1 - Get Data		50

Figure 7-19: Profiler example

You have two optimization techniques available to you when using a tabular model:

- **Use a different calculation:** Your goal is to use the formula engine if possible, and if not, use as little memory as possible.
- **Modify the underlying model.** You do this by moving some of the calculations from the query to the model, reducing the size of the model, or combining tables to reduce the number of joins the query engine has to make.

By utilizing these optimization techniques, you can ensure that your model will both process and return query results quickly.

Summary

The tabular model in Analysis provides a powerful interface for business users while being very easy for developers to code. After reading this chapter, you should understand the benefits of the tabular model, but know that this model type is not for every scenario. You now know how to create the model and enhance it to provide the best possible solution. Finally, you know how to troubleshoot any additional issues that may occur after you have completed the design.

The next chapter introduces you to the original model type, a multidimensional model, providing a similar view into the benefits and the creation process.

Developing a Flexible and Scalable Multidimensional Model

The tried-and-true semantic layer, Analysis Services' multidimensional model, offers a great interface to allow end users to see their data through an easy-to-use interface. As the engine has improved over the years, developers have found tips and tricks to optimize the model as much as possible. Designing the model correctly from the beginning will make your life easier as a developer and an end user. In addition, you can enhance and performance-tune the model if needed. After you have completed all your development, your model will satisfy all your business users' needs.

This chapter introduces the multidimensional model as part of the Analysis Services suite, discusses how to design the model optimally, and how to enhance it to ensure flexibility and scalability. Specifically, you will learn how to walk through the Cube Creation Wizard, modify measures and dimensions, and use the Business Intelligence Wizard. If there are any additional concerns with the cube, you will learn how to performance-tune both processing and querying to create the best possible model for your users.

Why Use a Multidimensional Model?

In addition to the tabular model, the Analysis Services product also includes the multidimensional model. As the original analysis portion of Analysis Services, the multidimensional model is a very mature product. (Be sure to read Chapter 5,

which explains the history of Analysis Services and why you would use one semantic model over another.) Although similar to other Online Analytical Processing (OLAP) cubes, the multidimensional model is the Microsoft version, and you should understand both it and its querying language, MDX (multidimensional expressions), to learn how to design a good model.

Understanding the Multidimensional Model

A multidimensional model is the technology between a data warehouse and a reporting/end-user tool. As a semantic layer, it provides a business “cube” to store business logic and allows end users to easily access the information. Additionally, it stores pre-aggregated information to make queries faster than running them on a traditional RDBMS.

MULTIDIMENSIONAL MODEL

A multidimensional model is a semantic model that stores its information in a pre-aggregated format to provide an easy-to-use tool for fast querying.

In contrast to the tabular model, the multidimensional model must sit on top of a dimensionally modeled data warehouse. Although the dimensional model may be physical or virtually created, the multidimensional model understands only facts and dimensions, as in a star schema dimensional model. Figure 8-1 shows a sample star schema.

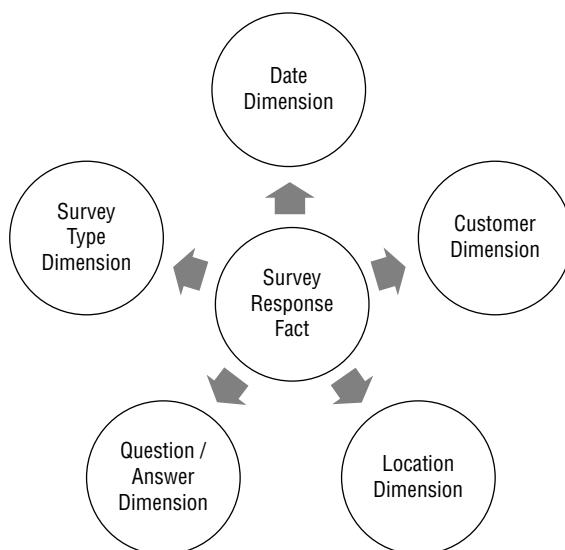


Figure 8-1: Star schema

After the dimensional model has been put into the multidimensional model, an end user will access it using a client tool. A common client tool, Excel, easily exposes the multidimensional model using a pivot table, where the dimensions are rows, columns, or filters, and the facts are the values within the pivot table.

Understanding the Multidimensional Model Process

When working with a multidimensional model, you have a few steps to perform, both before and after designing the model. The diagram shown in Figure 8-2 shows these high-level steps: Data Preparation, Data Modeling, Data Load, and Data Consumption.

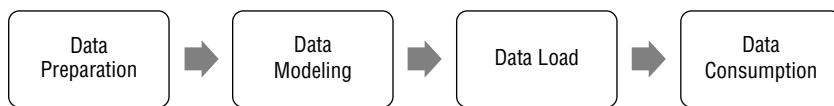


Figure 8-2: Multidimensional model process

In the Data Preparation phase, you bring your information from a variety of sources into a dimensional, star-schema model, as shown in Figure 8-1. Do not underestimate this huge task of data preparation! Give yourself plenty of time to correctly design and populate the underlying data warehouse. Next, you will design the multidimensional model, which is the bulk of what this chapter covers. After you deploy the model to a server, the model becomes instantiated as a multidimensional database. You can then load the database with data from the data warehouse, and end users can consume the data.

How Do You Design the Model?

Designing the multidimensional model in an optimized way is the first step in having a scalable and flexible model. The steps to design the model include accessing the data and creating measures and dimensions. This section covers how best to approach the model design before continuing on to enhance the model.

Creating Data Sources and the Data Source View

You start the creation of your model in SQL Server Data Tools Business Intelligence (SSDT-BI) using the Business Intelligence project templates. To create a new project and solution, follow these steps:

1. Click File \Rightarrow New Project.
2. Select the Analysis Services Multidimensional and Data Mining Project. This is shown in Figure 8-3. For this example, name the project **AdventureWorksMD** and click OK.

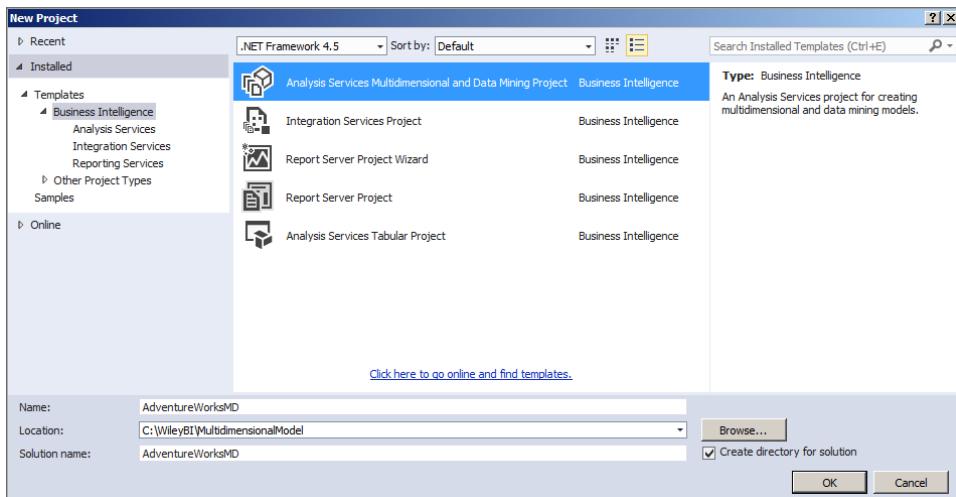


Figure 8-3: Creating a new multidimensional model solution

After creating the project, you see the available objects in the Solution Explorer on the right side of the development environment. Figure 8-4 displays the full list of objects in the Solution Explorer. The top two folders, Data Sources and Data Source Views, are explained in this section.

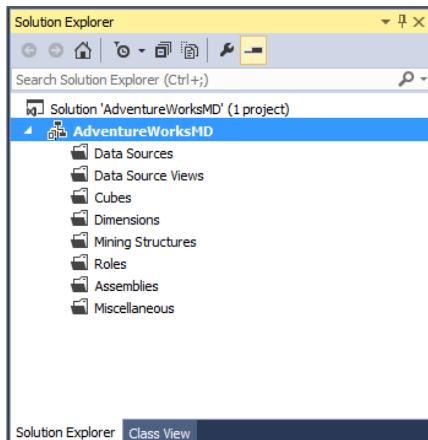


Figure 8-4: Solution Explorer

3. **Bring in the data necessary for the model.** The data source types that can provide data to the multidimensional model include Oracle, SQL Server, Analysis Services, and more. Ideally, the underlying data store has been modeled in a dimensional star-schema format. In fact, the closer the

underlying model is to a dimensional model, with facts and dimensions, the easier it will be to bring in the data.

DIMENSIONAL MODEL

A dimensional model stores data in de-normalized, business-friendly, reporting tables. Often called a star schema or snowflake schema, a dimensional model is the ideal data source for a multidimensional model.

4. **Create a data source.** Right-clicking the Data Source folder and selecting the Add New Data Source option. Select the type and enter the connection string for the data source. In this example, select the AdventureWorksDW database on your localhost server. Verify your connection, and then you are ready to create the data source view.
5. **Create the data source view.** Right-click the Data Source View folder and select the Add New Data Source View option. A wizard appears that walks you through the creation. Start by selecting tables you need. An Add Related Tables button appears at the bottom of the import data screen to add additional tables. In this scenario, select the FactInternetSales table and click the Add Related Tables button. Also add the DimProductCategory and DimProductSubcategory tables to assist in creating a hierarchy later. A view of the selected tables is shown in Figure 8-5.

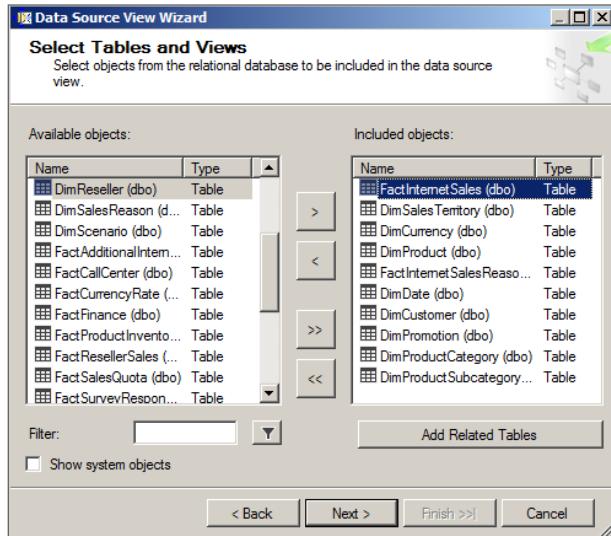


Figure 8-5: Selected tables

If the underlying data source contained relationships between the tables, the wizard imports the relationships into the data source view. If a relationship did not exist, you can add the relationship by dragging one column to another and defining the source and target for the relationship.

Additionally, you can convert any of the tables to a named query to restrict the columns returned, add additional calculations, or join multiple tables together.

Using the Cube Creation Wizard

You can accomplish the next step in the multidimensional model creation process, adding the querying objects, through the Cube Creation Wizard. The wizard takes you through the steps of selecting the appropriate rows and columns from the data source view to create measure groups, measures, and initial dimensions.

The first object, a measure group, corresponds to a fact table in the dimensional model. To start the process of creating the measure groups, follow these steps:

1. **Right-click the Cube folder, and select the New Cube option.**
2. **Select the option named Use existing tables.** This allows you to select the tables that you just imported using the Data Source View Wizard.

MEASURE GROUP

A *measure group* contains a series of values that can be aggregated in some fashion, such as summation, average, minimum, or maximum.

3. **Select the tables you want to include.** You can either click the Suggest button to check the boxes next to the tables that the wizard determines as appropriate measure group tables, or you can manually select the tables as measure groups. In data warehouses, the tables most commonly used for measure groups are fact tables. For the AdventureWorks example, make sure that FactInternetSales and FactInternetSalesReason are both checked, as shown in Figure 8-6.

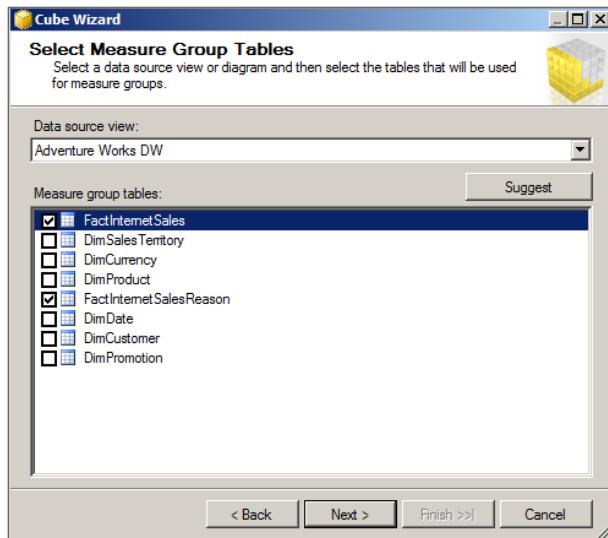


Figure 8-6: Select Measure Group Tables screen

MEASURE

A **measure**, which exists inside of a measure group, contains the value that the database will aggregate in some fashion, such as summation, average, minimum, or maximum.

4. **Select the measures for each measure group.** The designer populates the wizard screen with all numeric columns from the table assigned to the measure group. You can uncheck any values that should not be aggregated, such as order numbers or line numbers. For the AdventureWorks example, uncheck the Revision Number column. Also, keep in mind that the designer automatically adds an additional measure for each measure group to count all rows in the measure group. For the AdventureWorks example, uncheck the Fact Internet Sales Count row, but keep the Fact Internet Sales Reason Count row. The final view of the Select Measures screen should like Figure 8-7.

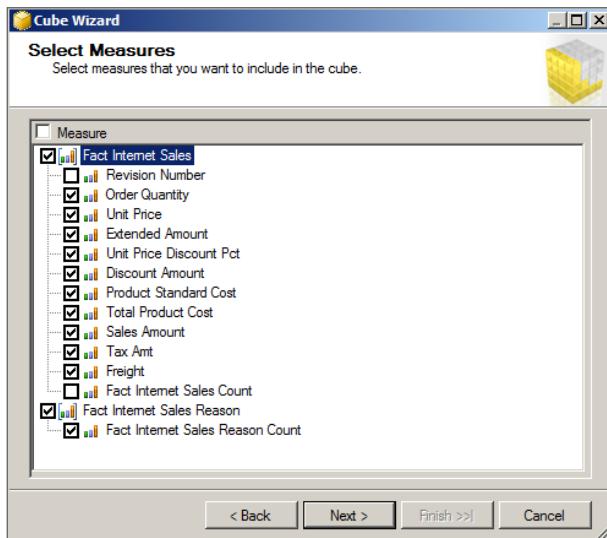


Figure 8-7: Select Measures screen

5. **Rename any of the objects in this screen to align more closely to your business terms.** You can do so by double-clicking the name and typing the correct name.
6. **Select the tables that you want to become cube dimensions.** The wizard determines what tables have properties that could work as dimensions, which means they have the descriptive properties of an entity. In the AdventureWorks scenario, leave all the dimensions selected. Finally, rename your dimensions, because the names listed here turn into identifiers after the wizard completes and cannot be changed. The final screen is shown in Figure 8-8.

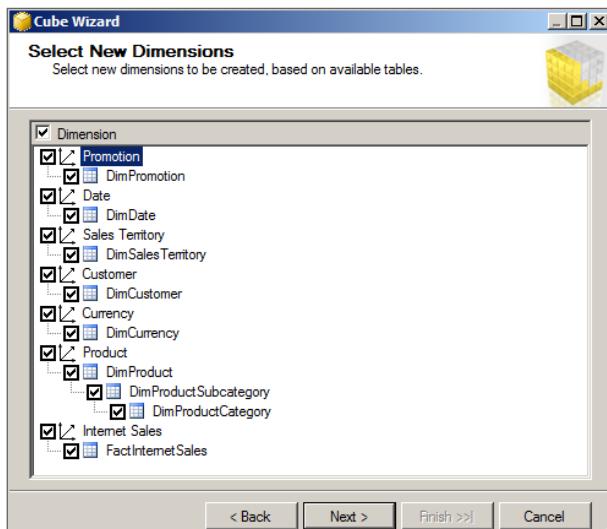


Figure 8-8: Select New Dimensions screen

At this point, you have created the initial shell of the cube. Next, you will finish setting up the measures and dimensions outside of the wizard interface.

Adjusting Measures

The Cube Creation Wizard completed most of the work to create the measures, but you can make a few changes to ensure each measure acts as expected. For this section, you focus on the Measures and Properties panes on the Cube Structure tab. After following the previously described steps, the Measures pane will look like Figure 8-9.

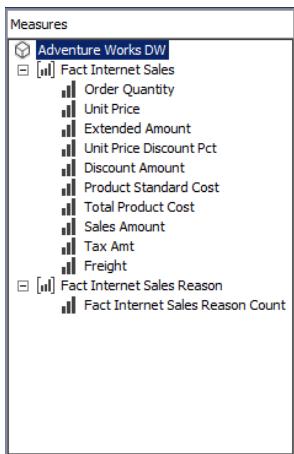


Figure 8-9: Measures pane

MEASURE GROUPS

Although you kept a *measure group* per fact table in this example, this is not always an ideal situation. If you have a measure that uses a DistinctCount aggregation function, separate that measure out to a different measure group for best performance.

To complete the creation of the measures, you need to adjust properties for each measure. The Properties pane is typically found in the bottom right corner of the SQL Server Data Tools designer. The most commonly used properties include:

- **AggregateFunction:** Describes how the measure value will roll up when grouped by attributes in the dimensions. The functions include, but are not limited to, sum, average, minimum, lastnonempty, and more.
- **DisplayFolder:** Groups measures into a folder that a client tool connecting to the Analysis Services cube can use. The free-form field allows you to enter any text that you want. Just use the same name for multiple measures to have them grouped together.

- **FormatString:** Provides a suggested format for a client tool to use for the measure. The value for this property can be a provided value, such as Standard, Currency, or Percent, or a format string comprised of pound signs (#), commas (,), and other symbols.

For this example, you set a few of the measures to highlight these properties:

- **For Unit Price Discount Pct:** Set the AggregateFunction to AverageOfChildren. You do this because adding up percentages results in a large number that has no business purpose.
- **For Product Standard Cost and Total Product Code:** Set the Display Folder to Product Measures. You do this to group the product-specific measure together and reduce the amount of measures that an end user has to see.
- **For Extended Amount, Discount Amount, and Sales Amount:** Set the FormatString to Currency.

You can use these measures later when querying the cube.

Completing Dimensions

The final step in completing the initial design of the model is to complete the dimensions. The cube actually contains two types of dimensions: database dimensions and cube dimensions. For “role-playing” dimensions that play multiple roles within the cube, you would have one database dimension and multiple cube dimensions. For example, a date dimension would exist once as a database dimension, and the cube dimensions that use the date dimension’s configuration include an order date dimension, shipping date dimension, and due date dimension.

DIMENSION TYPES

A *database dimension* lives once inside of an Analysis Services database and is where you will make most of your configuration changes.

A *cube dimension* is an instantiation of a database dimension within the cube.

All the base database dimensions that you imported through the Cube Creation Wizard appear in the Solution Explorer, under the Dimensions folder. The AdventureWorks example’s list of dimensions is shown in Figure 8-10. To change the properties for a database dimension, double-click the entry to open it. For this example, open the Product.dim dimension.

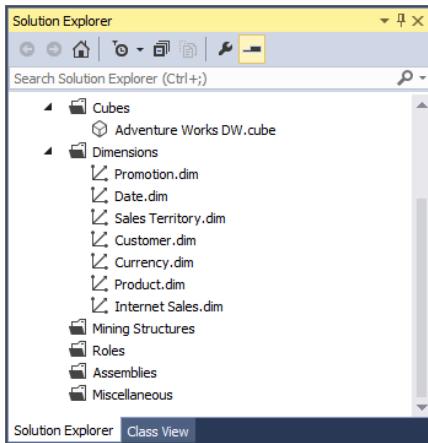


Figure 8-10: Database dimension list

The Cube Creation Wizard creates the Product dimension with three attributes named Product Category Key, Product Key, and Product Subcategory Key. You now add attributes to allow the end user to query on those attributes by dragging the desired fields from the Data Source View pane to the Attributes pane. For the AdventureWorks example, add the following attributes for the Product dimension: ProductAlternateKey, EnglishProductName, Color, Size, Weight, Class, EnglishDescription, and Status. From DimProductSubcategory, pull ProductSubcategoryAlternateKey and EnglishProductSubcategoryName. And from DimProductCategory, pull over ProductCategoryAlternateKey and EnglishProductCategoryName. The final list of attributes is shown in Figure 8-11.

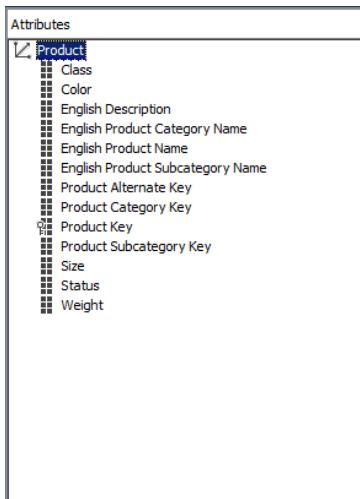


Figure 8-11: Product dimension attributes

Next, you define the properties for each of the attributes in the dimension. Some of the attributes to consider are:

- **AttributeHierarchyDisplayFolder:** Displays the attribute within a folder in the client application. This property is a great way to group similar attributes to make it easier for the end user to navigate to the appropriate attribute.
- **AttributeHierarchyVisible:** Hides the attribute from the end user. Use this property to hide the surrogate key, any attributes included in a hierarchy, and any attributes used in a calculation that don't make sense on their own.
- **DiscretizationMethod:** Groups the attribute values into queryable groups, based on an equal division or a cluster. This is a great option if the attribute contains numbers, such as inventory levels or age. This can be combined with the DiscretizationBucketCount property to decide how many groups should be created.
- **IsAggregatable:** Tells the model what values to aggregate. Set this property to False if you never want to roll up measures based on this attribute.

For the Product dimension example, set the Product Key's AttributeHierarchyVisible property to False and keep the other attributes' properties at the default. Next, you are ready to enhance the model that you just set up.

How Do You Enhance the Model?

The first step in creating a flexible and scalable multidimensional model is designing it to perform that way. The next step is enhancing the model to have it perform even better. Enhancing the multidimensional model includes additional querying functionality, better usability for end users, and faster performance in both the processing of the model initially and accessing it through a client tool. The following feature descriptions include methods you can use for these purposes.

Adding Navigation with Hierarchies

Hierarchies for dimensions provide navigation for end users and also help Analysis Services create appropriate aggregations to make the cube perform faster. You can build multiple hierarchies on a dimension, based on the attributes' values and desired functionality. See Chapter 7 for a more detailed and pictorial description of a hierarchy.

Similar to the hierarchy created in Chapter 7's tabular model, you can create a hierarchy in the product dimension in the multidimensional model. To create a hierarchy in the product dimension, go to the cube dimension screen that you

used to change the dimension attributes' properties. You create the desired hierarchies for this dimension in the center pane labeled Hierarchies.

For this example, you create a date category hierarchy that looks like Figure 8-12. It will contain the Calendar Year at the top level, the Calendar Quarter below it, and the lowest level will contain the English Month Name. Rename the hierarchy to **Calendar Hierarchy**, so that it is easy for users to understand the relationship.

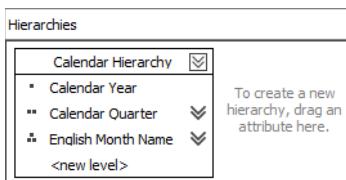


Figure 8-12: Date Calendar Hierarchy

It is very important that a value of an attribute relates to only one value of the attribute at the higher level. In this scenario, the Calendar Quarter attribute contains the values of 1, 2, 3, and 4. Because the year is not included, the optimizer doesn't know how to roll up the dates beyond the quarter. To fix this, add the Calendar Year to the key columns of the Calendar Quarter attribute, as shown in Figure 8-13. Similarly, add the Calendar Year attribute to the key columns of the English Month Name attribute.

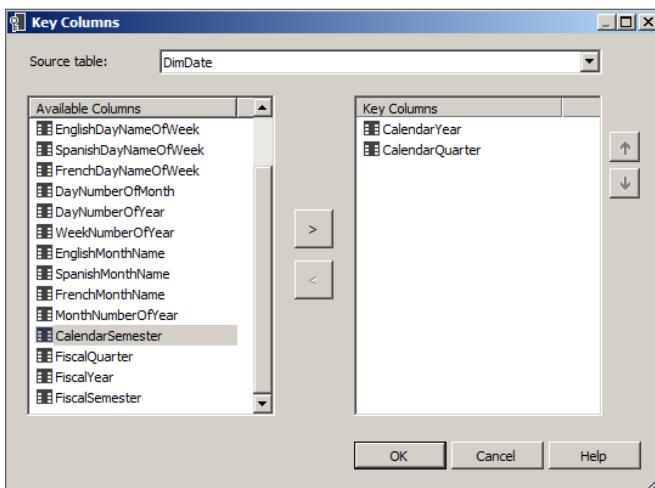


Figure 8-13: Calendar Quarter key columns

If you receive the error message similar to "Errors in the OLAP storage engine: A duplicate attribute key has been found when processing: Table: 'dbo_DimDate', Column: 'CalendarQuarter', Value: '4'. The attribute is 'Calendar Quarter,'" the

values in a lower level do not have one unique value that they roll up to. If your data is accurate, you may need to include multiple columns in the attribute to ensure uniqueness, as you did for the Calendar Quarter attribute.

The important step to ensure your hierarchy is optimized correctly is to set up the attribute relationships for the hierarchy. Go to the second tab on the Date dimension, named Attribute Relationships, which looks like Figure 8-14. An attribute relationship defines how the attributes within a hierarchy relate to each other. For the Calendar hierarchy, the final view looks like Figure 8-14.

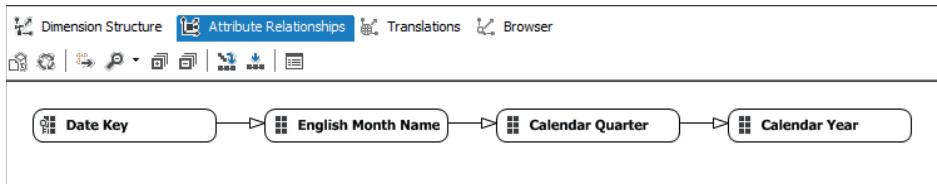


Figure 8-14: Calendar attribute relationships

Using the Business Intelligence Wizard for Calculations

One of the great powers of Analysis Services is the ability to include business logic within the semantic layer that might otherwise have to be built within the client tool. The cube contains calculations that are pieces of MDX that you can run within the cube to output a value to the client tool. Calculations are great for date logic, custom aggregations, and additional measures.

A great place to start for calculations is the Business Intelligence Wizard, which creates a standard set of calculations within the cube. In preparation, you need to tell Analysis Services some information about your dimension: the type of dimension and mappings to built-in fields. For the AdventureWorks example, you need to set the dimension type to a time dimension (which in this case means a date dimension), so that you can set up the business intelligence for date information. Additionally, you need to map each of the attributes you created to a built-in field, utilizing the Calendar types. Once completed, you can go to the cube file to run the Business Intelligence Wizard. In the Wizard, follow these steps:

- 1. Open the Business Intelligence Wizard.** Click the first button on the Calculations tab, titled Add Business Intelligence to open the Wizard. The first screen gives you the option to select what type of enhancement you want to add. The options are shown in Figure 8-15.
- 2. Select the Define time intelligence option and the Calendar Hierarchy you created on the Order Date dimension.** Select all desired calculations, but only one measure, such as Order Quantity. After the wizard has completed, you will modify the script to allow the calculations to run for all measures.

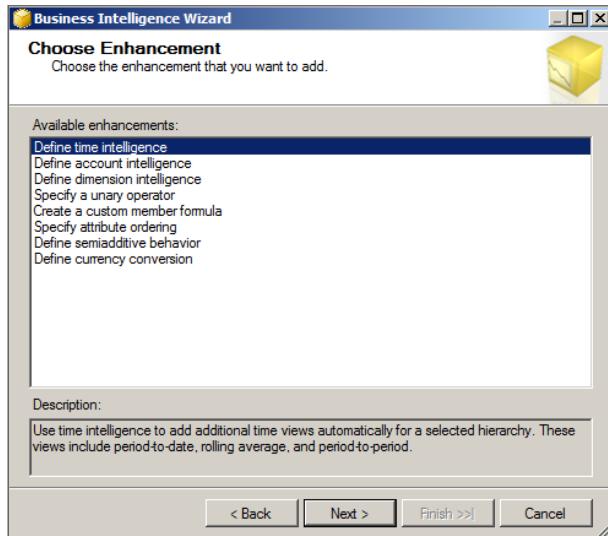


Figure 8-15: Business Intelligence Wizard options

NOTE The Calculations tab has two views: Form View and Script View. Because the Form View shows each calculation separately, you use this view to add new single calculations. In the case of reviewing or modifying many calculations, Script View is your best bet.

3. In the AdventureWorks date calculation example, you use Script View. This is shown in Figure 8-16.

```

/*
Begin Time Intelligence script for the [Order Date].[Calendar Hierarchy] hierarchy.

*/
[Create Member
CurrentCube.[Order Date].[Calendar Hierarchy Order Date Calculations].[Year to Date]
As "NA" ;]

[Create Member
CurrentCube.[Order Date].[Calendar Hierarchy Order Date Calculations].[Quarter to Date]
As "NA" ;]

[Create Member
CurrentCube.[Order Date].[Calendar Hierarchy Order Date Calculations].[Month to Date]
As "NA" ;]

[Create Member
CurrentCube.[Order Date].[Calendar Hierarchy Order Date Calculations].[Twelve Months to Date]
As "NA" ;]

[Create Member
CurrentCube.[Order Date].[Calendar Hierarchy Order Date Calculations].[Twelve Month Moving Average]
As "NA" ;]

```

Figure 8-16: Calculations tab Script View

4. The final enhancement for your date calculations is to remove the scope associated with the measure you selected. This ensures that the calculations can be used for all measures. To do this, remove the following code from the script:

```
Scope (
{
    [Measures] . [Order Quantity]
}
);
...
End Scope ;
```

In the same way, you can add business intelligence for other dimension types or add additional calculations on dimensions or measures.

Using Partitions and Aggregations

If the data in your cube becomes unwieldy through slow processing or slow query times, you need to review alternate approaches to your cube. One approach is to partition the measure group data so that you keep each section of data in its own container. Each container acts as its own unit of work, which can help performance in both querying and processing.

But what should that container look like? You set up a partition based on the data within the partition, typically your dimensional keys. A common method of creating partitions is to use the date dimension key, and create partitions for every month or every year. Alternatively, you could use a region identifier or even a customer grouping, if you have a limited number of customers.

When creating a partition, you want to follow these principles:

- Each partition should be approximately the same size.
- Queries that need to be fast should typically access one partition (although queries that cross partitions can also be fast).
- Don't put too much information into one partition. A common recommendation is to keep a partition around 20 million rows, but keep in mind that it is also dependent on the size of each row.

For the AdventureWorks example, you can add partitions to the Fact Internet Sales measure group. To do so, follow these steps:

1. Within the cube, click the Partitions tab to see that both measure groups have one partition by default. The view of the partitions is shown in Figure 8-17.

2. **Modify the partition.** You do this by navigating to the Source column and select the label FactInternetSales. You modify the existing partition by selecting the ellipses button that appears next to FactInternetSales. The window that appears shows you the information that is included in the partition.

Fact Internet Sales (1 Partition)

Partition Name	Source	Estimated Rows	Storage Mode	Aggregation Design
Fact Internet Sales	FactInternetSales	0	MOLAP	

[New Partition...](#) [Storage Settings...](#)

Fact Internet Sales Reason (1 Partition)

Partition Name	Source	Estimated Rows	Storage Mode	Aggregation Design
Fact Internet Sales Reason	FactInternetSalesReason	0	MOLAP	

[New Partition...](#) [Storage Settings...](#)

Figure 8-17: Initial Partitions tab

3. **Change the binding type to Query Binding.** This will make it easier to create new partitions and keep up with the partitions moving forward.
The view changes to show the query that Analysis Services will run to load data into the partition, but it is missing one piece: the filter!
4. **Replace the empty WHERE statement with the following code:**

```
WHERE [dbo].[FactInternetSales].[OrderDateKey]
      BETWEEN 20050101 AND 20061231
```

5. **When you finish the wizard, you are then ready to create a new partition.** You can do this by clicking the New Partition link. Select the FactInternetSales table and click Next to enter the second partition's query. This query will use a separate filter, using the following code:

```
WHERE [dbo].[FactInternetSales].[OrderDateKey]
      BETWEEN 20070101 AND 20081231
```

6. **Name the partitions differently to ensure that you can distinguish between them later.** It is also important to note that the data within the query must not overlap or have any gaps; if it does, the data will be double-counted or missed in queries.
7. **Automate a script to create new partitions as data arrives that falls outside the boundaries of the existing partitions.** Figure 8-18 shows the final view of the partitions you created.

Fact Internet Sales (2 Partitions)				
Partition Name	Source	Estimated Rows	Storage Mode	Aggregation Design
1. Fact Internet Sales 2007-08	SELECT [dbo].[FactInternetSales], [ProductKey], [dbo].[FactInternetSales].[OrderDateKey], [...]	0	MOLAP	[Progress Bar]
2. Fact Internet Sales 2005-06	SELECT [dbo].[FactInternetSales], [ProductKey], [dbo].[FactInternetSales].[OrderDateKey], [...]	0	MOLAP	[Progress Bar]

New Partition... Storage Settings....

Figure 8-18: Final Partitions tab

- Add aggregations to the partitions. Aggregations make Analysis Services hum and provide the fast querying that users know and love.

NOTE You have two options: Let the engine create aggregations automatically, or design your own. For the AdventureWorks example, you will let the engine create the aggregations.

- In the Aggregations tab from the Cube file, as shown in Figure 8-19, select the first measure group. Now, click the first icon on the toolbar to design the aggregations. Because these partitions are so similar, you can design the aggregations together. However, if you find that users will query the partitions differently, you may want to design the aggregations separately. Start by leaving the default values for the aggregation usage, and let the wizard count the number of attributes per each table.

	Aggregations	Estimated Partitio...	Partitions
- [ui] Fact Internet Sales (0 Aggregation Designs)	-	-	Fact Internet Sales 2005-06, Fact Internet Sales 2007-08
- [ui] Fact Internet Sales Reason (0 Aggregation Designs)	-	-	Fact Internet Sales Reason

Figure 8-19: Aggregations tab

Deciding how much aggregation to include in your partition is more of an art than a science. You need to balance the fast querying you can receive with the amount of time and storage it will take to create and hold the aggregations. A good rule of thumb is to watch the graph that the optimization engine makes, and when the curve starts to flatten out, stop creating any additional aggregations. The performance-gain percentage may vary based on your dataset.

Give it a try on the AdventureWorks model, and stop the aggregations wizard when it looks similar to Figure 8-20.

You've successfully created an Analysis Services cube that will scale with additional data and provide a flexible maintenance structure moving forward.

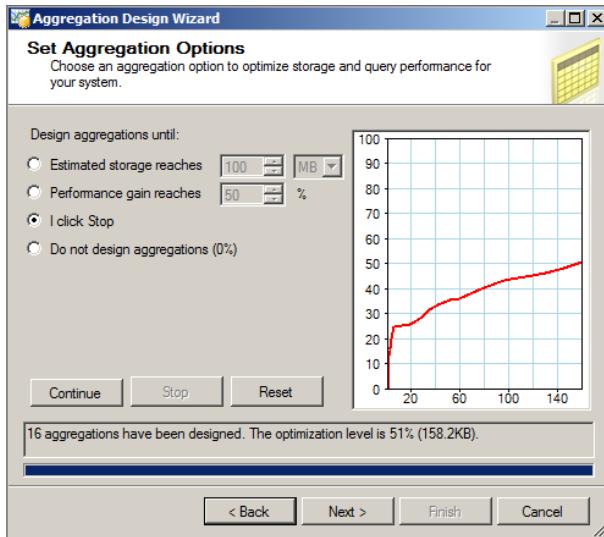


Figure 8-20: Aggregation optimization

How Do You Tune the Model?

At this point, you have created a model and enhanced it to respond quickly. Hopefully, your model works perfectly and you can start reading the next chapter. If not and you find any issues with your model, you may need to continue to tune the model. This section discusses the different storage modes, performance-tuning methods, and proper querying approaches.

Resolving Processing Issues

Similar to a tabular model, you could face slow processing of a multidimensional model. Processing issues depend on the type of storage mode and processing mode that you have configured the Analysis Services cube to use. You will learn about these options, and then learn about techniques to make processing faster.

Understanding Storage Modes

The multidimensional Analysis Services model has three storage modes, which determine how the data and pre-aggregations are stored within the Analysis

Services database. You change the storage mode per partition within the Partitions tab. The available storage modes include:

- **ROLAP:** Relational online analytical processing does not store any data in the Analysis Services database. Data is queried directly from the underlying source when a query is executed against the Analysis Services database.
- **HOLAP:** Hybrid online analytical processing stores the pre-aggregation data within the Analysis Services database. If any queries request detailed information, the data is requested from the underlying data source.
- **MOLAP:** Multidimensional online analytical processing stores all the data and pre-aggregations within the Analysis Services database. Variations of MOLAP exist (low latency, medium latency, automatic, and scheduled) that determine the frequency of the data processing. However, regular MOLAP is typically used, so you can schedule the processing after the underlying warehouse finishes its refresh.

Processing Modes

Processing the multidimensional Analysis Services database pulls in the data from the underlying data source that you specified when you initially created the model. Each object type in the database handles processing a little differently, and there are multiple options for processing. In general, think of processing in a few different ways: Wipe everything, load everything, load only changed data, and load only pre-aggregations.

It is important to only process at the necessary level for what has changed in the model or the underlying data. For example, if the product hierarchy has changed and no aggregations are built on the product hierarchy, you can just load the data. Or if you change the calculation script within the model, you don't have to process the aggregations, just the data. Additionally, you can process multiple partitions in parallel or use a default setting to allow the server to identify what it thinks should be processed.

Optimizing the Model

If your processing is slower than expected, you have some alternatives to adjust the model. Be sure to understand the effects of changing anything in the model to prevent data duplication, missed data, or worse performance than currently exists. With all that being said, here are some of the ways you can optimize your model's processing:

- **Adjust the storage mode if possible.** Is the underlying data really refreshed in real time, or would a nightly process (and a change to MOLAP storage mode) give the same results?

- Evaluate the processing mode for each object to determine if you are processing more than necessary. If so, can you modify the processing mode to not process everything?
- Look at the processing schedule. Can you parallelize the processing for it to complete faster?
- Optimize the query that pulls data from the data source by removing unnecessary columns and/or performance tuning the code. This option provides the most bang for your buck.

If none of these options works for you, see Books Online or Microsoft's operations white paper ([https://msdn.microsoft.com/en-us/Library/bb522607\(v=sql.120\).aspx](https://msdn.microsoft.com/en-us/Library/bb522607(v=sql.120).aspx)).

NOTE For a more detailed view of performance-tuning the database engine's processing, see the Microsoft white paper **Analysis Services Operations**, found here: <http://msdn.microsoft.com/en-us/library/hh226085.aspx>.

Querying

The second place where users experience slowdowns is from the client side, also known as *querying*. Although the multidimensional engine is known for its fast response, sometimes the query needs a little tweaking. Understanding how the query engine works, as shown in Figure 8-21, will help you better tune your queries. When a query request comes in, it can be served either by the formula engine (based on the calculation requested and the availability of the data in the cache) or by the storage engine (if it must dig deeper into the data than what is readily available).

It is important to isolate where the query issue is so that you can fix it in the appropriate layer. You can use SQL Server Profiler to run a trace against the multidimensional instance to evaluate the query. Some of the fixes you may make include rewriting the MDX query to use a new calculation or add appropriate filters to the query, adding or adjusting aggregations within the database, or even redesigning some of the dimensions or measure groups you created!

NOTE For additional information on query performance-tuning, see Microsoft's **Analysis Services Query Tuning** white paper: <http://msdn.microsoft.com/en-us/library/dn393915.aspx>.

As you might expect, the time to process versus query time is often inversely related. Based on any issues you have, you may want to move farther up or down the curve shown in Figure 8-22. Just be aware of the effects of the move on the opposite element.

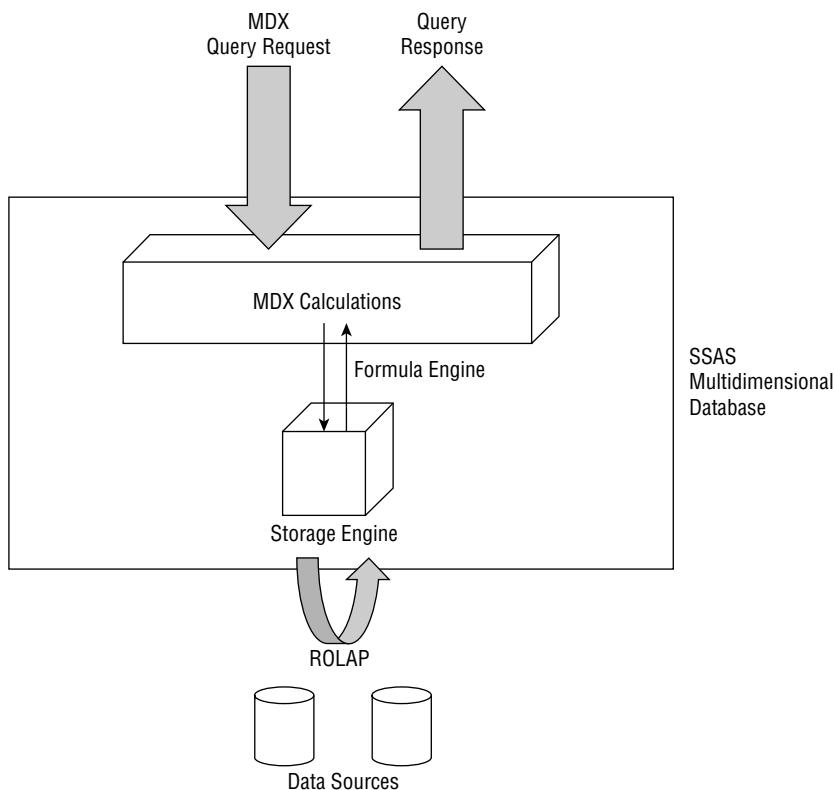


Figure 8-21: Query engine architecture

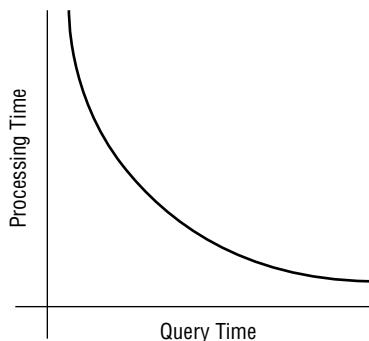


Figure 8-22: Processing time versus query time

Summary

This chapter took you through the steps of creating and optimizing an Analysis Services multidimensional model. You learned how best to design it and how to performance-tune if there are any issues. The next chapter introduces the other half of the Analysis Services product: data mining.

Discovering Knowledge with Data Mining

Data mining is the most advanced part of business intelligence. With statistical and other mathematical algorithms, you can automatically discover patterns and rules in your data that are hard to notice with on-line analytical processing and reporting. However, you need to:

- Thoroughly understand how the data mining algorithms work in order to interpret the results correctly.
- Prepare your data appropriately.
- Apply the complete data mining lifecycle for the best results in a project.

This chapter describes the process and the lifecycle of a data mining project. You will learn data mining basics, including the business value and common use cases. Next, you learn how to get an initial understanding of your data and prepare it for data mining analyses. Then you learn how the data mining algorithms in SQL Server Analysis Services (SSAS) work. This knowledge helps you to select the appropriate algorithm and create the *data mining model* for a specific analysis. You also learn how to evaluate the efficiency of different data mining models in order to select the best one for the deployment in production. Finally, you see how to maintain your mining models over time and how to integrate them in different products from the SQL Server suite and in Excel. Please note that SSAS in tabular mode does not support data mining; you need to install SSAS in Multidimensional and Data Mining mode.

Understanding the Business Value of Data Mining

With data mining, you deduce some hidden knowledge by examining, or training, the data.

DATA MINING

Data mining is a process of exploration and analysis, by automatic or semiautomatic means, of historical data in order to discover patterns and rules, which you can use later on new data to predict and forecast.

The unit of examination is called a *case*, which can be interpreted as one appearance of an entity, or a row, in a table. The knowledge is patterns and rules. In the process, you use attributes of a case, which are called *variables* in data mining terminology. For better understanding, you can compare data mining to On-Line Analytical Processing (OLAP), which is a model-driven analysis where you build the model in advance. Data mining is a data-driven analysis, where you search for the model. You examine the data with data mining algorithms.

There are many alternative names for data mining, such as knowledge discovery in databases (KDD) and predictive analytics. Originally, data mining was not the same as machine learning in that it gave business users insights for actionable decisions; machine learning determines which algorithm performs the best for a specific task. However, nowadays data mining and machine learning are in many cases used as synonyms.

Understanding Data Mining Techniques

Data mining techniques are divided into two main classes:

- **The directed, or supervised approach:** You use known examples and apply gleaned information to unknown examples to predict selected target variable(s).
- **The undirected, or unsupervised approach:** You discover new patterns inside the dataset as a whole.

Some of the most important directed techniques include classification, estimation, and forecasting. Classification means to examine a new case and assign it to a predefined discrete class. For example, assigning keywords to articles and assigning customers to known segments. Next is estimation, where you

are trying to estimate a value of a continuous variable of a new case. You can, for example, estimate the number of children or the family income. Forecasting is somewhat similar to classification and estimation. The main difference is that you can't check the forecasted value at the time of the forecast. Of course, you can evaluate it if you just wait long enough. Examples include forecasting which customers will leave in the future, which customers will order additional services, and the sales amount in a specific region at a specific time in the future.

The most common undirected techniques are clustering and affinity grouping. An example of clustering is looking through a large number of initially undifferentiated customers and trying to see if they fall into natural groupings. This is a pure example of "undirected data mining" where the user has no preordained agenda and hopes that the data mining tool will reveal some meaningful structure. Affinity grouping is a special kind of clustering that identifies events or transactions that occur simultaneously. A well-known example of affinity grouping is market basket analysis, which attempts to understand what items are sold together at the same time.

As already mentioned, data mining is the most advanced part of business intelligence. With data mining, you can extract information from your data that would otherwise remain hidden to you. It, therefore, provides you with more business value than any other business intelligence tool or process.

Common Business Use Cases

Some of the most common business questions that you can answer with data mining include:

- What's the credit risk of this customer?
- Are there any distinguishing market groups of my customers?
- What products do customers tend to buy together?
- How much of a specific product can I sell in the next period?
- What is the potential number of customers shopping in this store?
- What are the major groups of my web-click customers?
- Is this a spam email?

However, the actual questions you might want to answer with data mining could be by far broader, and would depend on your imagination only. For an unconventional example, you might use data mining to seek lowering the mortality rate in a hospital.

Data mining is already widely used in many different applications. Some of the typical usages, along with the most commonly used algorithms for a specific task, include the following:

- **Cross-selling:** Widely used for web sales with the Association Rules and Decision Trees algorithms.
- **Fraud detection:** An important task for banks and credit card issuers, who want to limit the damage that fraud creates, including that experienced by customers and companies. The Clustering and Decision Trees algorithms are commonly used for fraud detection.
- **Churn detection:** Service providers, including telecommunications, banking, and insurance companies, perform this to detect which of their subscribers are about to leave them in an attempt to prevent it. Any of the directed methods, including the Naïve Bayes, Decision Trees, or Neural Network algorithm, is suitable for this task.
- **Customer Relationship Management (CRM) applications:** Based on knowledge about customers, which you can extract with segmentation, using, for example, the Clustering or Decision Trees algorithm.
- **Website optimization:** To do this, you should know how your website is used. Microsoft developed a special algorithm, the Sequence Clustering algorithm, for this task.
- **Forecasting:** Nearly any business would like to have some forecasting, in order to prepare better plans and budgets. The Time Series algorithm is specially designed for this task.

NOTE Most of the times you want to use multiple algorithms for a single problem.

You do this in order to find the one that gives you the best result (best predictions, for example), or in order to control the quality of the predictions over time of one algorithm with another algorithm. Cross-selling is a simple problem, and using Association Rules might only be enough in most of the cases. However, fraud detection is a very complex problem, and always demands multiple algorithms. For further reading, please refer to the five articles about the fraud detection process, with this link pointing to the last one, which includes references to the previous ones: http://sqlblog.com/blogs/dejan_sarka/archive/2014/01/06/fraud-detection-with-the-sql-server-suite-part-5.aspx.

Driving Decisions, Strategies, and Processes Through Data Mining

Data mining projects execute in a cycle. Figure 9-1 shows the virtuous cycle of data mining.

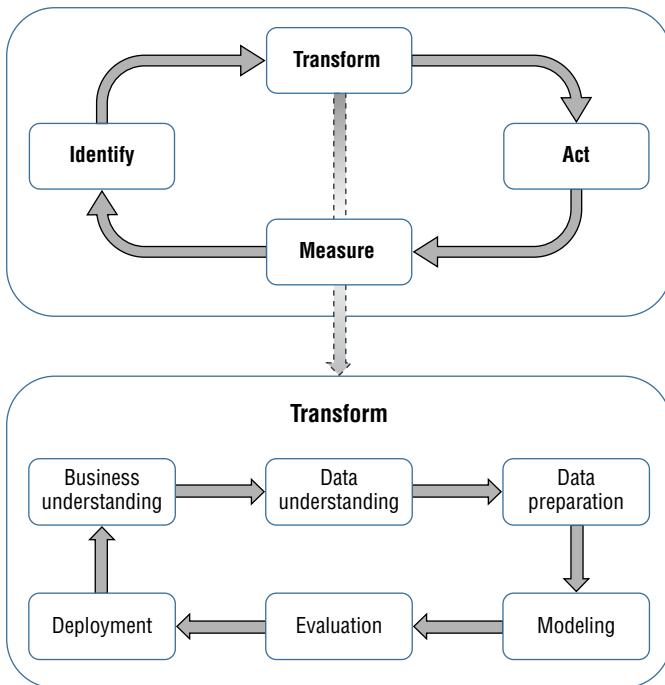


Figure 9-1: The data mining virtuous cycle

You start the data mining cycle by identifying the business problem. Then you use data mining techniques to transform the data into actionable information. By applying the data mining algorithms on your data, you get one or more trained data mining models with patterns and rules. Then you select one or more models for a deployment and usage in production for prediction, forecasting, reports and more. Then, over time, you must measure the efficiency of the models deployed. You need to refine models when they do not perform as well as they did right after the deployment. For example, predictions might become less and less accurate. By measuring the efficiency of the models, you can identify a problem with existing models, and re-start the cycle. In addition, after successfully solving one business problem, you can identify the next problem suitable for data mining.

The “transform” part of the virtuous data mining cycle has its own internal cycle. After you understand the business problem and the business itself, you start the “transform” step with data overview to gain a deeper understanding of your data. Then you must prepare the data appropriately for the algorithm you plan to apply. You then model data mining models—for each model, you select an algorithm as well as attributes you will use in the analysis. You then fine-tune the algorithms’ parameters and evaluate the models. If you don’t get the results you expected, you start the cycle again by further examining and

massaging the data. You exit this cycle when you decide which model or models you are going to deploy.

As you can see, data mining is not a single project. With data mining, you enter a continuous learning cycle. Over time, you obtain deeper and deeper insights in your business, and make better and better predictions.

Data flows through each step of the data mining cycle. Following the data flow also helps you understand the data mining process and where in your IT ecosystem data mining influences business decisions. Figure 9-2 shows the data mining data flow.

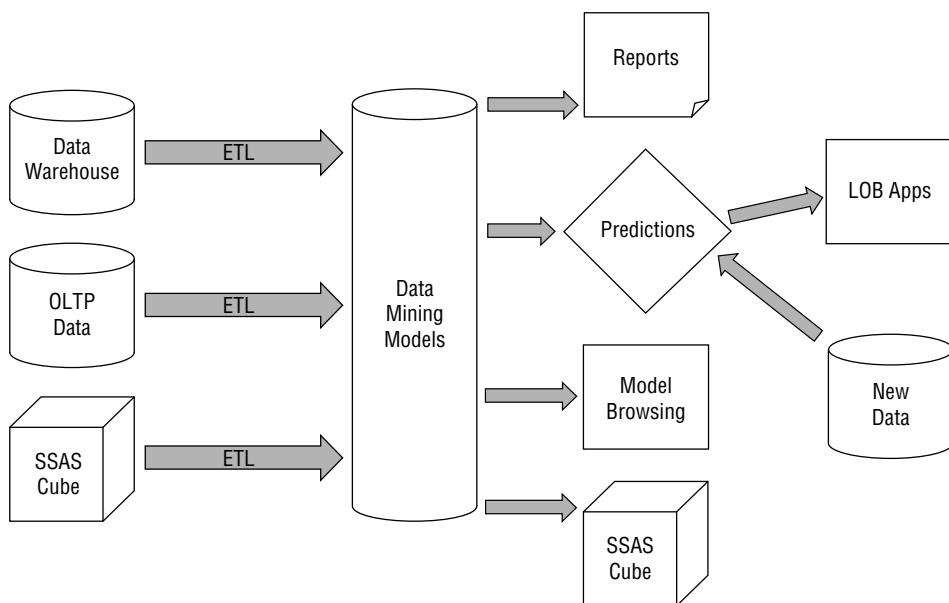


Figure 9-2: The data mining data flow

You gather data you need to analyze from relational and analytical sources. Then you need to prepare this data for data mining. Typically, you must merge, clean and transform this data with an extraction, transformation, and loading (ETL) process. You can use SQL Server Integration Services (SSIS) to develop the ETL process. Then you train the data using different mining algorithms. Depending on your data, some algorithms might give you better patterns than others. You can change the parameters of the algorithms as well. Then you evaluate which model is the best. Then you deploy one or more models in production and start using them. Usage of the models could be simple, for example you can explore the models in a browser, or create reports and distribute them. You can use the models in a more advanced way; for example, you can use the models for prediction queries that join the model with a new dataset. Predictions can

be used in line-of-business (LOB) applications. You can use the structures you found to restructure your SSAS models, for example, you can add a data mining model as a new dimension in a cube.

Getting the Basics Right

Before starting the analysis, you need to prepare the data. In data mining, you analyze cases using their variables. In a simple approximation, you can think of a case as a row in a table, and a variable as a column in the same table. For many data mining projects, you prepare a single table. You can map many cases to rows in a source table in a line of business (LOB) system database or in a data warehouse (DW). However, sometimes it is not so easy to define what exactly your case is. For example, for a credit risk analysis, you might define a family as a case, and not a single customer. When you prepare the data for data mining, you have to transform the source data accordingly. Examples of cases include:

- **Credit risk analysis:** Case = Family
- **Product profitability analysis:** Case = Product
- **Promotion success analysis:** Case = Promotion

For each case, you need to encapsulate all available information in the columns of the table you are going to analyze.

For simple cases, you can really prepare a single table, where each row represents an entity (in data mining terminology, a case) and each column represents an attribute (in data mining terminology, a variable). Now imagine that you need to do a market basket analysis. You must know which products were purchased together in a single transaction. The transaction (e.g., an order) groups products (e.g., details about the products in an order) together. The order details in this example are represented as a single table data type column of a row of each customer. This column is a *nested table*. Figure 9-3 shows an example of orders with customer demographics data and products purchased per order.

Order ID	Customer Age	Marital Status	Yearly Income	Product Purchases	
				Product	Quantity
1	35	M	120,000	TV	1
				Beer	6
				Chips	3
2	20	S	10,000	VCR	1
				TV	1
				Cake	12
3	57	M	70,000	Chips	2
				Beer	1

Figure 9-3: A nested table example

SSAS supports one level of nesting. For example, if you use the Customers table for your cases, you must join the Orders and Order Details tables into a single table and then define this joined table as a nested table in your Customers case table. You can have multiple columns that are nested tables. However, as mentioned, you can have only one level of nesting.

An important note if you are preparing data in an RDBMS like SQL Server: SQL Server does not support nested tables. When you prepare the data, you create separate table for each column of the case table that should be a nested table, and then use foreign keys to connect those tables to the main case table. You define these tables as nested tables when you create a data mining model.

Understanding the Data

Understanding your data is a crucial factor for a successful data mining project. First you need to understand how data values are measured in your dataset. You might need to check this with a subject matter expert and analyze the business system that is the source for your data. There are different ways to measure data values and different types of columns (in data mining terminology, different types of variables):

Using Discrete Values

These are categorical or nominal variables that have no natural order. Examples include states, status codes, and colors. Ranks can also take a value but only from a discrete set of values. They have an order but do not permit any arithmetic. Examples include opinion ranks and binned true numeric values.

Specific types of categorical variables are:

- **Single-valued (constants):** Are not very interesting because they do not contribute any information.
- **Two-valued (dichotomous):** Have two values that are minimally needed for any analysis.
- **Binary variables:** Are specific dichotomous variables that take on only the values 0 and 1.

Using Continuous Values

Intervals have one or two boundaries as well as an order, and allow some arithmetic-like subtraction but not always summations. Examples include dates, times, and temperatures. True numeric variables support all arithmetic. Examples include amounts and values. Specific types of continuous variables include:

- **Monotonic variables:** Increase monotonously without bound. If they are simply IDs, they might be not interesting. Still, you can transform them (binned into categories) if the ever-growing ID contains time order information (lower IDs are older than higher IDs).

Checking Data Distribution

After you understand how your data values are measured, you should check the *data distribution* of each variable you are going to use. A simple graphical representation of the data distribution can quickly reveal erroneous or suspicious values, specific variable types, like monotonic variables, and more. Such variables and values could have a great impact on the quality of your models, and you should probably exclude them from the analyses.

You can use Excel, SQL Server Reporting Services (SSRS) or Power View reports, or any third party tool for a graphical data distribution overview. You can also use Power Pivot and Excel for a quick overview of a distribution of discrete variables. Inside SSIS, you can use the Data Profiling task. Transact-SQL queries can be very useful for checking the data distribution as well.

DATA DISTRIBUTION OVERVIEW TOOLS

There is nearly an unlimited set of options for an overview of your data distribution. However, you don't need to buy expensive third party tools. You have many possibilities for the data overview in SQL Server suite. For example, in this section you learn how to use Power Pivot and Excel for the data distribution overview. You don't need to know all of the tools; you should use the tools you already have and know how to use.

In the following example, you learn how to quickly overview your discrete variables with Power Pivot and Excel:

1. **Create a new Excel workbook.**
2. **Open the Power Pivot window.**
3. **Import data from your SQL Server with the AdventureWorks data warehouse database.** For this scenario, you choose the *vTargetMail* view and import the data.
4. **After the import is finished, check the data in the Power Pivot window.** Highlight the *BikeBuyer* column (the last column).
5. **Create a new measure.** In the Measures group, click the AutoSum drop-down list and select Sum. Rename the measure to *BikeBuyerTotal*. Change the DAX formula to:

```
BikeBuyerTotal:=SUM( [BikeBuyer] )
```

6. **Create two additional measures.** These are count of cases and percentage of bike buyers. Use the following DAX formulas, formatting the last measure as a percentage:

```
BikeBuyerCount := COUNTA([BikeBuyer])
BikeBuyerPct := SUM([BikeBuyer]) / COUNTA([BikeBuyer])
```

7. **Create a pivot chart.** Use the BikeBuyerCount measure for the values and the EnglishOccupation attribute for the axis (categories) area. You should now have a graph like that shown in Figure 9-4. You can see the distribution of the EnglishOccupation variable in this graph.

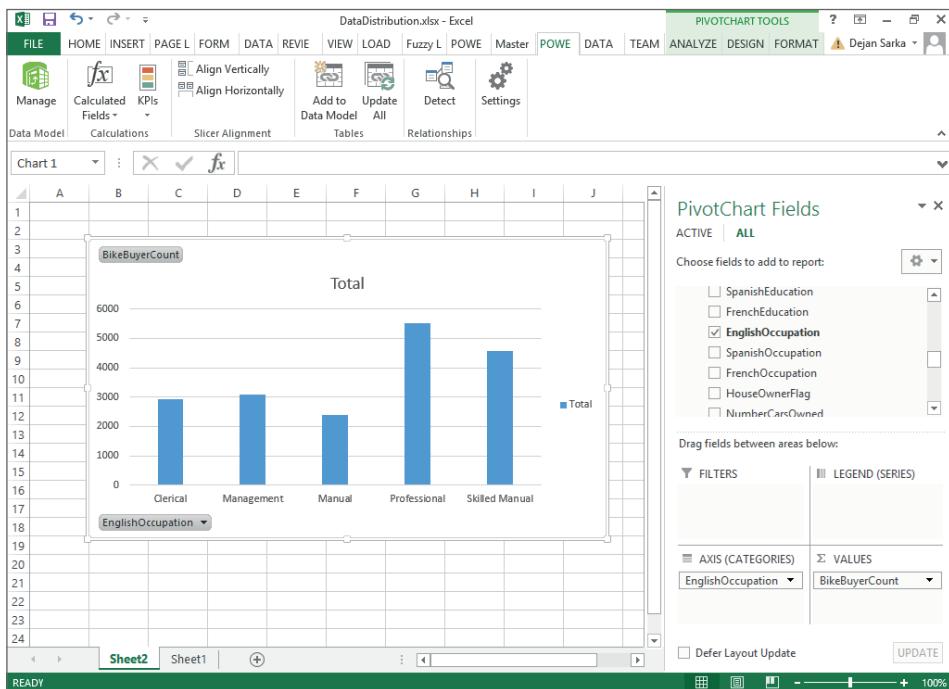


Figure 9-4: The distribution of the EnglishOccupation variable

With the Power Pivot model you just created, you can now quickly check the distribution of any discrete variable. In addition, you can also check how many bike buyers are in each class of a discrete variable, and the percentage of the bike buyers in this class.

Training and Test Datasets

After you understand your data, you should prepare it for data mining. This might be the most complex task of your data mining project. For example, many

times you do not have appropriate variables for an analysis in your transactional or data warehouse databases. *Derived variables* can lead to much better and more accurate analyses. However, you need to have domain knowledge and know your data in order to create useful derived variables. Variants of derived variables include:

- **Sequential ID:** These usually give you information about which case came into the system earlier and which came later.
- **Smoothing or filtering the peak values:** Moving averages can give you a clearer picture about trends.
- **Creating summarizations:** Summarizations can help with huge numbers of categories or with entities that quickly become obsolescent. For example, a single model of a mobile phone has quite a short lifetime.

Even simple combinations of input columns can give you useful derived variables. For example, in medicine, the obesity index tells doctors much more than pure height and pure weight of a patient. Just to give you an idea how to create derived variables in your projects, here are some examples of combinations:

- Height²/weight (obesity index)
- Passengers * miles
- Population/area (population density)
- Activation date – application date (time needed for activation)
- Credit limit – balance
- Prospect age/agent age (age ratio)

It's out of the scope of this chapter to go into the details for all possible ways to prepare the data for data mining. However, you should always do one thing when you create predictive models: Prepare your data to test the efficiency of the predictions. To do so, you simply split the data into training and test sets. You use the *training set* to train the model. A common practice is to use 70 percent of data for the training set. The remaining 30 percent of the data goes into the *test set*, which is used for predictions. When you know the value of the predicted variable, you can measure the quality of the predictions.

The problem with sampling is how to get truly random samples. When you split the data into the training and test sets, you should not create a pattern in the split. You should use a tool that does the statistically random sampling for you. For example, you can use the SSIS Percentage Sampling and Row Sampling transformations. In addition, SSAS can randomly split data into training and test sets for a single *data mining structure* or a single data mining model. If you want to create samples that you can use for multiple mining structures (and not only for training and test sets), you should use the SSIS sampling transformations.

Defining the Data Mining Structure

The data mining structure defines the overall data from which mining models are built. It defines all possible variables and cases for the models. Multiple models can share the same structure; however, each model can use a subset of variables only, and filter the cases as well. For a mining structure, you can also define optional partitioning into training and testing sets. The data mining structure specifies which data source view objects you use. Figure 9-5 shows the relationship between a data mining structure, data source, data source view, and data mining models.

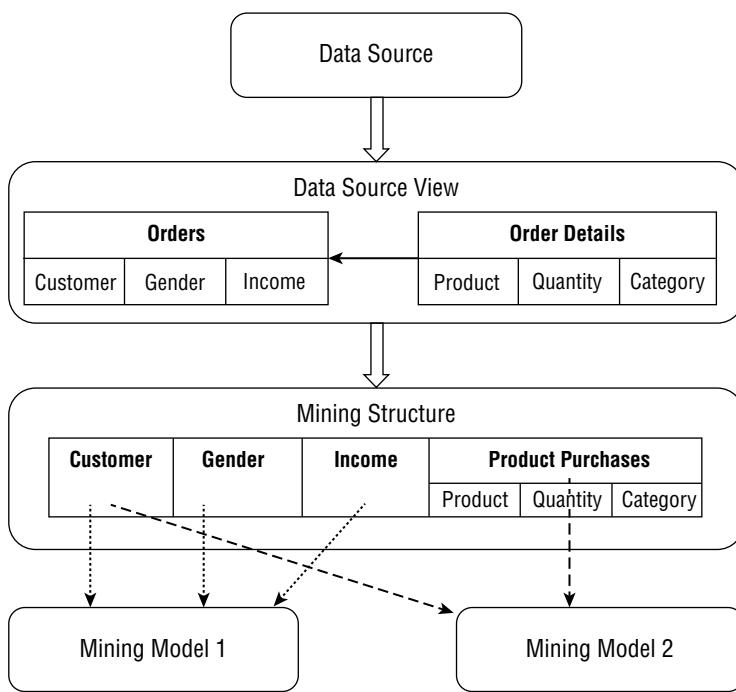


Figure 9-5: Data mining structure

The Data Mining Model

A data mining model is created when you apply a data mining algorithm on data. It is stored in an SSAS database as a table; however, this is not a SQL Server table—it can have nested columns as well. The content of this table is a set of data, statistics, and patterns that you can apply to new data to generate predictions and make inferences about relationships.

Mining models get data from a structure. When you process the structure, or train the models in the structure, the models then analyze that data by using data mining algorithms. The mining structure and mining models are separate objects.

The mining structure stores metadata, the information about the data source. A mining model stores metadata and data. The metadata is the variables and the algorithms used, and the parameters of the algorithms. The data is the information derived from statistical processing of the data, such as the patterns found as a result of analysis. A model is empty until you process the mining structure.

Applying the Microsoft Data Mining Techniques with Best Practices

After this quite lengthy introduction, it is time to create some data mining models. You will learn how to use different algorithms through examples. For a start, the following list gives you a brief overview of the SSAS data mining algorithms and their usage.

- **Association Rules:** The algorithm used for market basket analysis, this defines an itemset as a combination of items in a single transaction. It then scans the data and counts the number of times the itemsets appear together in transactions. Market basket analysis is useful to detect cross-selling opportunities.
- **Clustering:** This groups cases from a dataset into clusters containing similar characteristics. You can use the Clustering method to group your customers for your CRM application to find distinguishable groups of your customers. In addition, you can use it for finding anomalies in your data. If a case does not fit well to any cluster, it is kind of an exception. For example, this might be a fraudulent transaction.
- **Naïve Bayes:** This calculates probabilities for each possible state of the input attribute for every single state of predictable variable. Those probabilities predict the target attribute based on the known input attributes of new cases. The Naïve Bayes algorithm is quite simple; it builds the models quickly. Therefore, it is very suitable as a starting point in your prediction project. This algorithm does not support continuous attributes.
- **Decision Trees and Regression Tress:** The most popular DM algorithm, it predicts discrete and continuous variables. It uses the discrete input variables to split the tree into nodes in such a way that each node is more pure in terms of target variable, i.e., each split leads to nodes where a single state of a target variable is represented better than other states. For continuous predictable variables, you get a piecewise multiple linear regression formula with a separate formula in each node of a tree. A tree that predicts continuous variables is a Regression Tree.

- **Linear Regression:** Predicts continuous variables, using a single multiple linear regression formula. The input variables must be continuous as well. Linear Regression is a simple case of a Regression Tree, a tree with no splits.
- **Neural Network:** This algorithm is from artificial intelligence, but you can use it for predictions as well. Neural networks search for nonlinear functional dependencies by performing nonlinear transformations on the data in layers, from the input layer through hidden layers to the output layer. Because of the multiple nonlinear transformations, neural networks are harder to interpret compared to Decision Trees.
- **Logistic Regression:** As Linear Regression is a simple Regression Tree, a Logistic Regression is a Neural Network without any hidden layers.
- **Sequence Clustering:** This searches for clusters based on a model, and not on similarity of cases as Clustering does. The models are defined on sequences of events by using Markov Chains. Typical usage of the Sequence Clustering would be an analysis of your company's website usage, although you can use this algorithm on any sequential data.
- **Time Series:** You can use this algorithm to forecast continuous variables. Internally, the Time Series uses two different algorithms. For short-term forecasting, the Auto-Regression Trees (ART) algorithm is used. For long-term prediction, the Auto-Regressive Integrated Moving Average (ARIMA) algorithm is used. You can mix the blend of algorithms used by using the mining model parameters.

Using Microsoft Association Rules

The Microsoft Association Rules algorithm is specifically designed for use in market basket analyses. This knowledge can additionally help in identifying cross-selling opportunities and in arranging attractive packages of products. This is the most popular algorithm used in web sales. You can even include additional discrete input variables and predict purchases over classes of input variables.

The algorithm considers each attribute/value pair (such as product/bicycle) as an item. An *itemset* is a combination of items in a single transaction. The algorithm scans through the dataset trying to find itemsets that tend to appear in many transactions. Then it expresses the combinations of the items as *rules* (such as "if customers purchase potato chips, they will purchase cola as well").

Often association models work against datasets containing nested tables, such as a customer list followed by a nested purchases table. If a nested table exists in the dataset, each nested key (such as a product in the purchases table) is considered an item.

Understanding Measures

Besides the itemsets and the rules, the algorithm also return some measures for the itemsets and the rules. These measures include:

- ***Support***, or frequency, means the number of cases that contain the targeted item or combination of items. Therefore, support is a measure for the itemsets.
- ***Probability***, also known as confidence, is a measure for the rules. The probability of an association rule is the support for the combination divided by the support for the condition. For example, the rule “If a customer purchases cola, then they will purchase potato chips” has a probability of 33%. The support for the combination (potato chips + cola) is 20%, occurring in one of each five transactions. However, the support for the condition (cola) is 60%, occurring in three out of each five transactions. This gives a confidence of $0.2 / 0.6 = 0.33$ or 33%.
- ***Importance*** is a measure for both, itemsets and rules. When importance is calculated for an itemset, then when importance equals one, the items in the itemset are independent. If importance is greater than one, then the items are positively correlated. If importance is lower than one, then the items are negatively correlated. When importance is calculated for a rule “If {A} then {B},” then the value zero means there is no association between the items. Positive importance means that the probability for the item {B} goes up when the item {A} is in the basket, and negative importance of the rule means that the probability for the item {B} goes down when the item {A} is in the basket.

Fine-Tuning with Algorithm Parameters

You can fine tune the algorithm through algorithm parameters. The most important parameters include:

- ***Minimum_Support***: Specifies the minimum number of cases that must be in an itemset before the algorithm generates a rule. You can set this parameter as a percentage or as a number. If this value is less than 1, the value represents a percentage of the total cases.
- ***Maximum_Support***: Specifies the maximum number of cases in an itemset. You can use this parameter to eliminate items that appear frequently and therefore potentially have little meaning.
- ***Minimum_Probability***: Specifies the minimum probability that a rule is generated. For example, setting this value to 0.5 specifies that no rule with less than 50% probability is generated.

- **Minimum_Importance:** Specifies the minimum importance threshold. The algorithm filters out the rules with an importance lower than the value specified.

Testing the Algorithm

You can use the AdventureWorks data warehouse data to test the Association Rules algorithm.

1. **Create a new project.** Open SQL Server Data Tools (SSDT) and create a new Analysis Services Multidimensional and Data Mining project.
2. **Create a new data source for your AdventureWorksDW data warehouse.** Use the Inherit option for the impersonation information. Then create a new data source view using the data source you have just created. Add the following table and views from the data warehouse to the data source view: ProspectiveBuyer, vAssocSeqOrders, vAssocSeqLineItems, vTargetMail, and vTimeSeries. For this example, you need only the first two views. However, you will use the other views and the table from the data warehouse in the examples that follow later in this chapter.
3. **Create a virtual foreign key between the vAssocSeqLineItems and vAssocSeqOrders views.** Click the OrderNumber column in the vAssocSeqLineItems view to select it. Drag and drop it to the OrderNumber column in the vAssocSeqOrders view.
4. **Create a new mining structure.** In Solution Explorer, right-click the Mining Structures folder and select the New Mining Structure option to start the Data Mining Wizard. Use the existing relational database or data warehouse. Select the Microsoft Association Rules technique. Use the data source view you have just created when you are asked to select the data source view by the Data Mining Wizard. Select the vAssocSeqOrders view as the case and the vAssocSeqLineItems view as the nested table. Use the OrderNumber column from the vAssocSeqOrders view and the Model column from the vAssocSeqLineItems view as the key columns. Select the Model column as Predictable. Your selection should look like the one shown in Figure 9-6. Accept the suggested column's content and data types. Specify zero percent data for testing. You keep aside around 30 percent of data for testing for predictive algorithms only. Testing predictive algorithms is explained later in this chapter. Give the model and the structure a name. Allow drillthrough if you wish. When this option is allowed, you can drill through the cases used to train the model from the mining model viewer.

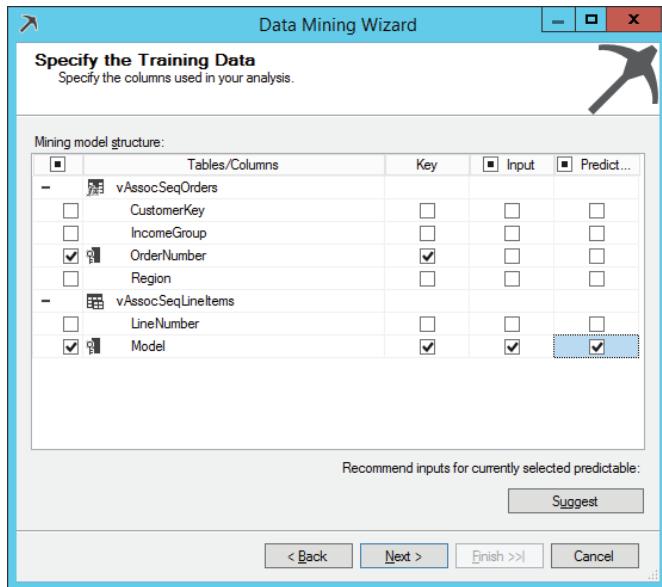


Figure 9-6: Specifying the case and the nested tables and columns

5. In the Mining Models view (use the Mining Models tab in the designer, the second tab from the left, near the Mining Structure tab), right-click the model and select the Set Algorithm Parameters option, as shown in Figure 9-7. Check the parameters. Set Minimum_Probability to 0.1 and Minimum_Support to 0.01. Save, deploy, and process the model (or the whole project).

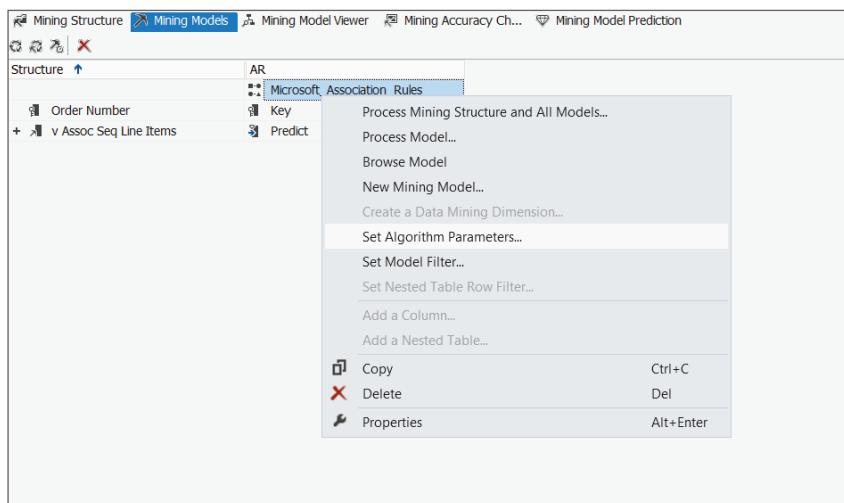


Figure 9-7: Setting the algorithm parameters

6. **Now it is time to harvest the results.** Click the Mining Model Viewer tab. In the Rules viewer, select the Show attribute name and value option from the Show drop-down list. In the Filter Rule drop-down list, select Sport-100. Check the rules for the Sport-100 models. Also click the Itemsets and the Dependency Network tabs to check the information you can get from the other two data mining viewers for the Association Rules algorithm.

Grouping Data with Microsoft Clustering

Clustering is the process of grouping the data into classes or clusters so that objects within a cluster have high similarity in comparison to one another, but are very dissimilar to objects in other clusters. Dissimilarities are assessed based on the attribute values describing the objects. You typically use the Clustering algorithm to discover distinct groups of your cases, for example groups of customers.

Understanding the Clustering Algorithm

Microsoft Clustering actually includes two different clustering algorithms. The first one divides a dataset into a predetermined ("k") number of clusters around the average location (mean). Therefore, this is also called the *K-Means* algorithm. With the K-Means algorithm, each object is assigned to exactly one cluster, with a probability equal to 1.0; and to all other clusters with a probability equal to 0.0. This is called *hard clustering*.

Instead of distance, you can use a probabilistic measure to determine cluster membership. For example, you can cover the objects with bell curves for each dimension with a specific mean and standard deviation. A case is assigned to every cluster with a certain probability. Because clusters can overlap, this is called *soft clustering*. The *Expectation-Maximization* (EM) method changes the parameters of the bell curve to improve covering of the cases with clusters in each iteration.

The most important parameters for the Microsoft Clustering algorithm are:

- **Clustering_Method:** Specifies one of the available clustering methods:
 - Scalable EM
 - Non-scalable EM
 - Scalable K-Means
 - Non-scalable K-Means

- **Cluster_Count:** Specifies the number of clusters to be built by the algorithm.
- **Minimum_Support:** Specifies the minimum number of cases in each cluster.
- **Stopping_Tolerance:** Specifies the value that is used to determine when the algorithm is finished building the model. The model is built when the overall change in cluster probabilities is less than the ratio of the Stopping_Tolerance parameter divided by the size of the model.
- **Maximum_Input_Attributes:** Specifies the maximum number of input attributes before the algorithm invokes feature selection and selects the most influential attributes only.
- **Maximum_States:** Specifies the maximum number of attribute states of the input variables that the algorithm supports. If the number of states that an attribute has, is larger than the maximum number of states, the algorithm uses the attribute's states with the highest frequency and ignores the remaining states.

A Clustering Algorithm Example

In the next example, you use the Clustering algorithm to check whether there are hidden clusters of customers based on demographic and purchasing data.

1. **In SSDT, open the project you created for testing the Association Rules algorithm.** You should have prepared the data source and the data source view in that project.
2. **Create a new data mining structure.** Use the existing relational database or data warehouse. Select the Microsoft Clustering technique. Use the data source view you already have in the project. You use the dbo.vTargetMail dataset again for this example.
3. **Input values.** Use CustomerKey as the key column (selected by default), BikeBuyer as the input and predictable attributes, and the following columns as the input columns:
 - CommuteDistance
 - EnglishEducation
 - EnglishOccupation
 - Gender
 - HouseOwnerFlag
 - MaritalStatus
 - NumberCarsOwned

- NumberChildrenAtHome
- Region
- TotalChildren

4. In the **Specify Columns' Content and Data Type** screen, click the **Detect** button. The content Type of all columns except for Customer Key should be Discrete.
5. In the **Create Testing Set** screen, leave the percentage of data for testing at 30%. Give the structure and the model a name and finish the wizard. Save, deploy, and process the model (or the whole project).
6. Click the **Mining Model Viewer** tab. Use the default Cluster Diagram view. Color the background using value 1 of the Bike Buyer attribute.

You can see that some clusters have a high density of buyers. Check also the Cluster Profiles, Cluster Characteristics, and Cluster Discrimination viewers, to get the comprehension about the class of the customers each cluster represents.

Building Mining Models with Microsoft Naïve Bayes

The Microsoft Naïve Bayes algorithm quickly builds mining models that you can use for classification and prediction. It calculates probabilities for each possible state of the input attribute, given each state of the predictable attribute. You can later use the probabilities to predict an outcome of the target attribute based on the known input attributes. The probabilities that generate the model are calculated and stored during the processing of the mining structure. The algorithm supports only discrete or discretized attributes, and it considers all input attributes to be independent. The Naïve Bayes algorithm produces a simple mining model that you can consider a starting point in the data mining process. Because most of the calculations in creating the model are generated during mining structure processing, results are returned quickly. This makes the model a good option for exploring the data and for discovering how various input attributes are distributed in the different states of the predicted attribute.

You use the Naïve Bayes algorithm for classification. You want to extract models describing important data classes and then assign new cases to predefined classes. Some typical usage scenarios include:

- Categorizing bank loan applications (safe or risky) based on previous experience
- Determining which home telephone lines are used for Internet access
- Assigning customers to predefined segments (note that the segments are predefined, while with the Clustering algorithms you are searching for the segments themselves)
- Quickly obtaining a basic comprehension of the data by checking the correlation between input variables

Microsoft Naïve Bayes includes the following parameters:

- **Maximum_Input_Attributes:** Specifies the maximum number of input attributes before the algorithm invokes feature selection and selects the most influential attributes only.
- **Maximum_Output_Attributes:** Specifies the maximum number of output attributes before the algorithm will select the most important outputs (the most popular).
- **Maximum_States:** Specifies the maximum number of attribute states of the input variables that the algorithm supports.
- **Minimum_Dependency_Probability:** Parameter defines the threshold for input attributes' influence on the prediction of the output. Larger values reduce the number of attributes in the content of the model.

You will add a Naïve Bayes model to the mining structure you used for testing the Clustering algorithm later; at the same time you will add a Decision Trees model.

Using the Microsoft Decision Trees

Decision Trees is the most popular data mining algorithm. The algorithm is not very complex, yet it gives very good results in most of the cases. In addition, you can easily understand the result. You use Decision Trees for classification and prediction. Typical usage scenarios include:

- Predicting which customers will leave
- Targeting the audience for mailings and promotional campaigns
- Explaining reasons for a decision

Microsoft Decision Trees is a directed technique. Your target variable is the one that holds information about a particular decision, divided into a few discrete and broad categories (yes/no; liked/partially liked/disliked, etc.). You are trying to explain this decision using other gleaned information saved in other variables (demographic data, purchasing habits, etc.). With limited statistical significance, you are going to predict the target variable for a new case using its known values of the input variables based on the results of your trained model.

You use *recursive partitioning* to build the tree. The data is split into partitions using a certain value of one of the explaining variables. The partitions are then split again and again. Initially the data is in one big box. The algorithm tries all possible breaks of both input (explaining) variables for the initial split. The goal is to get purer partitions considering the classes of the target variable. The tree continues to grow using the two new partitions as separate starting points and

splitting them more. You have to stop the process somewhere. Otherwise, you could get a completely fitted tree that has only one case in each class. The class would be, of course, absolutely pure and this would not make any sense; you could not use the results for any meaningful prediction. This phenomenon is called *over-fitting*. You control the growth of the tree and prevent over-fitting with algorithm parameters. The Microsoft Decision Trees method actually includes another algorithm, the Regression Trees, which is explained later in this chapter. I am including here only the most important parameters that pertain to Decision Trees:

- **Minimum_Support:** Determines the minimum number of cases that is required to generate a split in the decision tree. By raising this number you limit the splits and thus prevent over-fitting.
- **Complexity_Penalty:** Another parameter that controls the growth of a tree. A low value increases the number of splits, and a high value decreases the number of splits.

In the next example, you will add two mining models to the mining structure you created when you tested the Clustering algorithm example. You will use the Naïve Bayes and the Decision Trees algorithms.

1. **In SSDT, open the project you created for testing the mining algorithms in this chapter.** In Solution Explorer, double-click the mining structure you created when you created a Clustering model. Click the Mining Models tab.
2. **Create a new model.** Right-click the Clustering model. Select New Mining Model from the pop-up menu. This way, you are creating a new model using the same structure you used previously.
3. **Select the Microsoft Decision Trees algorithm.**
4. **Repeat the same process to add also a Naïve Bayes model.**
5. **Save, deploy, and process the model (or the whole project).**
6. **Click the Mining Model Viewer tab.** In the Mining Model drop-down list, select the Decision Trees model. Color the background using value 1 of the Bike Buyer. This way, you can easily find branches with a higher percentage of bike buyers in the train set. Change the Default Expansion to five levels. Hover the mouse over some node to get the details of the node. Figure 9-8 shows the tree.

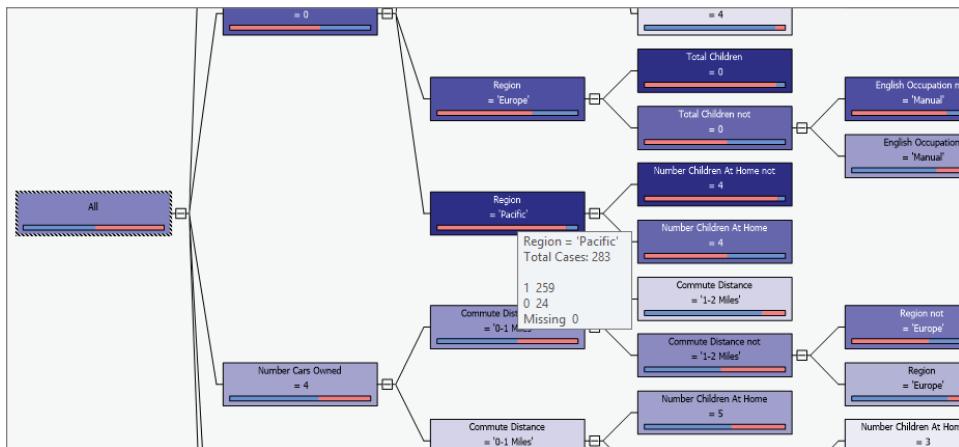


Figure 9-8: Decision Trees Algorithm

Using Microsoft Neural Network and Microsoft Logistic Regression

A neural network is a powerful data modeling tool that captures and represents complex input/output relationships. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform “intelligent” tasks similar to those performed by the human brain. Neural networks resemble the human brain in the following two ways:

- A neural network acquires knowledge through learning.
- A neural network’s knowledge is stored within connected *neurons*.

The Neural Network algorithm is an artificial intelligence technique that explores more possible data relationships than other algorithms. Because it is such a thorough technique, the processing of it is usually slower than the processing of other classification algorithms.

A neural network consists of basic units modeled after biological neurons. Each unit has many inputs that it combines into a single output value. These inputs are connected together, so the outputs of some units become inputs into other units. The network can have one or more middle layers called hidden layers. The simplest are feed-forward networks, where there is only a one-way flow through the network from the inputs to the outputs. There are no cycles in the feed-forward networks.

Understanding Activation Functions

As mentioned, units combine inputs into a single output value. This combination is called the unit's activation function. An activation function has two parts:

- The combination function that merges all of the inputs into a single value (weighted sum, for example)
- Transfer function, which transfers the value of the combination function to the output value of the unit. The linear transfer function would do just the linear regression. The transfer functions are S-shaped, like the sigmoid function:

$$\text{Sigmoid}(x) = 1 / (1 + e^{(-x)})$$

A single hidden layer is optimal, so the Neural Network algorithm always uses a maximum of one (or zero for Logistic Regression).

The Microsoft Neural Network algorithm uses the hyperbolic tangent activation function in the hidden layer and the sigmoid function in the output layer. Figure 9-9 shows a neural network schema.

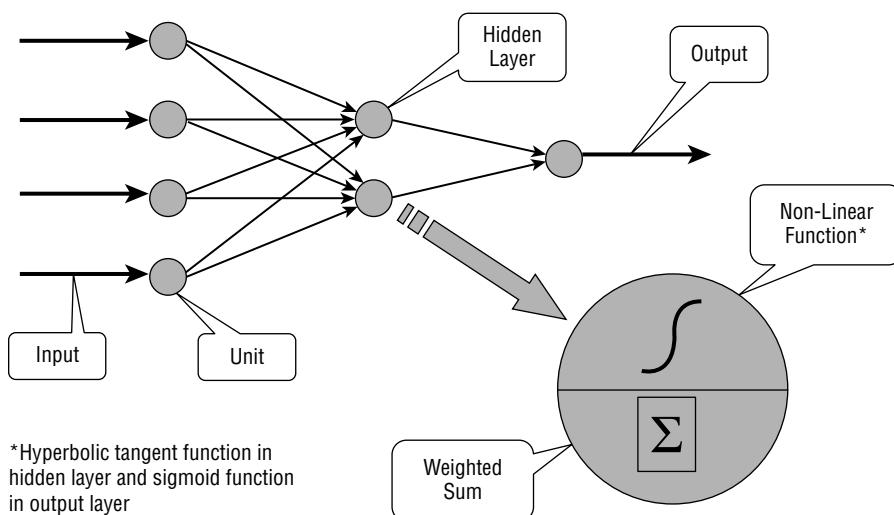


Figure 9-9: Neural Network Schema

Understanding Backpropagation

The learning process of a neural network is called *backpropagation*.

BACKPROPAGATION

Backpropagation is the process of setting the best weights on the inputs of each of the units.

The backpropagation process does the following:

- Gets a training example and calculates outputs
- Calculates the error—the difference between the calculated and the expected (known) result
- Adjusts the weights to minimize the error

The most important parameters for the Microsoft Neural Network algorithm are:

- **Hidden_Node_Ratio:** Specifies the ratio of hidden neurons to input and output neurons. If you specify zero, then no hidden layer is created.
- **Maximum_States:** Specifies the maximum number of attribute states of the input variables that the algorithm supports.

The Microsoft Logistic Regression algorithm is a variation of the Microsoft Neural Network algorithm, where the HIDDEN_NODE_RATIO parameter is set to 0. This setting creates a Neural Network model that does not contain a hidden layer and therefore is equivalent to logistic regression.

A Neural Network Example

In the next example, you are adding a Neural Network and a Logistic Regression mining model to the structure with the Clustering, Decision Trees, and Naïve Bayes models.

1. **In SSDT, open the project you created for testing the mining algorithms in this chapter.** In Solution Explorer, double-click the mining structure you created when you created a Clustering model. Click the Mining Models tab.
2. **Right-click the Clustering model.** Select New Mining Model from the pop-up menu. This way, you are creating a new model using the same structure you used previously. Select the Microsoft Neural Network algorithm.
3. **Repeat the same process to add also a Logistic Regression model.**
4. **Save, deploy, and process the model (or the whole project).**
5. **Click the Mining Model Viewer tab.** In the Mining Model drop-down list, select the Neural Network model. Check what favored value 1 and what favors value 0 of the Bike Buyer target variable. Check the results of the Logistic Regression algorithm as well.

Using Microsoft Linear Regression and Microsoft Regression Trees

In Linear Regression, you want to show one continuous variable (e.g., sales quantity) as a function of another continuous variable. The linear function between two variables is a line determined by its slope and its intercept. Logically, your goal is to calculate the slope and the intercept.

You start with the formula for the line, where the slope is denoted with b and the intercept with an a:

$$Y' = a + b * X$$

Multiple regression allows a response variable Y to be modeled as a linear function of a multidimensional feature vector:

$$Y = a + b_1 * X_1 + b_2 * X_2 + \dots$$

If you want to get the Regression Trees, you need to use the Microsoft Decision Trees algorithm with a continuous predictable variable and at least one continuous input variable. When the Decision Trees algorithm builds a tree based on a continuous predictable column, each node contains a regression formula. The Decision Trees algorithm converts to the Regression Trees. A split occurs at a point of non-linearity in the regression formula. In addition, a split can occur for different classes of a discrete input variable, if you use one. In each node of a tree you get a separate multiple linear regression formula.

If no split is made (a tree with a single node), you get the Linear Regression algorithm. To simplify development, a separate Linear Regression algorithm is provided in SSAS. That way, you do not have to set the parameters of the Regression Trees algorithm to get it.

The only parameter for the Microsoft Decision Trees algorithm that pertains to the Regression Trees only is:

- **Force_Regressor parameter:** This forces the algorithm to use the indicated columns as regressors. An attribute used as a regressor is not used for splits; it is used for the multiple linear regression formula.

The Microsoft Linear Regression algorithm supports Forced_Regressor, Maximum_Input_Attributes, and Maximum_Output_Attributes parameters only.

To test the Regression Trees and Linear Regression, you have to add a new mining structure to the project you are developing in this chapter.

1. **In SSDT, open the project you created for testing the mining algorithms in this chapter.** Add a new mining structure.
2. **Select the Microsoft Decision Trees algorithm.** Use vTargetMail as the case table. Select NumberCarsOwned as the predictable attribute, and CommuteDistance, Age and YearlyIncome as input attributes. You do not need to create a test dataset.
3. **Set the FORCE_REGRESSOR parameter to YearlyIncome.**
4. **Using the same mining structure, add a new model using the Microsoft Linear Regression algorithm.** Note that the CommuteDistance attribute, which is discrete, has to be ignored for this algorithm.
5. **Save, deploy, and process the model (or the whole project).**

6. **Click the Mining Model Viewer tab.** Check the multiple linear regression formula for the NumberCarsOwned attribute in a couple of nodes of the Regression Tree, and the overall formula calculated by the Linear Regression model.

Microsoft Sequence Clustering

The Microsoft Sequence Clustering algorithm analyzes *sequence*-oriented data that contains discrete-valued series. Usually the sequence attribute in the series holds a set of events with a specific order (such as a click path). By analyzing the transition between states of the sequence, the algorithm can predict future states in related sequences.

A typical usage scenario for this algorithm is web customer analysis for a portal site. The Microsoft Sequence Clustering algorithm can group these web customers into more-or-less homogenous groups based on their navigation patterns. These groups can then be visualized, providing a detailed understanding of how customers are using the site. For example, you can use the Sequence Clustering algorithm for an advanced market basket analysis, where not only the products that are sold together are known, but also the order of putting them into the basket is known.

The most important parameters of the algorithm are:

- **Cluster_Count:** Specifies the number of clusters to be built by the algorithm.
- **Maximum_Sequence_States:** Specifies the maximum number of states that a sequence can have.
- **Maximum_States:** Specifies the maximum number of discrete states for a non-sequence attribute that the algorithm supports.

To test the Microsoft Sequence Clustering algorithm, follow these steps:

1. **Add a new mining structure to the project you are developing.** Select the Microsoft Sequence Clustering technique. Specify vAssocSeqOrders as the case table and vAssocSeqLineItems as the nested table. Use OrderNumber and LineNumber as key columns, and check that the LineNumber is also an input column. Select the Model column as Input and Predictable. Note that you select the same columns as for the Association Rules algorithm; however, you use them in a different way.
2. **You don't need a test dataset.** Finish the Data Mining Wizard and save and deploy the project.
3. **Use all five data mining viewers available for the Sequence Clustering algorithm to understand the itemsets and sequences you got.**

Forecasting with Microsoft Time Series

In a time series, you want to show one variable (e.g., sales quantity) as a function of time. Time series are used for forecasting. It helps answer questions such as "What's the sale amount of a product next year in a specific region?"

The Microsoft Time Series algorithm uses internally two algorithms: the Auto-Regression Trees with Cross-Prediction (ARTXP), and the Auto-Regressive Integrated Moving Average (ARIMA) algorithm. The ARTXP algorithm gives better short-term forecasts, while the ARIMA algorithm does better long-term forecasting.

Time series usually have *seasonal* patterns. Large cycles are also part of time series. The Microsoft Time Series algorithm adds historical data points based on the seasonality parameter (Periodicity_Hint). The Periodicity_Hint parameter accepts a set of values. You can use it to define the seasonality and cycles. For example, you might have monthly data. Then you could set the Periodicity_Hint parameter to {12, 48} to denote a seasonality of 12 months and a cycle of 4 years.

You can test the quality of the forecasts with historical forecasting. The idea of historical predictions is that if the algorithm predicted results in the past correctly, you can trust it more for future forecasts than the models that didn't predict past values correctly. In order to get the historical predictions, you do not have to train the model with all historical data points. You can build historical models based on the Historical_Model_Count and Historical_Model_Gap parameters. End-users see only a single continuous model.

The most important parameters of the Time Series algorithm are:

- **Historic_Model_Count:** Specifies the number of historic models that will be built.
- **Historical_Model_Gap:** Specifies the time lag between two consecutive historic models.
- **Periodicity_Hint:** Provides a hint to the algorithm as to the periodicity of the data. For example, if sales vary by year and the unit of measurement in the series is months, the periodicity is 12. This parameter takes the format of {n [, n]}, where n is any positive number. The n within the brackets [] is optional and can be repeated as frequently as needed.
- **Forecast_Method:** Specifies which algorithm to use for analysis and prediction. Possible values are ARTXP, ARIMA, or MIXED. The default is MIXED.
- **Prediction_Smoothing:** Specifies how the model should be mixed from ARTXP and ARIMA to optimize forecasting.

You can also test the Time Series algorithm with the following steps:

1. **Add a new mining structure to the project you are developing.** Select the Microsoft Time Series technique. Specify vTimeSeries as the case table.

Use TimeIndex and ModelRegion as key columns. You will forecast the quantity and the amount of sales of different models in different regions. Select the Amount column as Input and Predictable. Name the structure and the model and exit the Data Mining Wizard.

2. **Save, deploy, and process the model (or the whole project).**
3. **Select the Charts tab. Select the M200 Europe:Amount, M200 North America: Amount, and M200 Pacific:Amount check boxes.** Click the Show Deviations check box. Read the forecast for the selected models from the chart. Figure 9-10 shows the forecasts.

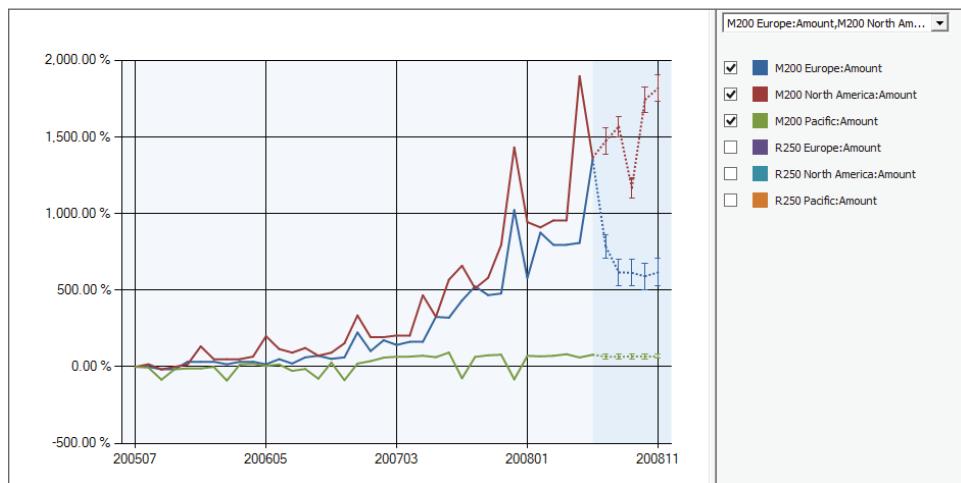


Figure 9-10: Time Series forecasts

Developing and Deploying a Scalable and Extensible Data Mining Solution

After you created your data mining model, you need to select one or more for the production. You need to test the models' efficiency. For example, for the predictive models, you test the quality of the predictions. You also need to decide whether to mine an SSAS cube or the relational data warehouse. Finally, there are some tasks that you cannot resolve through a graphical user interface—you need to create the *Data Mining Extensions (DMX)* language manually.

Choosing Between a Relational or a Cube Source for Your Data Mining Structure

You can also mine OLAP cubes. You use data from OLAP cubes for training a mining model. If you mine OLAP cubes, you already have:

- Well-formed and clean data
- Aggregations, which let you mine on higher levels of hierarchies

However, the data is read-only and thus not flexible. For example, the Time Series algorithm has very strict prerequisites for the data. It needs a numeric time key column, no period laps, etc.

A dimension can serve as the case-level table. You can add measures from an associated fact table as columns to the case-level dimension. You can use other dimensions related to the same fact table as nested tables.

Deploying Data Mining Models

The most important decision for the deployment of the mining models you created is which model to deploy. You need to test the efficiency of the models. The Mining Accuracy Chart view in the SSDT and SSMS uses the concept of a *lift chart* to test the predictive models. Usually when you create mining models, you set aside a subset of the data for testing purposes. Because the test data already contains values for the predictable attribute, you can create predictions and compare what the model finds with known values. You create a lift chart by plotting the results of these prediction queries against known values.

You can display the results of the prediction queries with two additional lines: one that represents the ideal model and one that represents a random guess. An ideal model line creates perfect predictions—the predictions are never wrong. The random guess line depicts what would happen if you had no mining model and you guessed the result of the predictable column. Any improvement over the random line is called lift. The more lift, the more effective the model is.

For example, consider the case in which the members of the Marketing department at AdventureWorks want to create a targeted mailing campaign. From past campaigns, they know that a 10% response rate is typical. They have a list of 10,000 potential customers stored in a table in the database, which means that they can expect 1,000 customers to respond. In addition, consider that the budget for the project is less than the amount of money needed to reach every customer in the database. Out of the 10,000 potential customers, they can mail an advertisement only to 5,000. The marketing staff members have two choices:

- Randomly select 5,000 customers to target.
- Use a mining model to target the 5,000 most likely customers.

If marketing randomly select 5,000 customers, they can expect to reach only 500 of the estimated 1,000 positive responses (because only 10 percent typically respond). This scenario is what the random line in the lift chart represents. If the Marketing staff members use a mining model, they can expect a higher response rate because they can target the most likely customers. If the model was perfect, they could expect to reach all 1,000 of the estimated responses by mailing to 1,000 potential customers recommended by the model. This scenario is represented by the ideal line in the lift chart. In reality, the mining model most likely falls between these two extremes and any improvement in response from the random guess is considered to be lift.

In the next example, you are going to test the predictive models you created in this chapter.

- 1. In SSDT, open the project you created for this chapter.** Open the data mining structure with the Clustering, Decision Trees, Naïve Bayes, Neural Network, and Logistic Regression models. Click the Mining Accuracy Chart tab.
- 2. Make sure that the Synchronize Prediction Columns and Values check box is checked.** Select 1 for the Predict Value column for all mining models. In the Select dataset to be used for Accuracy Chart option group, select the Use mining structure test cases option.
- 3. View the chart.** Select the Lift Chart tab to view the lift chart. In this case, the Decision Trees model should have the maximum lift from the random guess line, thus meaning that it performs the best, as Figure 9-11 shows.

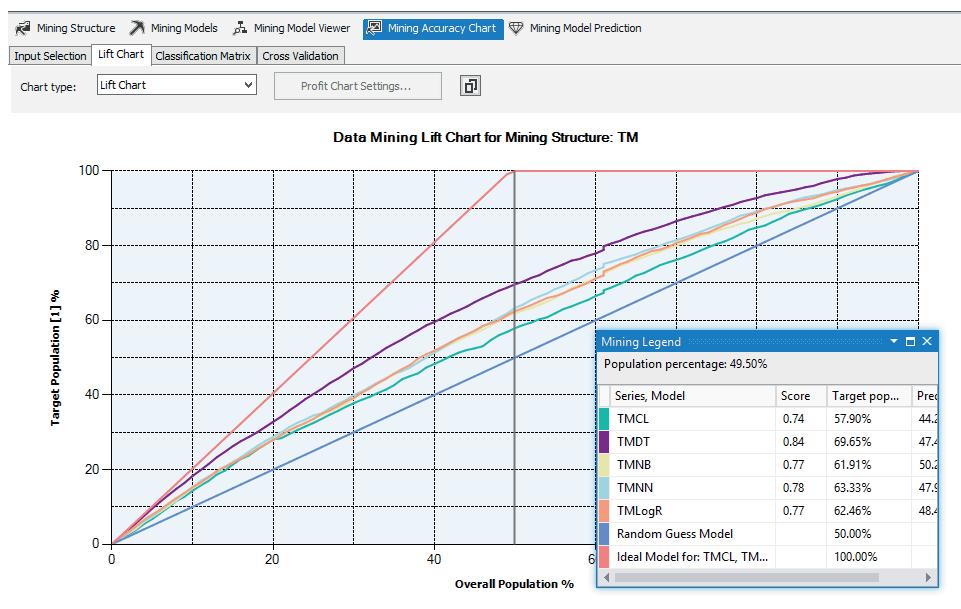


Figure 9-11: Lift Chart for the predictive models

Using DMX to Query Data Mining Models

Data Mining Extensions (DMX) is a language that you can use to create and work with data mining models in SSAS. You can use DMX to create the structure of new data mining models, to train these models, and to browse, manage, and predict against them. DMX is composed of Data Definition Language (DDL) statements, Data Manipulation Language (DML) statements, and functions and operators.

You can use DMX statements to create, process, delete, copy, browse, and predict against data mining models. There are two types of statements in DMX: data definition statements and data manipulation statements. You can use each type of statement to perform different kinds of tasks.

Data definition statements in DMX are part of DDL. You can perform the following tasks with the data definition statements in DMX:

- Create a mining structure and add a mining model to the structure.
- Create a mining model and associated mining structure simultaneously.
- Export a mining model and associated mining structure to a file, and import a mining model and associated mining structure from a file.
- Copy the structure of an existing mining model into a new model, and train it with the same data.
- Remove a mining model or a complete mining structure from a database.

You use data manipulation statements in DMX to work with existing mining models, to browse the models, and to create predictions against them. Data manipulation statements in DMX are part of DML. You can perform the following tasks with the data manipulation statements in DMX:

- Train a mining model.
- Remove all the trained data from a model or a structure.
- Change the node caption column in the data mining model (for nodes of the Clustering and Sequence Clustering models only).
- Use the SELECT statement to browse the information that is calculated during model training and stored in the data mining model, such as statistics of the source data.
- Create predictions that are based on an existing mining model by using the PREDICTION JOIN clause of the SELECT statement.

The SELECT statement is the basis for most DMX queries. Depending on the clauses that you use with such statements, you can browse, copy, or predict against mining models. The prediction query uses a form of SELECT to create predictions based on existing mining models. Functions extend your ability to browse and query the mining models beyond the intrinsic capabilities of the data mining model.

Maintaining Data Mining Models

Mining models' content becomes obsolete over time, as the patterns and rules might change. Therefore, you need to maintain the models over time. In order to realize when you need to change something, you need to measure the efficiency of the models, for example the quality of the predictions, over time.

Fine-Tuning the Data Mining Structure

The first action you should take when you notice that the efficiency of your models has dropped is to reprocess the data mining structure that contains these models. You don't need to reprocess the structure with new data as frequently as you need to process the DW and the SSAS cubes, which you typically process every day. The patterns of the mining models do not change so often. Based on testing the efficiency, you can generate a separate schedule for reprocessing the mining structures, for example, once per month.

Fine-tuning the structure means changing the attributes and changing or updating the dataset used to train the models. This simply means that you need to go back to the data overview and data preparation stages.

Keeping the Data Model Relevant

Similarly, like the data mining structure, you also need to keep the data mining models relevant. This means that you also need to fine-tune the models by using the new input attributes from the structure, if you added them to the structure, and maybe ignoring some existing ones. In addition, you can change some of the models' parameters.

Continuous Learning Cycle

To properly maintain your mining model, you need to implement the continuous learning cycle. What steps exactly you implement in the cycle depends on the business problem you are solving. Figure 9-12 shows an example for a continuous learning cycle for fraud detection.

You start by creating the directed models, assuming that the customer has already flagged frauds in the existing data. You evaluate the directed models and then use the best one to predict the frauds in the new data. You also create the undirected models, evaluate them, and use the best one for selection of potential frauds. You do this over time and check the difference between the number or percentage of frauds caught with the directed and the undirected model deployed. When this difference drops, it is time to refine the directed model. In addition, you store the predictions of both models and the actual,

confirmed or reported frauds in a data warehouse. When the percentage of the predicted frauds in the total number of frauds drops, it is time to refine both models. You can use an OLAP cube on the top of the DW to measure the efficiency of the models over time.

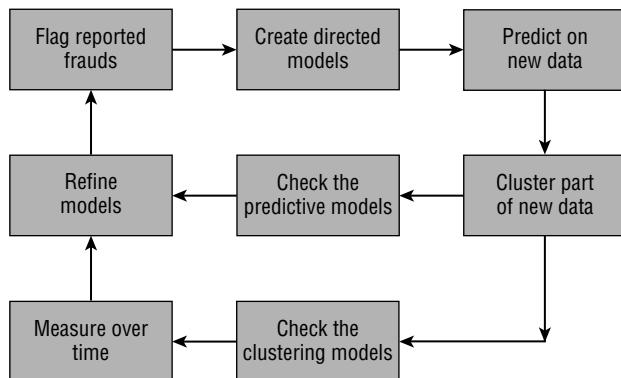


Figure 9-12: Fraud detection continuous learning cycle

Integrating Data Mining with Your BI Solution

Deployment of a mining model in production can also mean using it in your enterprise BI solution. You can integrate mining models in your SSIS ETL process, in a DW and SSAS cubes, and in SSRS reports.

Integrating Data Mining in Your DW and ETL Processes

Integrating data mining in SSIS is done through a couple of tasks and transformations.

The Data Mining Query task runs prediction queries based on data mining models built in SSAS. The prediction query creates a prediction for new data by using mining models. The query is a Data Mining Extensions (DMX) statement. The task can query multiple mining models that are built on the same mining structure.

The Analysis Services Processing task processes SSAS objects such as cubes, dimensions, and mining models. Multiple objects can be processed at the same time. When processing multiple objects, you define settings that apply to the processing of all the objects in the batch.

The Analysis Services Execute DDL task runs Data Definition Language (DDL) statements that can create, drop, or alter mining models and multidimensional objects such as cubes and dimensions. It uses an Analysis Services connection manager to connect to an SSAS instance. If the DDL code is stored in a separate

file, the Analysis Services Execute DDL task uses a File connection manager to specify the path of the file.

The Execute SQL task can execute DMX statements.

The Data Mining Query transformation performs prediction queries against data mining models. This transformation contains a query builder for creating Data Mining Extensions (DMX) queries. The query builder lets you create custom statements for evaluating the transformation input data against an existing mining model using the DMX language. One transformation can execute multiple prediction queries if the models are built on the same data mining structure.

The Data Mining Model Training destination pushes data from the pipeline to SSAS. You can also create a new mining structure because the Data Mining Wizard is incorporated into the destination.

You can integrate data mining with OLAP cubes in the opposite direction as well. You can deploy data mining models as dimensions. If you mine a BISM multidimensional cube and use a Clustering, Decision Trees, or Association Rules algorithm, you can deploy a mining model as a dimension in a very simple manner, directly from SSDT.

Remember that you need to measure the efficiency of a deployed mining model over time. OLAP cubes typically include a time dimension. OLAP cubes with additional data mining dimensions are a very natural way for measuring the efficiency of the models over time.

Integrating Data Mining with Reporting Services

You can create a report based on a data mining model directly, without having deployed a data mining model as a dimension. In such a case, you use the Data Mining Extensions (DMX) language. Report Designer includes the DMX Prediction Query Builder. DMX queries can be parameterized as well. You can also write DMX queries manually.

However, note that a report dataset is a table with columns and rows. It is two-dimensional and does not support nested tables. Your DMX query must return a flattened result. The DMX Prediction Query Builder automatically takes care of this issue. If you write a DMX query manually, you have to make sure that the result of your query is flattened.

Data Mining in Excel

The SQL Server Data Mining Add-ins for Office bring the power of data mining to desktop applications. The add-ins include:

- **Data Mining Client for Excel:** Enables you to go through the entire lifecycle of a data mining project, including preparing data, building, evaluating and managing mining models, and predicting results using either spreadsheet data or external data accessible through your SSAS database.

- **Table Analysis Tools for Excel:** Allows you to analyze your spreadsheet data in powerful ways with just a few mouse clicks. It employs SQL Server data mining behind the scenes.
- **Data Mining Templates for Visio:** Lets you render, annotate, and share your mining model patterns as Visio diagrams.

Note that Excel does not become a data mining engine with these add-ins. To train a model, Excel needs a connection to SSAS installed in the multidimensional and data mining mode. However, the data for data mining comes from an Excel worksheet, either organized as a cell range or as a table.

Summary

Data mining is the most powerful part of a business intelligence solution. However, implementing it requires more knowledge than you need for other parts of a business intelligence solution. After reading this chapter, you should understand the benefits of introducing data mining in your enterprise business intelligence solution, and also have the knowledge needed for the actual implementation.



Business Intelligence for Reporting

In This Part

- Chapter 10:** Choosing the Right Business Intelligence Visualization Tool
- Chapter 11:** Designing Operational Reports with Reporting Services
- Chapter 12:** Visualizing Your Data Interactively with Power View
- Chapter 13:** Exploring Geographic and Temporal Data with Power Map
- Chapter 14:** Monitoring Your Business with PerformancePoint Services

Choosing the Right Business Intelligence Visualization Tool

Microsoft has a variety of visualization tools within its business intelligence offering, but which one should you pick? Deciding between tools can be a daunting task, but one that can be approached very methodically if you know what to do. It is important to factor in your organization, options of tools, and business and technical priorities. Following the plan described in this chapter will ensure you have the right tool for the job.

This chapter helps you to choose the right business intelligence tool for your organization. By following a standard process, you can ensure that you incorporated all necessary requirements for a business intelligence visualization tool. To start, you will read about the importance of picking the right business intelligence tool. Then, you learn how to gather these requirements and incorporate them into an evaluation matrix for assessment. You also learn about some of the visualization tools available to you to include in the matrix. Finally, you complete the matrix and come up with a solution that fits your organization.

Why Do You Need to Choose?

A business intelligence *visualization tool* enables a person to see information in a way that allows them to make a good business decision. Also known as *performing analysis* or *gaining insights*, the visualization tool highlights something

important, such as a trend or an outlier. Microsoft provides many visualization tools, many of which you already can access or easily download.

VISUALIZATION

A **visualization** is a way to see data in a report or view that allows someone to gather insights and perform analyses to make better business decisions.

While tempting to let your developers and end users use any of the many tools available to them, you will quickly run into issues you may not have realized. Having a standard business intelligence tool helps your organization and its employees in many ways. This section digs into some of the reasons to choose a business intelligence visualization tool.

Identifying Users

To begin, you must recognize the number of people in your organization affected when someone uses a business intelligence visualization tool.

- **End users who run the reports:** This is the most obvious group. They want advanced features and accessibility of the tool, and need their data quickly and accurately.
- **Report developers:** They create the reports for the end users and need a tool that allows them to create those reports efficiently and accurately.
- **Support staff:** These people ensure that the reports are recovered in any sort of disaster scenario.
- **Management:** These people are concerned with the bottom line, including the number of users, developers, and staff needed to support the tool, and the cost of the tool itself.

The needs that become apparent from listing out the user base include:

- Development time
- Downtime
- Number of resources
- Price

The more tools that your organization uses and supports will most likely increase all these factors. Due to these varying needs, you should strive to reduce the number of visualization tools that your organization uses.

Selecting Tools

Now that you understand the need to reduce the number of visualization tools, and hopefully pick one of them, you can focus on the actual process to do this. The five core steps are shown in Figure 10-1. Essentially, you:

1. Start by understanding the criteria important to your organization.
2. Interview your key stakeholders (end users, developers, support staff, and management) to identify the priorities and additional criteria.
3. Identify and investigate the business intelligence tools available.
4. Fill out an evaluation matrix that records all the information you just learned.
5. Wrap up the process by verifying the results. The rest of the chapter explains each of these steps in more detail.

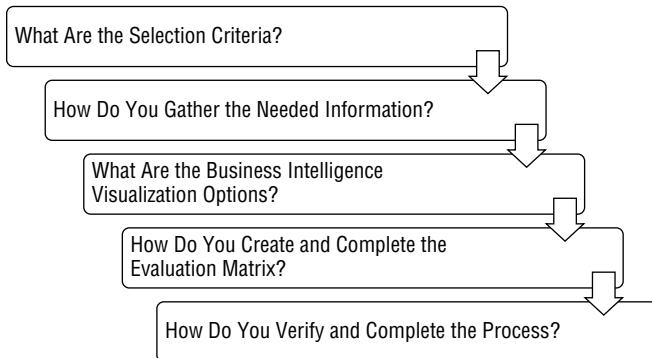


Figure 10-1: Tool selection process

What Are the Selection Criteria?

The first step in the visualization tool selection process is to understand each selection criterion and identify the importance to your department or organization. Each selection criterion highlights an aspect of the visualization tool that could make it a good or bad fit, so identifying all the criteria ensures you select the right tool. This section lists and describes several options that you should consider, but ultimately you must decide which criteria to include or which additional items you should add.

The selection criteria falls under two categories: business versus technical capabilities. The business capabilities include factors that a business analyst or end user needs to perform their duties, and the technical capabilities include

factors that developers or the information technology department needs to satisfy their requirements.

Be sure to describe each capability in terms that make the most sense to your environment, so that everyone is on the same page. The high-level business criteria includes: ease of use, functionality, and advanced features. The high-level technology criteria includes: ease of development, ease of maintenance, and cost. The following sections discuss each criteria set in more detail.

Business Capabilities

Although often overlooked, the business users' requirements are pivotal in picking the right visualization tool. The wrong tool can make it quite difficult for a user to complete the task at hand, which may result in additional work for either the user or the IT department.

Some of the requirements that the business might have include:

- Standard report creation
- Ad-hoc report creation
- Drillthrough/down data ability
- Overall ease of use
- Functionality (such as visualizations)
- Advanced features (analytics)

Technical Capabilities

Just as important as the business considerations is the technical side of the organization. You must make sure that the organization can support the tool as more users adopt it and additional visualizations for it become available.

The requirements to consider from the technical side include:

- Ease of development
- Ease of maintenance
- Price
- Security
- Customization
- Mobile capability
- Data integration

How Do You Gather the Necessary Information?

Now that you have a good understanding of the type of selection criteria you must create, you can gather the information needed to identify additional selection criteria, to prioritize the information, and to understand existing skillsets. To gather the information, you must look at existing documentation and talk to the organization's people to find out their key concerns. It is important to follow this process instead of assuming you know the answers intuitively.

You should look at several types of documentation within the organization for the first part of this exercise. Identify and read any standards documentation for software, servers, and editions. If possible, read any long-term planning or strategy documents for the company and the information technology department. Look for any information that falls into your selection criteria or highlights a new criterion that you should add.

The next step involves interviewing people from each of your stakeholder groups. The stakeholder groups include your end users, developers, support staff, and management. Some of the questions you may want to ask include, but are not limited to:

- **End user:** How comfortable are you with designing your own reports?
- **End user:** What types of graphs and charts do you need?
- **Developers:** What is your technical skillset?
- **Developers:** How much time do you have to dedicate to reporting?
- **Support staff:** What kind of downtime is allowable for a report?
- **Management:** What budget is available for reporting needs?

After you have gathered the necessary organizational information, you can move onto learning about the available visualization tools.

What Are the Business Intelligence Visualization Options?

The next step in the selection process is understanding the business intelligence visualization options available to you in the Microsoft stack. The three tools discussed in this section are Reporting Services, Power View, and Power Map.

Using SQL Server Reporting Services

Initially provided as an add-on to SQL Server 2000, SQL Server Reporting Services (Reporting Services) was the start of Microsoft's venture into reporting.

Reporting Services, now part of the SQL Server business intelligence suite, allows developers to create enterprise-level reports. In recent years, the tool has gained a greater following and added more advanced features. This section provides a brief overview of Reporting Services development, publishing, consumption, and requirements. For more detailed information on Reporting Services, see Chapter 11.

Understanding Development Tools

Reporting Services has two development tools, depending on the type of user. IT developers can use specific business intelligence templates installed in Visual Studio IDE. As of SQL Server 2014, you can install the templates through SQL Server Data Tools—Business Intelligence. Power users on the business side can use Report Builder, a one-click application accessed through the consumption interface.

When developing reports through either tool, you can use a variety of data sources, including databases, OLAP cubes, and files. Reporting Services displays the data pulled from the data source per row, and groups, transforms, or aggregates each data element. Additionally, you can add graphs, charts, and maps to enhance the usability of the report. Figure 10-2 shows a view of the designer.

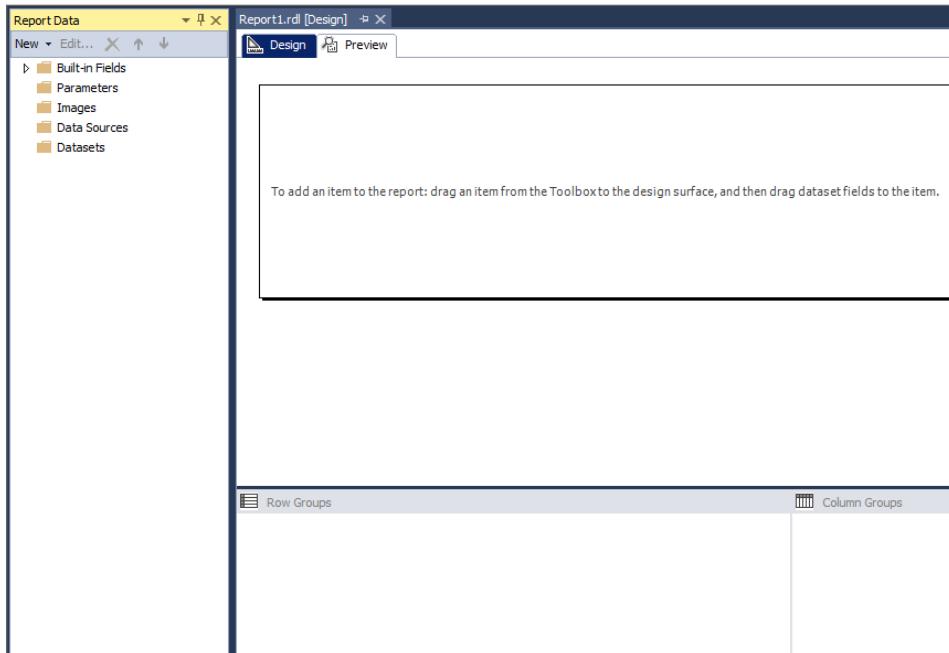


Figure 10-2: Reporting Services designer

Publishing and Consuming Reports

After developing the report, you can upload it to a server for end users to access it. Depending on the configuration mode (either SharePoint Integrated or Native), you will use either a SharePoint site or Report Manager to upload the reports. From either server, the end user can access the report directly, or a developer can embed the report in an external application. Additionally, you can create subscriptions to publish the report on a scheduled basis by sending an email or putting the report on a file share.

Because Reporting Services has a flexible user interface, end users can adjust parameters to modify the data shown in the report. They also can use drilldown functionality through collapsible/expandable groups, and drillthrough functionality through actions within the report that can link to another report or a section in the current report. The drilldown and drillthrough functionality must be developed prior to publishing the report. You can also filter the charts, graphs, maps, and data within the report based on the parameter selection. Figure 10-3 shows a sample Reporting Services report.



Figure 10-3: Sample Reporting Services report

Understanding Requirements

All editions of SQL Server include Reporting Services. As you move up the chain of editions (from Express, through Standard, up to Enterprise), the number of functions increase. If you use SharePoint Integrated mode, you need a SharePoint server and license. Also, include the Reporting Services server in any backup and maintenance services your organization has in place.

Using Power View

Power View, introduced with SQL Server 2012, provides end users with an ad-hoc data exploration visualization tool. By utilizing its tight integration to the Power Pivot semantic model, users drag and drop elements from a business-focused model to the design pane in an Excel-like tool. By creating their own visualizations, users can quickly identify trends and outliers to satisfy their analytics requirements. This section explains how you can use Power View to design and share reports with others and the requirements needed to use Power View. For information on how to design in Power View, see Chapter 12.

Designing with Power View

Power View's designer, meant for end users and business analysts, is modeled after a tool familiar to those folks: Excel! You design for Power View in SharePoint, Office 365, or Excel. For this chapter, you will learn about the Power View in Excel, as shown in Figure 10-4. In Power View, each section is called a *view* and is shown on a different tab in an Excel workbook. Figure 10-4 shows one *view*. The information for the view is a Power Pivot or SSAS tabular model, saved directly in your Excel workbook.

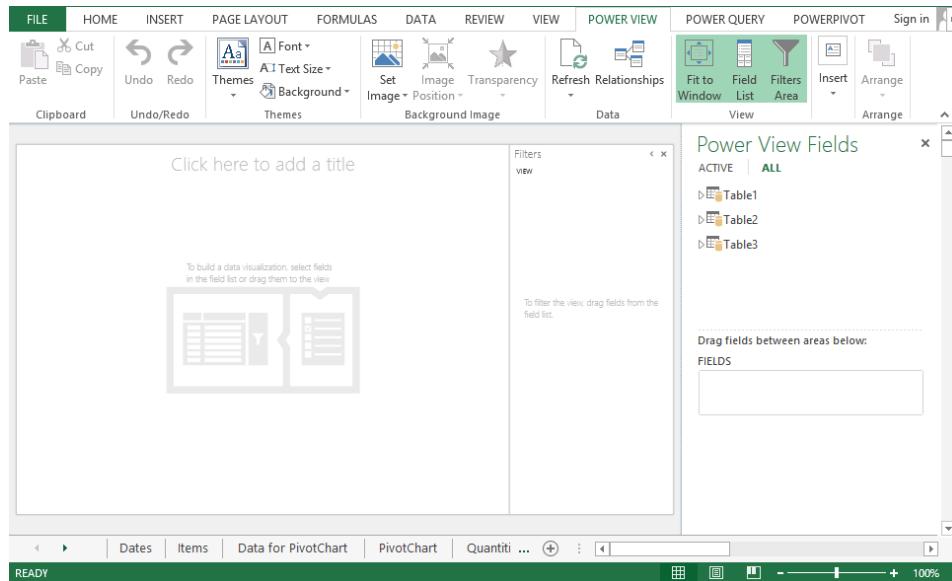


Figure 10-4: Power View designer

One of the core tenets of Power View is to provide a minimal click approach, so the designer does a lot of work for you. The end users can drag and drop

data elements from the model to the design pane and choose how they want the information to appear. Power View does a great job of associating the correct elements to the correct place in a visualization, but if the designer makes a mistake, the user can correct it.

Another great thing about Power View is the interaction among the page elements, which includes features such as filtering, highlighting, and animated bubble charts. Power View includes this interactivity out of the box, so it requires no additional development time from the visualization side. Figure 10-5 shows a view in Power View that has a category highlighted.

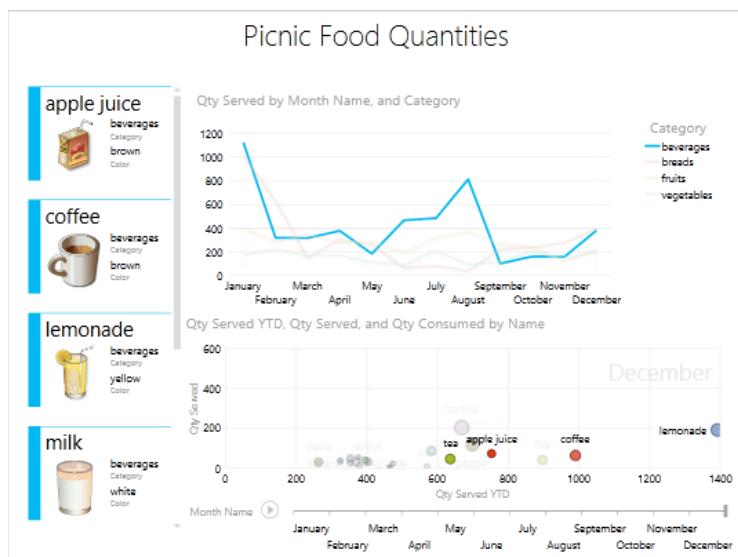


Figure 10-5: Sample view in Power View

Understanding Power View Requirements

Power View is an Enterprise feature in SQL Server and Excel 2013. You also need an Enterprise SharePoint Server edition to publish the Excel views and access the data within Power View on SharePoint. Finally, you need to have a tabular model or Excel Power Pivot workbook as a data source.

Using Power Map

Power Map, the newest addition to Microsoft's business intelligence visualization stack, allows users to tie their data to a geographical view, also called a map. Additionally, Power Map supports custom maps, such as bus routes and shopping mall areas. Similar to Power View, Power Map uses a Power Pivot semantic model for its data source and provides a similar Excel-like interface.

to drag and drop elements as needed to the map view. This section provides an overview of how to design and use tours in Power Map. For additional information, see Chapter 13.

Designing with Power Map

Meant for end-user and business-analyst usage, you can design Power Map tours in SharePoint, Office 365, or Excel. As of the writing of this book, Power Map is available as a Preview version in Excel. This chapter focuses on the Excel version of Power Map; you can see the designer in Figure 10-6. The data and map information combine into an object called a *tour*, which runs multiple scenes over time or as a static view. The information for the scene comes from a Power Pivot model, saved directly in the Excel workbook in which you are designing.

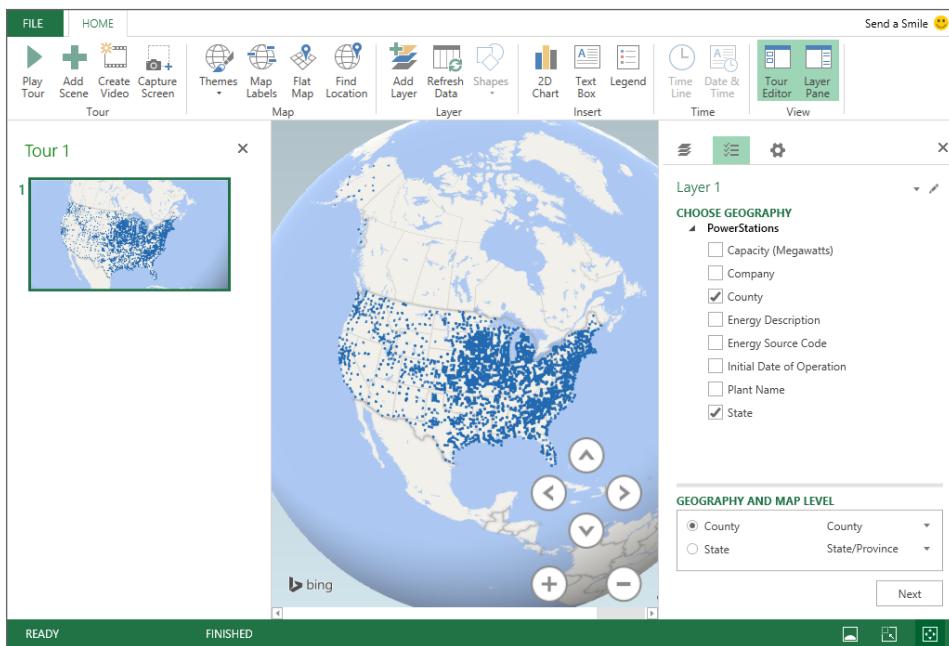


Figure 10-6: Power Map designer

Power Map provides multiple design layers to create the scene: The layers include the following:

- Geographical information, provided from the data as a latitude/longitude combination, an address, or variations of an address, such as city and state
- Data mapped to the address, such as your additive information or grouping categories
- Display data

Power View uses Bing Maps to translate the geographical information to map the tour information. You can modify this layer to different views as well. Figure 10-7 shows a sample of a tour in Power Map.

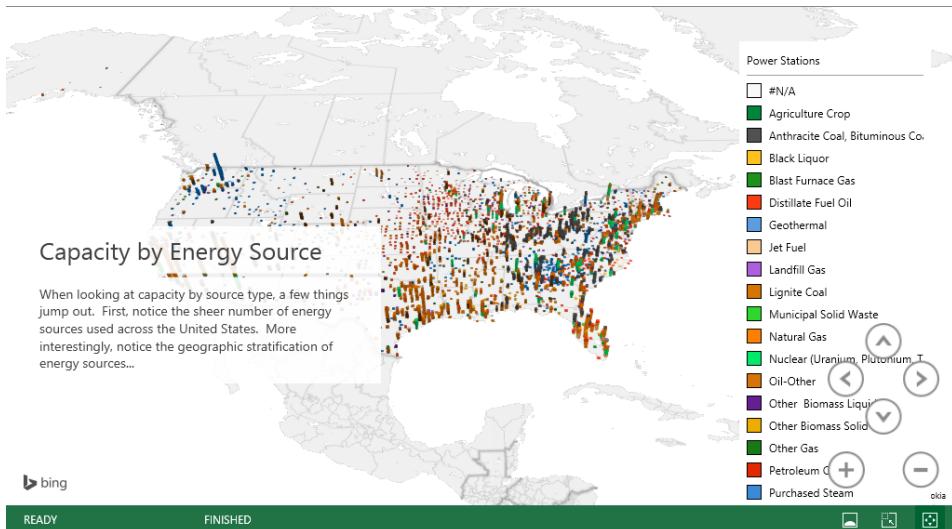


Figure 10-7: Sample tour in Power Map

Understanding Power Map Requirements

Power Map is a feature in SQL Server and Excel Professional Plus 2013 or Excel Standalone 2013. Alternatively, you can create a Power Map tour in SharePoint or Office 365. As previously mentioned, a tour uses an Excel Power Pivot workbook as its data source. Finally, if you use the Excel Power Map, you need to have Internet connectivity.

Next, you will learn how to utilize your tool knowledge in creating an evaluation matrix.

How Do You Create and Complete the Evaluation Matrix?

After you understand the business intelligence visualization selection criteria and are aware of the different tools available, you can create and complete the evaluation matrix. Figure 10-8 shows a sample matrix, although the numbers have no meaning as of yet. To complete the matrix, you need to take the following steps: Finalize selection criteria, prioritize criteria, and evaluate criteria scores. This section takes you through each of these steps.

Organizational Priority	16%	16%	18%	10%	10%	10%	10%	100%
Visualization Tool	Business			Technical				Score
	Ease of Use	Functionality	Advanced Features	Ease of Development	Ease of Maintenance	Price	Ability to Customize	
Reporting Services	1	1	1	1	1	1	1	1
Power View	3	3	3	3	3	3	3	3
Power Map	5	5	5	5	5	5	5	5

Figure 10-8: Completed evaluation matrix

To finalize the selection criteria, review the interview notes and background information you have gathered from your organization; ideally they should be prioritized by this time. Although the sample matrix in Figure 10-8 groups Advanced Features into one bucket, you may want to split out the ability to design maps and the ability to create movable bubble charts into two criteria. In another scenario, you may decide that the Ability to Customize criteria in the sample matrix does not apply to your organization. Remove any columns from the matrix that fall under that classification, and add any new criteria in another column. The important thing is to make the matrix reflect the needs and goals of your organization.

After you have finalized the selection criteria, you can prioritize them by giving them each a “piece of the pie.” Along the top of the matrix, in the Organizational Priority row, you see a set of percentages that add up to one hundred percent; you assign a percentage to each of the criterion listed below the matrix. Pay attention to the Business versus Technical grouping, because you don’t want to give too much priority to one over the other—or at least, you want it to reflect your organization accurately. Once completed, the final box should show a total of 100 percent.

NOTE Although this chapter focuses on the Microsoft business intelligence visualization tools, you could easily add non-Microsoft tools to this evaluation matrix to perform a cross-vendor selection. However, if you decide to do this, you may want to add some selection criteria focused around the stability and support associated with the other vendor.

Finally, you can begin scoring your tools based on the criteria. Although you might think that scores should be universal, they are actually quite dependent on your organization. For example, if your nonprofit organization gets discounts for licensing, the “Price” criteria might not apply. On the other hand, if your for-profit organization does not currently have a SharePoint server, the price for Power View and Power Map could be high. Based on your interviews and knowledge of the organization and of the tools, evaluate how each criteria applies to your specific organization. Give each criteria/tool combination a score of 1–5, where 1 is worst (such as expensive, high development time, hard to use) and 5 is the best (such as cheap or cost-negligible, low development time, easiest to use).

Once you have filled out all the scores, you should see a total score at the end of each row from 1–5. Don’t jump to a decision just yet, however; you still must verify the results, as described in the next section.

How Do You Verify and Complete the Process?

The final steps in the selection process are to verify the proposed solution, adjust it if needed, and arrive at a decision. Based on your evaluation matrix, each tool has a score of 1–5. Oftentimes, two or more tools can be very close to the highest value, so how can you ensure you have actually picked the best one? Although the methodology to score each tool is straightforward, you should look at the results with an unbiased eye to vet your answer.

Start by comparing the combined scores. Is the final result a surprise to you? If so, look at each of the individual selection criterion scores. Should any change? Finally, look at the organizational priority. Should you increase or decrease a few of the values? If so, readjust the numbers and see if the final value changes. Don’t get to where you have to change them multiple times to get the result you want. The idea here is to verify that the values you originally determined still make sense.

Unquestionably, this evaluation matrix is fairly subjective. If possible, give a blank evaluation matrix (without the organizational priority or the selection criteria scores) to the members of the organization and ask them to fill it out. Even if they don’t pick exactly the same numbers as you did, does the highest-ranked tool match? If not, go back and look at the biggest discrepancies in your values to see where the disconnect lies and see if adjustments can be made.

Every organization should have a different set of numbers and values, but you may notice some common trends among organizations. The following two evaluation matrices show the different values and results that might occur for different types of organizations.

Evaluation Matrix #1

The first organization, a small to midsize organization, has recently decided to start reporting from their data warehouse. The shop currently uses Microsoft products, including an on-site SharePoint Server installation. The analysts are comfortable with the technology, but the developers have little time to develop new reports. The organization prioritized getting the information out as quickly as possible for the analysts to make new decisions. After performing the selection matrix, the organization agreed on the evaluation matrix shown in Figure 10-9. They decided to go with Power View for their future reporting needs.

Organizational Priority		20%	20%	5%	10%	15%	20%	0%	10%	100%
Visualization Tool	Business			Technical						Score
	Ease of Use	Functionality	Advanced Features	Ease of Development	Ease of Maintenance	Price	Ability to Customize	Data Integration		
Reporting Services	2	3	3	3	3	2	4	3	2.6	
Power View	4	4	3	4	2	2	1	3	3.15	
Power Map	4	1	2	4	2	2	1	3	2.5	

Figure 10-9: Evaluation matrix #1

Evaluation Matrix #2

The second organization, larger than the first, has already invested in some Reporting Services. They are a Microsoft shop, but do not yet have any SharePoint Servers available. Their developers have skills in Reporting Services, but their analysts do not have enough time to spend developing their own reports. The analysts want the information provided to them and need only limited interactivity. After completing the evaluation matrix shown in Figure 10-10, the second organization decided on Reporting Services for their main business intelligence visualization tool.

Although the goal of this exercise is to whittle down the tool options to one, your organizations may need two tools to satisfy their needs. As an example, if your organization must have rotating maps and also pixel-perfect reporting, you may need both Power Map and Reporting Services. Just pay attention to the other selection criteria to make sure you don't blow the budget or resource limits by choosing two options!

Organizational Priority	20%	10%	5%	15%	20%	10%	10%	10%	100%
Visualization Tool	Business			Technical					Score
	Ease of Use	Functionality	Advanced Features	Ease of Development	Ease of Maintenance	Price	Ability to Customize	Data Integration	
Reporting Services	2	3	3	3	3	4	4	3	3
Power View	4	4	3	4	2	1	1	3	2.85
Power Map	4	1	2	4	2	1	1	3	2.5

Figure 10-10: Evaluation matrix #2

Finally, run your decision by your stakeholders. Explain the process, each of the scores, and the recommendation. They may bring up a new point that you didn't consider that you will need to refactor into your decision. Don't worry if you missed something; the real goal is to come up with the tool that works for your organization in the long run.

Summary

In this chapter, you learned the process for choosing a business intelligence visualization tool. By now you should understand the importance of limiting the number of tools that your organization uses, how to identify what is important to your organization, and how to put together an evaluation matrix to make this important decision. The next few chapters explain in detail each of the business intelligence visualization tools highlighted in this chapter, starting with Reporting Services.

Designing Operational Reports with Reporting Services

Operational reports provide the necessary data to keep the business moving forward. Reporting Services allows you to create all kinds of reports, including operational. Combining the power of operational reporting and Reporting Services will help your company complete its day-to-day business activities.

This chapter introduces you to the world of operational reporting within SQL Server Reporting Services (Reporting Services) by discussing, in order, operational reports, Reporting Services, and then some of the best practices available to you for operational reports in Reporting Services. These best practices fall under several areas: development, performance, and functionality. Each of the best practices provides steps to let you create your own operational reports in Reporting Services.

What Are Operational Reports and Reporting Services?

Report Services developers tend to group reports into two categories: operational reports and analytical reports. Although the line between the two categories can be blurry, you can decide in which category your report falls based on a few criteria: source data, audience, and display. You need both types of reports to successfully run your organization; together, these two report types provide insight into both how your organization performs today and where it can go in the future.

Understanding Analytical Versus Operational Reports

Analytical reports intend to provide insights into your data that you wouldn't otherwise know about. Analytical report data tends to come from a data warehouse or an OLAP (online analytical processing) cube. Although not always the case, using a data warehouse as a source is more common because the data has already been gathered from multiple places into one, which makes the analysis easier on the reporting side. Analytical reports are typically written for business analysts or end users, the people who make business decisions and adjust processes based on the information in the report. Finally, an analytical report has certain design aspects, such as scorecards or aggregated information that allows end users to quickly see where they need to focus. For example, an analytical report that shows the sales per region associated with the weather over the past three years would allow a business analyst to identify which regions the weather most affects.

ANALYTICAL REPORT

An *analytical report* provides insight into the business based on past data.

OPERATIONAL REPORT

An *operational report* provides the necessary data to keep the business moving forward.

Operational reports perform a different role within the organization: to provide the information needed to perform a certain task. Operational report data tends to come from either the application database itself or an operational data store (ODS)/reporting database created specifically for this purpose. People involved in the day-to-day operations of your business, such as your infrastructure team or shipping manager, will get the most benefit out of operational reports. The most important design features for an operational report include clean lines—so the user can see the information they need immediately—the ability to drill down to more data if needed, and a way to link to the application system to see more information. Figure 11-1 contains a sample operational report.

Using Reporting Services

You can easily create operational reports using SQL Server Reporting Services. As previously discussed in Chapter 10, Microsoft started their foray into reporting

with the Reporting Services in the early 2000s. Although this chapter uses Reporting Services 2014 for its descriptions and examples, many of the examples apply to the earlier versions. Throughout its evolutions, Reporting Services has remained a good way to create static or dynamic reports.

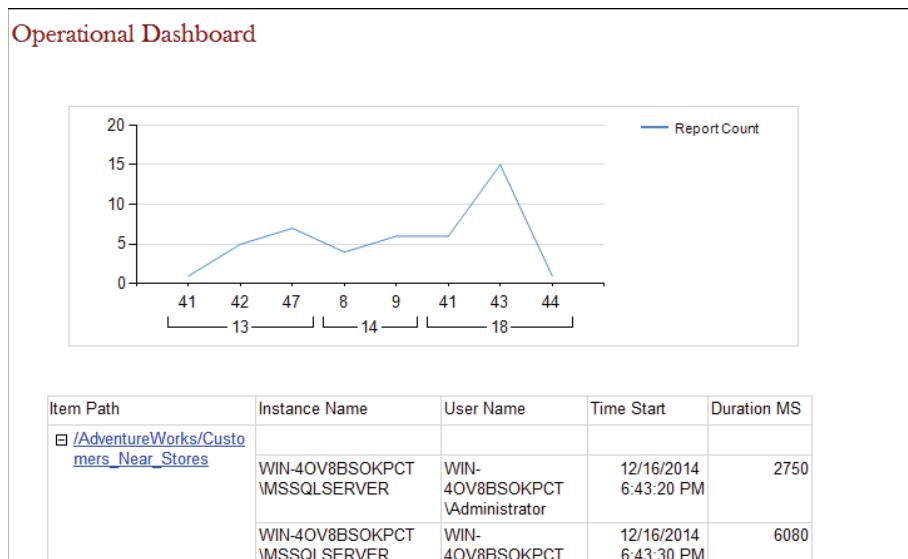


Figure 11-1: Sample operational report

The Reporting Services product includes both a server and client component. This section discusses the Reporting Services server and client architectures and explains the different components within the Reporting Services tool available for operational reporting.

The Reporting Services server component includes the Reporting Services engine, databases to store the information, and a web service layer to access the report information. The type of server depends on how you set it up, which can be one of two flavors: native mode or SharePoint mode. You should understand the different architectures available and in what mode your particular instance exists to ensure you develop correctly.

The Reporting Services client component, sometimes known as the development tool, also comes in two flavors. You can use either Report Builder or SQL Server Data Tools (SSDT). End users and analysts tend to use Report Builder 3.0 to create reports, and developers tend to use SSDT to create reports. Figure 11-2 shows the SSDT designer, which you'll see used in the rest of the chapter examples.

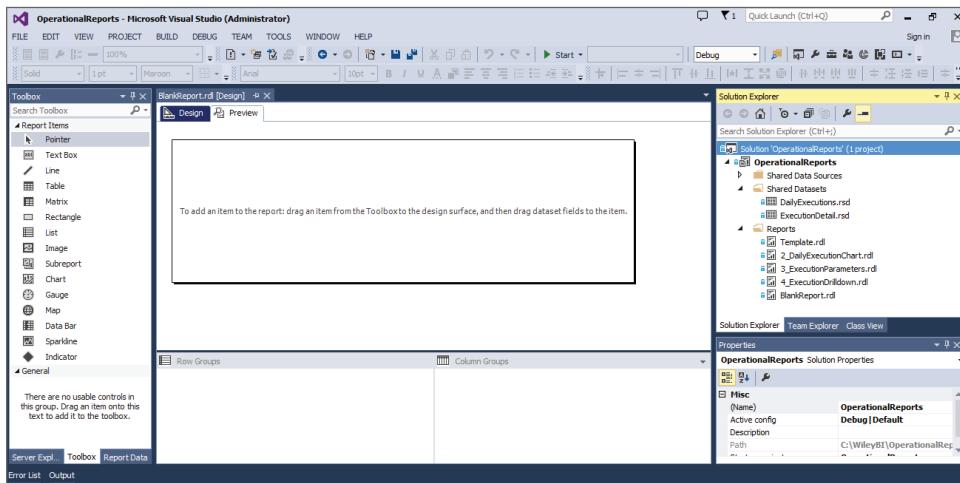


Figure 11-2: SQL Server Data Tool designer

Reporting Services allows you to pull data from a variety of sources and display that information on a report. The report items to display data are shown in the Toolbox in Figure 11-2. Some of the more common items are described here:

- **Textbox:** A textbox is a simple, but oh-so-powerful report item. It allows you to display static, dynamic, and formatted text.
- **Tablix:** A tablix is the first report item that most people use in a Reporting Services report and is still your go-to item. A tablix provides you with the ability to list data directly, embed charts, and sort or group the data.
- **Chart:** A chart displays data in a visual format. Although you may think of these items as being more for analytical reports, they can be a great benefit in operational reports as well!

The rest of this chapter walks through several examples of implementing sections of operational reports in Reporting Services. Best practices in developing operational reports center on a few areas: development, performance, and functionality. Development best practices center around creating the reports. Performance best practices are about getting the report to perform quickly. Functionality best practices include the ways to make the report respond in the best fashion. These best practices are described in more detail in the next few sections.

What Are Development Best Practices?

As a report developer, you want to develop your Reporting Services report both quickly and accurately. You want to ensure that your end users keep their

confidence in the final report that you provide to them. With certain projects, it can appear that speed and quality are inversely related. With reporting best practices, you want to reduce that inverse relationship to make the development easier.

Under development, you can follow several best practices: source and version control, shared data sources and datasets, and creating a template report to use for future development.

Using Source and Version Control

Behind the scenes, the reporting development environment, SQL Server Data Tools (SSDT), uses the Visual Studio application. Along with using Visual Studio comes all the benefits of a full integrated development environment (IDE), including plug-ins and automatic tie-in to source control. If your organization uses Team Foundation Server (TFS) to store its code, you can store your reports in a repository as well.

Storing your reports in a source control repository provides many benefits. You can keep track of changes that anyone on your team made to the report. This helps if you need to roll back to an earlier version of a report or even explain to the business what change happened. Additionally, if your server dies and you cannot restore the backups of the machine or code, you can redeploy the code to a server and be back up in no time! Finally, if a co-worker leaves the company or his or her personal computer dies before the reports have been uploaded to the production server, you have a way of getting the underlying code to modify and promote to production.

If you're using a web-based version of TFS and Visual Studio, you may already have some of these items in place. However, if you use a "plain jane" Visual Studio installation, follow these steps to connect to your TFS server after you have been given access to a team project by your TFS administrator:

1. **Download and install the Team Explorer for Visual Studio add-in from Microsoft.** For Visual Studio 2013, you can download the add-in from: <http://www.microsoft.com/en-us/download/details.aspx?id=40776>. You will know it has been installed correctly when you see the Team menu in Visual Studio.
2. **Click Team ↗ Connect to Team Foundation Server.**
3. **In the Team Explorer pane, click the Select Team Projects link.**
4. **In the Connect to Team Foundation Server window, select the Servers button.**
5. **Click the Add button.** Then enter your TFS server's information.
6. **Click OK.** This validates the connection information.

After you have completed these steps, you should have a Team Explorer pane that looks similar to Figure 11-3. Keep in mind that you don't have to use TFS for your source control server. Some of the other available tools include Git, CVS, and Subversion. As long as the other tool has a plug-in for Visual Studio, you can use it for your reports instead.

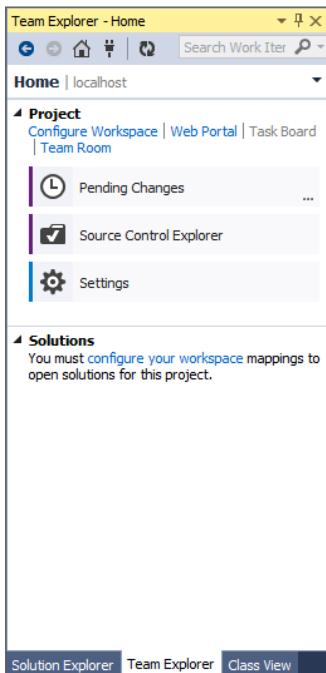


Figure 11-3: Team Explorer pane

After you have created a reporting solution that you want to put into source control, you can add it to TFS through the Solution Explorer. To do this, right-click the report project name, navigate to the Source Control menu header, and select the Add Solution to Source Control option, as shown in Figure 11-4.

Editing a report automatically checks it out; just be sure to check your report in with comments on a regular basis to receive the full benefit of the source and version control! To check the report out, check it in, or even view the change history, right-click on the desired file and navigate to the Source Control menu. You can see each of the options available to you in Figure 11-5.

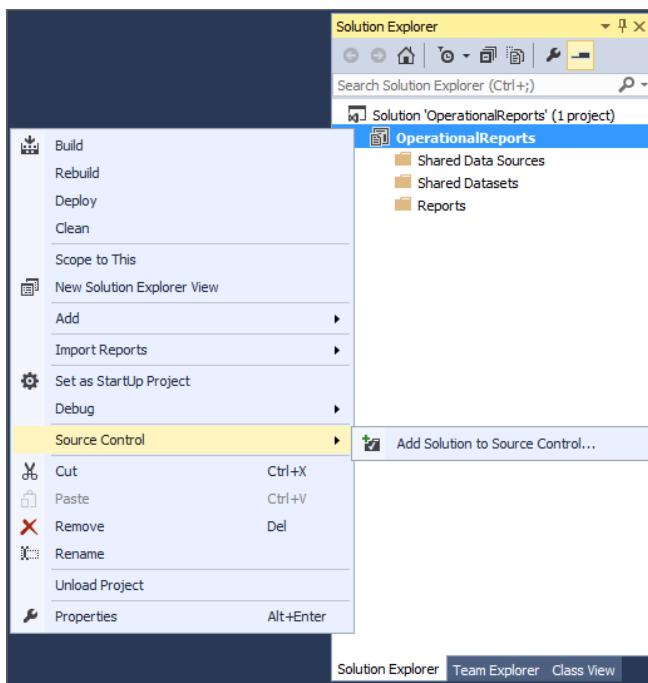


Figure 11-4: Add Solution to Source Control

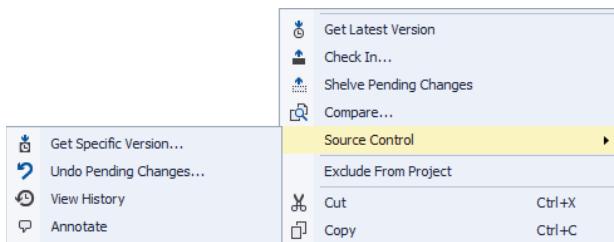


Figure 11-5: Source Control options

This section just touched the surface of what source and version control can do for you and how you can use TFS to manage your source and version control. For more advanced options and information for handling your application lifecycle management within TFS, see Books Online, <http://msdn.microsoft.com/en-us/library/fda2bad5.aspx>.

Using Shared Data Sources and Datasets

Now that you have backed up your work on the source control server, you can start creating your code. The number one best practice that the authors can share with you is to use a shared data source in the report. A shared data source takes the connection information out of the report and puts it in another object. This abstraction layer of the connection information means that you can move reports through environments without touching the meat of the report, which opens up the door to accidentally modify something you didn't mean to. Additionally, you can use a shared data source in more than one report.

Shared datasets go hand in hand with shared data sources, although the former have been around a lot less time than the latter. A shared dataset contains the query information that says what data will be returned to the report. Multiple reports can use the same shared dataset. If you ever need to change the business logic in a query or change the name of a stored procedure, a shared dataset is a great way to do this because you change it in one place for all linked reports to use.

To create a shared data source, follow these steps:

1. Right-click the **Shared Data Sources** folder in the Solution Explorer and click the **Add New Data Source** option. Fill out the appropriate connection information, in this case the ReportServer database, as shown in Figure 11-6. Click OK, and you have created your shared data source!

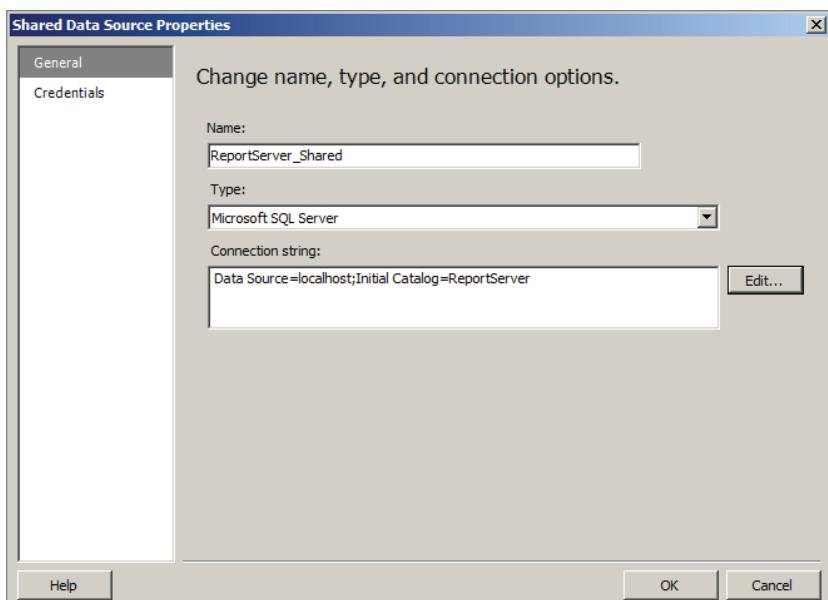


Figure 11-6: Shared Data Source Properties window

2. **Create a shared dataset using a similar process.** Right-click the Shared Datasets folder and select the Add New Dataset option. Rather than the connection information, you enter the data source and query that you want to pull data. The final shared dataset screen will look like Figure 11-7, if you use this query:

```
select DATEPART(HOUR, TimeStart) AS Hour  
      , DATEPART(MINUTE, TimeStart) AS Minute  
      , COUNT(*) as ReportCount  
  from ExecutionLog3  
 group by DATEPART(HOUR, TimeStart)  
      , DATEPART(MINUTE, TimeStart)
```

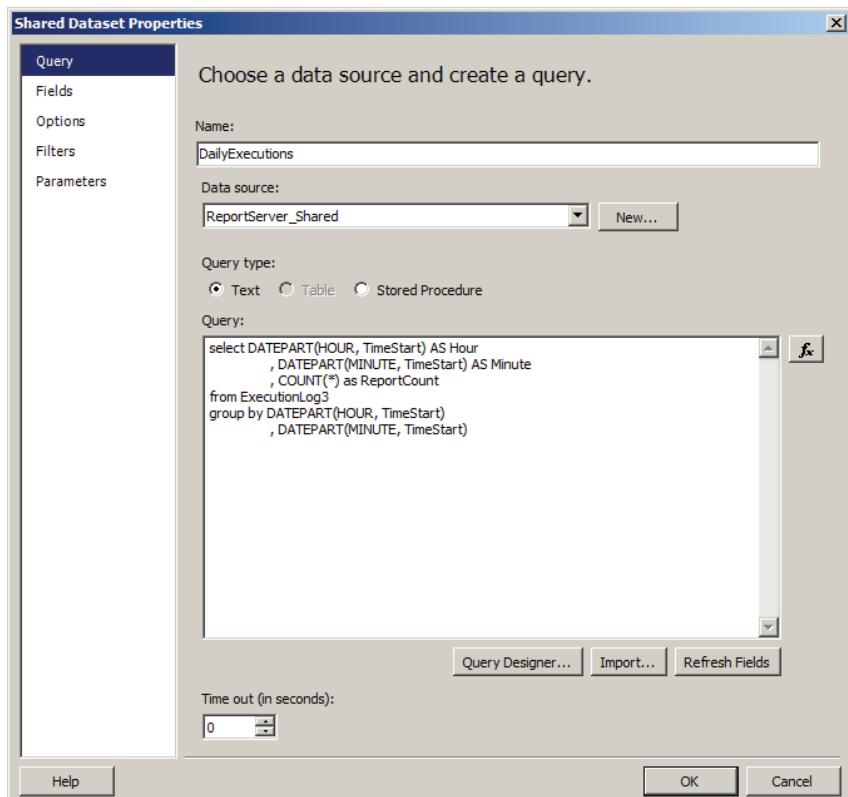


Figure 11-7: Shared Dataset Properties window

As you implement additional best practices in the rest of this chapter, you will learn how to use these shared items within your reports.

Creating Templates

The last best practice under the Development category is to create a template that you can use as a start for any of your reports. In Reporting Services, a template is a starting point for future reports; it is not a base object that you can modify in the future and have the changes reflect in all the reports that have been created with that template. That being said, you can quickly get started on a new report if you use a template.

A report template should contain the elements most common to all reports that you create. For example, if you tend to use a common data source for most of your reports, you could add the link to the shared data source in the report. Additionally, you can include a common header and footer used for all your reports, including a sample report title, any parameters the report used to filter its data, and the corporate color palette for all reports. Finally, you can include any other common items in the report, such as the date the report was run, common tables or graphs, or the format for the page numbers (see Figure 11-8).

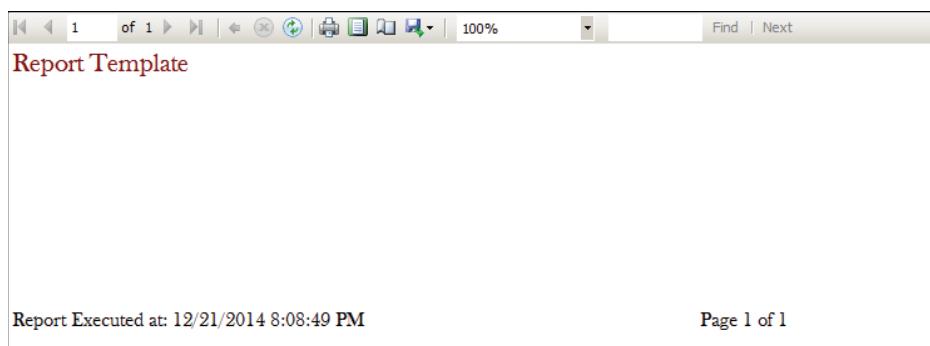


Figure 11-8: Sample report template

For this scenario, you will create the template report shown in Figure 11-8. If you don't see the header, right-click the beige space around the report and click the Add Report Header option and the Add Report Footer option. Because a header and footer repeat on every page, they are good places to put the report name and common elements. For this scenario, use the name "Report Template" on the header, and add the following code snippet to record the time the report was executed to the left of the footer:

```
= "Report Executed at: " & Globals!ExecutionTime
```

Add the following code snippet to list out the page number and total pages of the report to the right of the footer:

```
= "Page " & Globals!PageNumber & " of " & Globals!TotalPages
```

Change the font type and font color to match your corporate standard, add a logo, and you are ready to make a copy of the report to start developing!

What Are Performance Best Practices?

An important feature that can make or break an end user's report experience is the performance of the report. The performance includes how fast the initial report loads and how quickly an end user can interact with the report. The faster the report responds for end users, the happier they will be. This section describes how to find the performance issue and then how to improve the performance issue.

Investigating Performance

If you have a report performing slowly, you should look in the ReportServer database, where Reporting Services stores its internal information and metadata. The ReportServer database contains a set of views that help you learn about how many times each report has run, who ran the report, and how long the report took. As of Reporting Services 2014, the latest view is called ExecutionLog3. The main three columns that you need for performance investigation are:

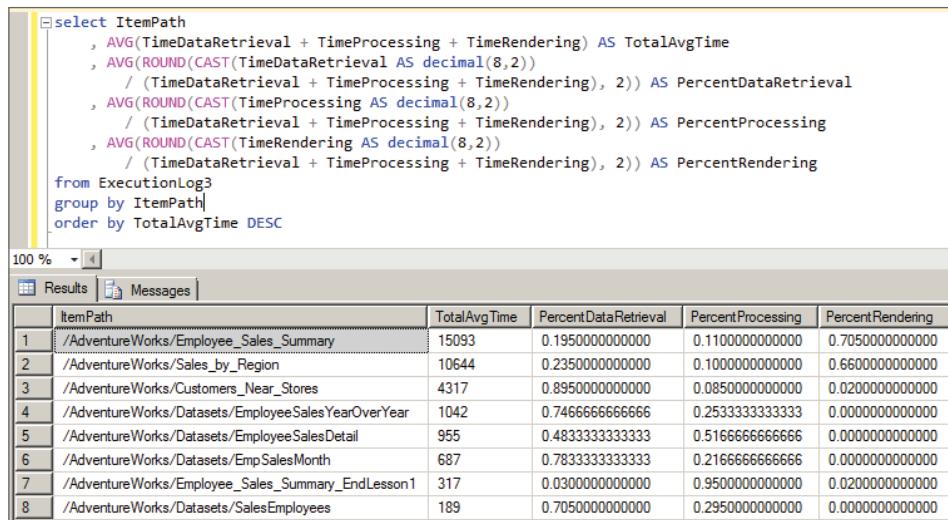
- **TimeDataRetrieval:** Contains the time spent retrieving the data from the source in milliseconds
- **TimeProcessing:** Contains the time spent processing the report so that it can be displayed in milliseconds
- **TimeRendering:** Contains the time spent rendering (displaying) the report in milliseconds

You can easily average these values per report and see the percentage of the total time by using the following query:

```
select ItemPath
      , AVG(TimeDataRetrieval + TimeProcessing + TimeRendering)
            AS TotalAvgTime
      , AVG(ROUND(CAST(TimeDataRetrieval AS decimal(8,2))
                  / (TimeDataRetrieval + TimeProcessing + TimeRendering), 2))
            AS PercentDataRetrieval
      , AVG(ROUND(CAST(TimeProcessing AS decimal(8,2))
                  / (TimeDataRetrieval + TimeProcessing + TimeRendering), 2))
            AS PercentProcessing
      , AVG(ROUND(CAST(TimeRendering AS decimal(8,2))
                  / (TimeDataRetrieval + TimeProcessing + TimeRendering), 2))
            AS PercentRendering
  from ExecutionLog3
```

```
group by ItemPath
order by TotalAvgTime DESC
```

For example, in the results of the same query shown in Figure 11-9, you can see that the report that takes the most time to return on average is the Employee_Sales_Summary report. Starting with this report for performance tuning will give you the largest bang for your buck.



The screenshot shows a SQL Server Management Studio window. The top pane contains a T-SQL query that calculates average execution times and their percentages for various report item paths. The bottom pane displays the results of this query, which are presented in a tabular format with five columns: ItemPath, TotalAvgTime, PercentDataRetrieval, PercentProcessing, and PercentRendering. The results show that the Employee_Sales_Summary report is the most time-consuming.

	ItemPath	TotalAvgTime	PercentDataRetrieval	PercentProcessing	PercentRendering
1	/AdventureWorks/Employee_Sales_Summary	15093	0.19500000000000	0.11000000000000	0.70500000000000
2	/AdventureWorks/Sales_by_Region	10644	0.23500000000000	0.10000000000000	0.66000000000000
3	/AdventureWorks/Customers_Near_Stores	4317	0.89500000000000	0.08500000000000	0.02000000000000
4	/AdventureWorks/Datasets/EmployeeSalesYearOverYear	1042	0.74666666666666	0.25333333333333	0.00000000000000
5	/AdventureWorks/Datasets/EmployeeSalesDetail	955	0.48333333333333	0.51666666666666	0.00000000000000
6	/AdventureWorks/Datasets/Emp SalesMonth	687	0.78333333333333	0.21666666666666	0.00000000000000
7	/AdventureWorks/Employee_Sales_Summary_EndLesson1	317	0.03000000000000	0.95000000000000	0.02000000000000
8	/AdventureWorks/Datasets/SalesEmployees	189	0.70500000000000	0.29500000000000	0.00000000000000

Figure 11-9: Performance Query results

Based on the highest value between PercentDataRetrieval, PercentProcessing, and PercentRendering, you know where you should focus your efforts. The next section provides some guidance on where to start with performance tuning each of these areas.

Performance Tuning

Once you know how long each portion of the report execution process takes, you can start the tuning process. If the PercentDataRetrieval is the highest value out of PercentDataRetrieval, PercentProcessing, and PercentRendering, you will spend your time with the data source of the report. If PercentProcessing is the highest value, you want to focus on the server. And if PercentRendering is the highest value, you want to focus on the items within the report.

For each issue, you want to follow a similar process:

1. **Isolate the offending stage.** If the data source connects to a SQL Server, try executing the query on the machine to ensure the time matches the data retrieval time. If this query runs significantly faster, check the connection

string to see if the connection uses a linked server or a server on a faraway domain.

2. **Benchmark the offending area.** Once you have determined the problem area, run the test to determine the duration several times to get an accurate time. For example, run the query from SQL Server, or, if the issue is the web service, call the web service with the same parameters as the report.
3. **Make a change in that area.** Speeding up each of these areas involves a different skillset. For example, speeding up a query written against SQL Server could include reengineering the query or adding new indexes. Reducing the processing time could mean evaluating the workload on the server at that time. You can even fix the rendering time by using caching or snapshots.
4. **Retime the offending area and check for a marked improvement.**

Iterate through the process until you are satisfied with the final result. For more information on Reporting Services performance tuning, see the Books Online page at <http://msdn.microsoft.com/en-us/library/bb522786.aspx>.

What Are Functionality Best Practices?

The final area for best practices that this chapter covers is functionality. Depending on the type of operational report, the report could need to display the needed information to feed a process or highlight any important elements that need action. The functionality discussed in the section—visualizations, filters, and parameters—assist with these functions.

Using Visualizations

Visualizations are powerful in both analytical and operational reporting. Whereas analytical visualizations show new insights that could modify how you do business, operational visualizations tend to condense a lot of information into a few objects to make it easier to see something. For example, for a server uptime report, you want to show how many of the servers are down and how long they have been down.

To create this visualization, you use a line graph to show how many reports were run throughout the day. This gives the report server administrator an idea of the peak usage times of the report server and helps balance the load if needed. Follow these steps to create the chart and add it to your report:

1. **Create a dataset.** You want to do this using the following query against the ReportServer database:

```
select DATEPART(HOUR, TimeStart) AS Hr  
      , DATEPART(MINUTE, TimeStart) AS Minute
```

```

    , COUNT(*) as ReportCount
from ExecutionLog3
group by DATEPART(HOUR, TimeStart)
    , DATEPART(MINUTE, TimeStart)

```

2. **Add a line chart to the report.** To do this, use the dataset you just created.
3. **Set up the following parameters for the line chart:**
 - **Values:** ReportCount
 - **Category Groups:** Hour, Minute
4. **Adjust any of the titles and legends to your liking.** Execute the report to see your rendered chart, similar to Figure 11-10.

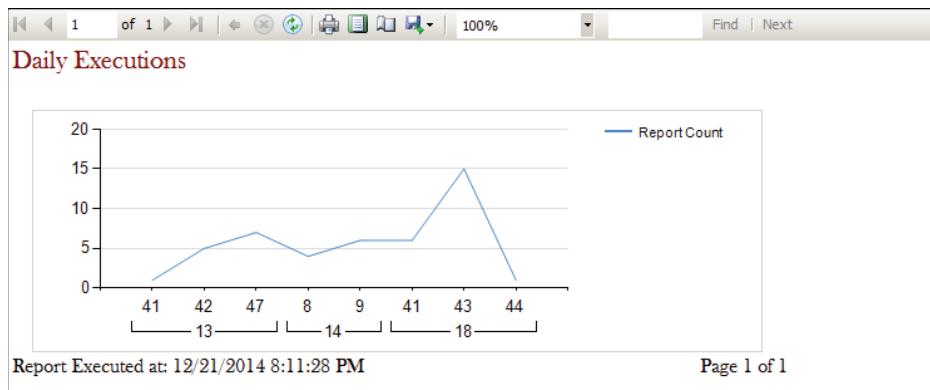


Figure 11-10: Sample visualization

Using Filters and Parameters

Report developers often make the common mistake of providing too much information. Although a business user commonly requests “everything,” you, as a report developer, must limit the noise and provide only the meaningful data. The first way to accomplish this is to identify correct parameters and filters that show only the appropriate data.

Depending on your organization and what department you write reports for, you may be able to identify common parameters for multiple reports’ usage. If so, include these in the template that you created previously. Most reports should have some sort of date parameter, or even multiple date parameters. Some variations of the date parameters include: start date/end date, year/month/day, and fiscal/calendar selector. Other common selectors include department, customer, and product. Think about your data: If you have any hierarchies or standard groupings, definitely include them as parameters. Additionally, you can cascade your parameters to reduce the data even further.

The important part of setting up the parameters is to filter the data from your dataset. You will now walk through setting up a report parameter and tying it to a query parameter to filter your data. Almost all reports have a date parameter (or set of date parameters), so that's what you will create in the following steps:

- 1. On the Report Data pane, create two report parameters with a data type of Date/Time.** Name them **StartDate** and **EndDate**.
- 2. Create a new dataset against the ReportServer database.** Use the following query, which uses two query parameters:

```
select ItemPath, InstanceName, UserName, TimeStart
      , DATEDIFF(MILLISECOND, TimeStart, TimeEnd) AS Duration_MS
  from ExecutionLog3
 where TimeStart between @start and @end
```

- 3. On the Dataset's Parameters pane, map the @start query parameter to the StartDate report parameter.** Also map the @end query parameter to the EndDate report parameter.
- 4. Add a table and display all the fields from the dataset.** Run the report to see the date parameters that you can enter. If you change the parameters, they filter the data displayed on the report. You can see the completed report in Figure 11-11.

Instance Name	Item Path	User Name	Time Start	Duration MS
WIN-4OV8BSOKPCT\\MSSQLSERVER	/AdventureWorks/Datasets/SalesEmployees	WIN-4OV8BSOKPC\\T\Administrator	12/16/2014 6:41:25 PM	2686
WIN-4OV8BSOKPCT\\MSSQLSERVER	/AdventureWorks/Employee_Sales_Summary	WIN-4OV8BSOKPC\\T\Administrator	12/16/2014 6:41:33 PM	30563
WIN-4OV8BSOKPCT\\MSSQLSERVER	/AdventureWorks/Datasets/EmployeeSalesYearOverYear	WIN-4OV8BSOKPC\\T\Administrator	12/16/2014 6:41:33 PM	30593

Figure 11-11: Report with parameters

Providing Drilldown and Drillthrough

An operational report must also let workers continue their duties as needed. You can easily accomplish this by providing a lower detail of data or even a drillthrough to a different set of information based on the data. The drilldown functionality uses an expandable and collapsible region, and the drillthrough functionality uses the actions feature in Reporting Services.

Creating a Drilldown Effect

To create the drilldown effect, you use the visibility option of a row or group and set the toggle to use a textbox. Follow these steps to create the drilldown effect on a report name to drill into the details of multiple executions:

1. **Follow the steps under the Filters and Parameters section to create a report.** The report should execute against the ReportServer database, use the same query, and display execution data.
2. **Right-click the Details row of the table and add a parent group.**
3. **Group on the ItemPath field.** Then, check the box to add a group header.
4. **Delete the original ItemPath column.**
5. **Right-click the Details row of the table.** Then select the Row Visibility option.
6. **Set the properties to initially hide the row and have the ItemPath1 textbox toggle the row.** Running the report creates the view shown in Figure 11-12, with the toggle available to the left of the report name. Clicking the toggle expands the selection to show each of the executions for that report, as shown in Figure 11-13.

Item Path	Instance Name	User Name	Time Start	Duration MS
<input checked="" type="checkbox"/> /AdventureWorks/Customer_Near_Stores				
<input checked="" type="checkbox"/> /AdventureWorks/Datasheets/EmployeeSalesDetail				

Figure 11-12: Collapsed drilldown table

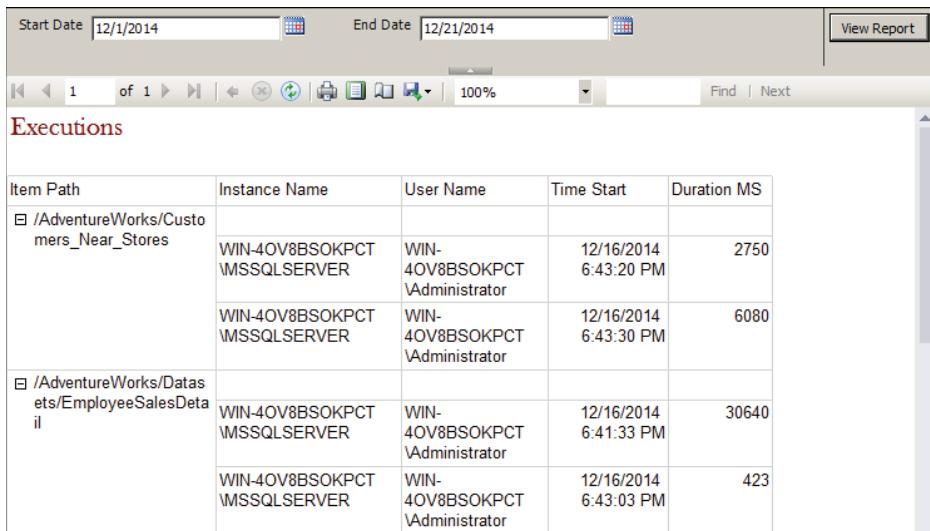
Creating a Drillthrough Effect

To create the drillthrough effect, you use a Reporting Services action. An action allows you to create a dynamic or static web link in the report. To ensure you have created the easiest report to use, you create a dynamic link to access a web page based on the report name table in this scenario. Follow these steps to create the action:

1. **Right-click the textbox that contains the ItemPath field.** Then, select the Text Box Properties option.

2. On the Action tab, select the Go To URL radio button.
3. Click the Expression button to the right of the dropdown.
4. In the Expression window, type:

```
= "http://myapplication?Param=" & Fields!ItemPath.Value
```
5. Format the font color of the text box to blue. Then underline the font to differentiate the text as a URL and execute the report.
6. Click the report name. This opens a web browser with the URL address, including the name of the report that you clicked.

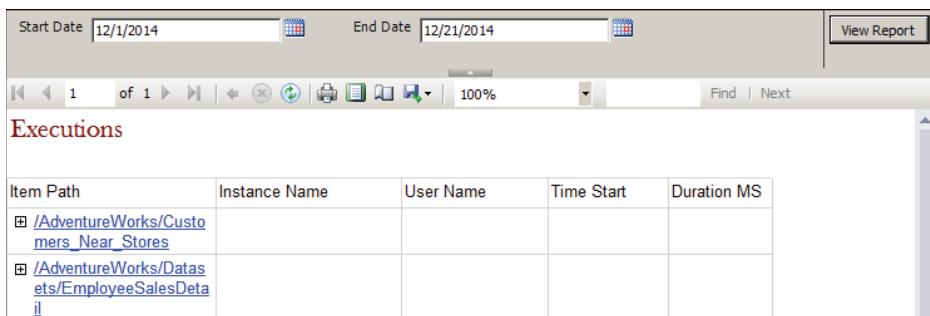


The screenshot shows a report interface with a toolbar at the top containing 'Start Date' (12/1/2014), 'End Date' (12/21/2014), 'View Report' button, and various navigation and search controls. Below the toolbar is a section titled 'Executions'. A table displays data with the following columns: Item Path, Instance Name, User Name, Time Start, and Duration MS. The data is grouped by Item Path, with two rows for each group. The first group is for '/AdventureWorks/Customers_Near_Stores' and the second for '/AdventureWorks/Datasets/EmployeeSalesDetail'. The table has a light gray background with alternating row colors.

Item Path	Instance Name	User Name	Time Start	Duration MS
/AdventureWorks/Customers_Near_Stores	WIN-4OV8BSOKPCT\\MSSQLSERVER	WIN-4OV8BSOKPCT\\Administrator	12/16/2014 6:43:20 PM	2750
	WIN-4OV8BSOKPCT\\MSSQLSERVER	WIN-4OV8BSOKPCT\\Administrator	12/16/2014 6:43:30 PM	6080
/AdventureWorks/Datasets/EmployeeSalesDetail	WIN-4OV8BSOKPCT\\MSSQLSERVER	WIN-4OV8BSOKPCT\\Administrator	12/16/2014 6:41:33 PM	30640
	WIN-4OV8BSOKPCT\\MSSQLSERVER	WIN-4OV8BSOKPCT\\Administrator	12/16/2014 6:43:03 PM	423

Figure 11-13: Expanded drilldown table

Figure 11-14 shows the view of the executed report.



The screenshot shows a report interface with a toolbar at the top containing 'Start Date' (12/1/2014), 'End Date' (12/21/2014), 'View Report' button, and various navigation and search controls. Below the toolbar is a section titled 'Executions'. A table displays data with the following columns: Item Path, Instance Name, User Name, Time Start, and Duration MS. The data is grouped by Item Path, with two rows for each group. The first group is for '/AdventureWorks/Customers_Near_Stores' and the second for '/AdventureWorks/Datasets/EmployeeSalesDetail'. The table has a light gray background with alternating row colors. The 'Item Path' column contains hyperlinks that are underlined and colored blue.

Item Path	Instance Name	User Name	Time Start	Duration MS
/AdventureWorks/Customer_Near_Stores				
/AdventureWorks/Datasets/EmployeeSalesDetail				

Figure 11-14: Table actions

Summary

In this chapter, you learned about designing operational reports in Reporting Services, and that an operational report assists in the continued, regular operations of your business. To satisfy this goal, you can use a variety of features within Reporting Services, such as charts, actions, and performance monitoring.

The following chapter looks at how best to use Power View to visualize your analytical data.

Visualizing Your Data Interactively with Power View

With the release of SQL Server 2012 came a new Microsoft reporting tool called Power View, codenamed Project Crescent. Power View is different from the existing reporting tools made available by Microsoft. It won't take you long to realize that Power View is designed to be quick and easy yet colorful, visual, and dynamic. Creating dynamic and interactive reports with Power View is simply a matter of clicking, dragging, and dropping. Microsoft purposely designed Power View to be as simple as possible, so simple in fact that at this time there is no place in the tool to write code, add expressions, or use any kind of formula.

Power View is the perfect reporting tool for the end user who may not necessarily be technically savvy. With its simplicity, Power View begs the user to explore the data using its interactive visualizations. These impressive visualizations and Power View's simple-to-use interface make Power View the ideal tool for ad hoc reporting, data exploration, and interactive presentations.

This chapter delves into the features and abilities of Power View and discusses how you and your organization can utilize one of the newest Microsoft reporting technologies. By the end of this chapter, you will have gained a clearer understanding of the capabilities of Power View and will have learned how to create Power View reports in SharePoint as well as Excel.

Where Does Power View Fit with Your Reporting Solution?

Because of the tool's simplicity, Power View is a great data exploration and ad hoc reporting tool. It's very possible to develop a robust and interactive report within a matter of seconds after opening the Power View report designer. One of the strengths of Power View is the ability to change data visualizations on the fly. Imagine you're viewing a clustered column chart, but you think you could gain better insights into your data if you viewed the data as a line chart. Changing the column chart to a line chart is just a single click away. The ability to quickly develop reports with multiple data regions featuring different data visualizations makes Power View the perfect tool for exploring your data and querying your data model on the fly.

Power View requires a well-developed data model supporting the reports. For users to have the best experience and get the best use of Power View, data models supporting Power View should have the correct metadata definitions in place and follow user-friendly naming conventions. Power View limitations prevent users from changing dimension, table, and field names, for example. With a well-defined data model, you can enable some very powerful features in Power View, such as support for images, geographic data types, tool tip descriptions, and more.

Power View reports can be created using Excel 2013, which makes it an excellent choice for data exploration and ad hoc reporting. Chances are most of your users are comfortable with Excel and will feel right at home when creating their first Power View report in Excel.

Power View reports can also be developed and viewed in SharePoint. By using a Power Pivot Gallery in SharePoint, users can easily view existing Power View reports within your web browser. Power View reports can also be developed using a Power Pivot model or a SQL Server Analysis Services database as the data source, also within the web browser.

Power View is the ad hoc data exploration tool your users have been wanting and waiting for. With the ability to explore, manipulate, and visualize your data with just a few clicks, Power View is a great tool to allow your users to quickly gain insights into the success of the business.

Power View System Requirements

Power View in Excel requires that you install Excel 2013 with Office Professional Plus 2013. Microsoft Silverlight 5 must also be installed. To use Power View in SharePoint Server, the following system requirements must be met:

- SQL Server 2012 Enterprise or Business Intelligence Edition with Service Pack 1 or SQL Server 2014 Enterprise or Business Intelligence Edition

- Microsoft SharePoint Server 2010 or 2013 Enterprise Edition
- SQL Server Reporting Services in SharePoint Integrated Mode
- Microsoft Silverlight 5

Creating Power View Data Source Connections

One of the advantages of using Excel to author your Power View reports is that you have the ability to create connections to multiple data sources within a single workbook. Creating a connection to your data sources is easy but is also the first step to unleashing the capabilities of Power View. In this section, you'll walk through creating a connection to SSAS so you can begin developing your Power View reports.

Creating Data Sources Inside Excel

Within Excel, you have the ability to create Power View reports based on a Power Pivot data model, SSAS tabular model, or any data that simply exists in a table within the same Excel workbook.

To create a connection to a tabular model, follow these steps:

1. **In Excel, navigate to the Data ribbon.**
2. **Select From Other Sources.** Then select From Analysis Services, as seen in Figure 12-1.

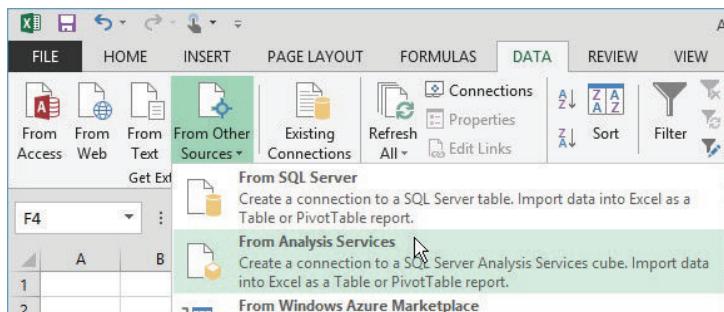


Figure 12-1: Creating a connection to SSAS

3. **In the Data Connection Wizard, enter the name of the SSAS instance you want to connect to.** Click Next.
4. **Select the database you want to use.** Highlight the appropriate cube or model and click Next.
5. **Specify the Data Connection File Name and click Finish.**

6. In the Import Data window, select the radio button next to Power View Report. Click OK. (See Figure 12-2.)

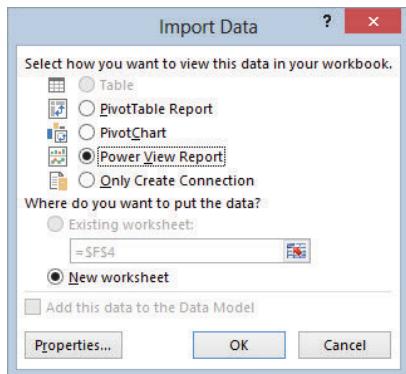


Figure 12-2: Importing data into a Power View report

To create a Power View report based on data that exists within a table in an Excel workbook, follow these steps:

1. Open the Excel workbook that contains the data you want to use.
2. Highlight a cell in the table containing your data.
3. Navigate to Excel's Insert ribbon. Click the Power View button. This creates a Power View sheet in the Excel workbook, as shown in Figure 12-3.

SalesAmount	CalendarYear	MonthNumber	ProductSubcategoryName
39,360.00	26098	85	Bike Racks
39,591.00	26098	85	Bike Stands
56,798.19	26098	85	Bottles and Cages
19,688.10	26098	85	Caps
7,218.60	26098	85	Cleaners
46,619.58	26098	85	Fenders
35,020.70	26098	85	Gloves
225,335.60	26098	85	Helmets
40,307.67	26098	85	Hydration Packs
172,950.68	26098	85	Jerseys
9,952,759.56	72234	234	Mountain Bikes
14,520,584.04	72234	234	Road Bikes
71,319.81	26098	85	Shorts
5,106.32	26098	85	Socks
245,529.32	26098	85	Tires and Tubes
2,014,001.00	72234	70	Touring Bikes

Figure 12-3: Using an Excel table in Power View

Creating Data Sources Inside SharePoint

In SharePoint 2013, Power View reports can use an SSAS multidimensional cube, SSAS tabular model, or a Power Pivot model deployed to SharePoint. To create a Power View report based on an SSAS multidimensional cube or SSAS tabular model, you must create a shared Report Data Source Connection (.rsds) in a SharePoint document library. To create a Power View report using a Power Pivot model as a data source, the Power Pivot model should be deployed to a SharePoint document library.

Creating a Shared Report Data Source

To create the .rsds file, you must first add the Report Data Source content type to a SharePoint document library or in a Power Pivot Gallery. The following example adds the Report Data Source content type to the Power Pivot Gallery.

1. **Click the Library ribbon.** Click Library Settings.
2. **Scroll down to find the Content Types section.** Click Add from existing site content types.
3. **Select Business Intelligence from the Select site content type from drop-down box.** Highlight Report Data Source and click the Add button. Then click OK.

Now you're ready to create a new Report Data Source file. The .rsds file contains the connection string for the data source. Before you can create the Report Data Source, you need to know:

- The SSAS server name
- The SSAS database name
- The SSAS cube name if more than one cube exists within the database and if you are connecting to an SSAS multidimensional instance

To create a new .rsds file, follow these steps:

1. **Navigate to the SharePoint document library where you will create your .rsds file.** Click the Files ribbon, select New Document, and click Report Data Source.
2. **Specify the Name of the .rsds file in the Name text box.**
3. **In the Data Source Type drop-down list, select Microsoft BI Semantic Model for Power View.**

4. In the Connection string text box, enter the connection string. This should include the Data Source, Initial Catalog, and Cube properties if you are creating a connection to an SSAS multidimensional cube. If you are creating a connection to an SSAS tabular model, the Cube property of the connection string is not required because an SSAS tabular database contains only one model.
5. Next to Credentials, leave the default selection of Windows authentication (integrated) or SharePoint user.
6. Click Test Connection to ensure you can access the data source. Click OK. (See Figure 12-4).

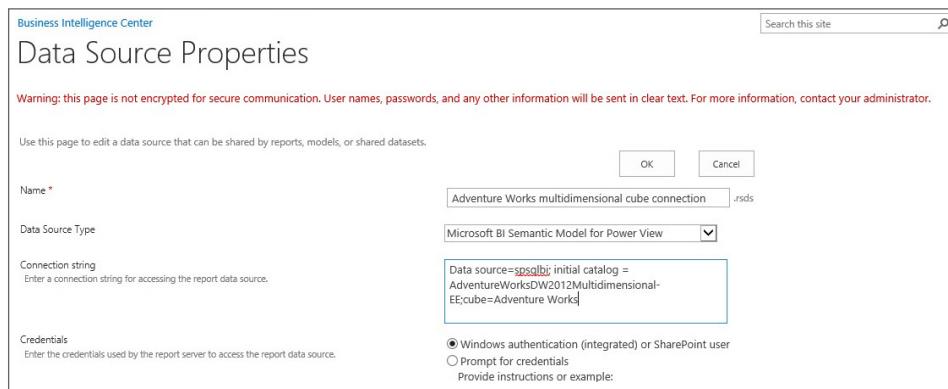


Figure 12-4: Shared Report Data Source properties

Once the new .rsds file is created, a user needs only to double-click it to open a new blank Power View report.

Deploying a Power Pivot Report to SharePoint for Use by Power View

To create a Power View using a Power Pivot model as a data source, you must deploy the Power Pivot model to a SharePoint document library or a Power Pivot Gallery. The following example uploads an Excel workbook containing a Power Pivot model to the Power Pivot Gallery.

1. Navigate to the Power Pivot Gallery you will use for the Power Pivot model. Select the Files ribbon, and click Upload Document.
2. Click the Browse button. Navigate to the Excel workbook you want to use. Click Open and then click OK. After uploading the Excel workbook, the properties window of the workbook should appear.

3. **Change the Content Type to Report Data Source.** Modify the Name and Title if necessary. Click Save and close the window.

To begin developing a Power View report using the Power Pivot model, navigate to the Power Pivot model in the Power Pivot Gallery and click the Power View icon, as shown in Figure 12-5.

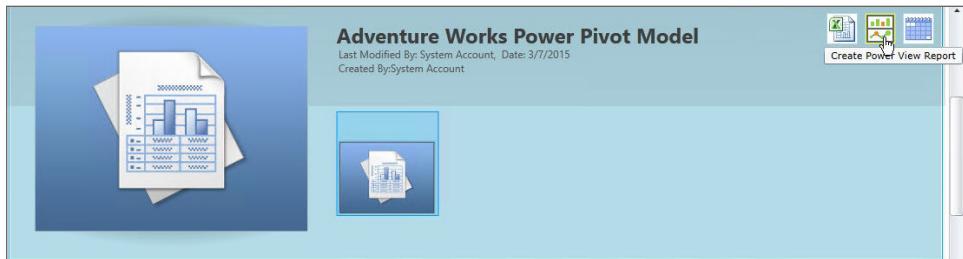


Figure 12-5: Creating a Power View report using Power Pivot as a data source

Creating Power View Reports

Compared to other reporting technologies, Power View is one of the easiest to use. Because Power View was designed with the end user in mind, creating dynamic, interactive, and useful reports is simple and straightforward. Chances are the biggest challenge you'll face is deciding on the type of report to build. In this section, you'll step through creating Power View reports in both Excel and SharePoint.

Using SharePoint to Create Power View Reports

The development of Power View reports for SharePoint all takes place within your web browser in SharePoint. To create a Power View report in SharePoint using a Power Pivot model as a data source, navigate to the Power Pivot model in the Power Pivot Gallery and click the Create Power View Report icon.

To create a Power View report by using a shared report data source (.rsds) file, double-click the .rsds file to automatically open the Power View report designer.

The steps required to create a report visualization in Power View using SharePoint are very similar to creating a Power View report in Excel. In SharePoint, however, the report visualization types are accessible from the Quick Access menu in the Design ribbon, as shown in Figure 12-6.

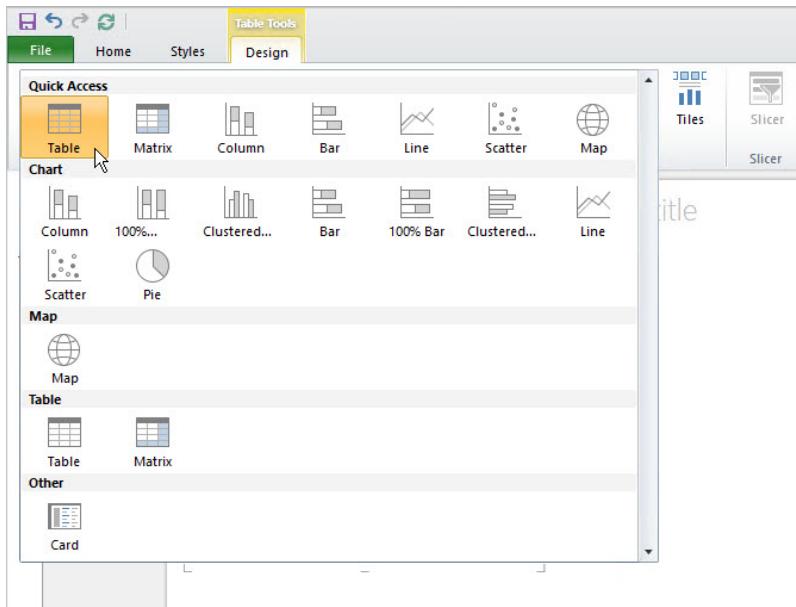


Figure 12-6: Switching report visualization types

Once you've finished developing your Power View report, save your report by clicking the Save icon so that other members of your team can view and/or edit the reports from the Power Pivot Gallery in SharePoint.

Using Multiple Views in Power View

Power View reports created in SharePoint can contain multiple views. Each view in a Power View report can contain its own visualizations and filters. The visualizations and filters within an individual view are independent of any other view included as part of the Power View report.

A new view can be created as a blank view or as a duplicate of an existing view. To create a new, blank view in your Power View report, navigate to the Home ribbon and click the New View button. To create a copy of an existing view, first navigate to the view you want to duplicate. Then on the Home ribbon, click the drop-down arrow on the New View button and select Duplicate View (see Figure 12-7).

NOTE Consider creating Power View reports with multiple views when designing reports for similar business processes and areas of your organization. Duplicate views are also great for creating visualizations similar to each other featuring only minor differences to enhance reporting capabilities for your team.

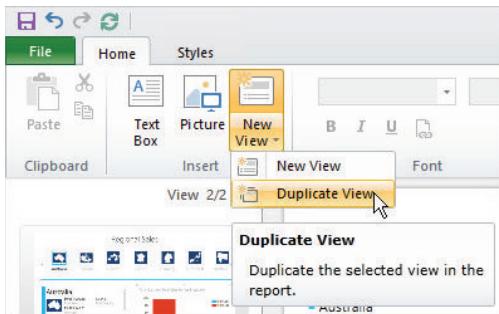


Figure 12-7: Creating new views in Power View

Creating Power View Visualizations

As previously mentioned, Power View is a tool designed to allow end users to quickly and easily explore their data and perform ad hoc analysis of their organization's metrics. Creating interactive, intuitive, and dynamic visualizations in Power View is simple and requires only a handful of clicks. The following sections cover the different types of Power View visualizations you can create and how you can convert one visualization type to another as needed. In these examples, you'll be building the Power View reports using SharePoint 2013.

Creating Tables

You use a table visualization when you want to display your data in a simple but easy-to-read list format. By default, numeric fields are summarized at the bottom of the table. A table is the default visualization when you first begin adding data to your Power View report. Any visualization you create will first begin as a table, which can then be converted to other visualization types.

You can create a table using a number of different actions, including:

- Double-clicking a table in the Fields list, which adds the default fields to the report. Note that the default field list is a property that can be set within the data model.
- Dragging a field from the Fields list into the report area.
- Dragging a field from the Fields list into the Fields section below the list.
- Clicking a check box next to a field in the Fields list.

When you select a field to add to a report, a table is automatically created to display the data, as shown in Figure 12-8. Once you have selected a few fields to add to the report, you can modify the order of the fields in the report by simply

changing the fields' order in the Fields section. The formatting of the data and default sort order for each field are properties managed within the data model.

Country	Month Year	Total Sales	Total Cost
Australia	07-2005	\$209,652.90	\$124,323.72
Australia	08-2005	\$222,538.29	\$132,265.88
Australia	09-2005	\$173,993.51	\$105,083.23
Australia	10-2005	\$217,993.38	\$129,828.75
Australia	11-2005	\$210,683.56	\$125,241.09
Australia	12-2005	\$274,185.55	\$163,735.37
Australia	01-2006	\$223,109.11	\$132,448.01
Australia	02-2006	\$164,161.08	\$99,255.69
Australia	03-2006	\$232,584.98	\$137,757.27
Australia	04-2006	\$224,451.06	\$132,686.15
Australia	05-2006	\$205,688.58	\$121,717.42
Australia	06-2006	\$209,659.38	\$123,811.70
Australia	07-2006	\$130,989.75	\$76,804.89
Australia	08-2006	\$149,740.46	\$87,627.32
Australia	09-2006	\$116,250.79	\$70,095.26
Australia	10-2006	\$188,257.15	\$109,231.44
Australia	11-2006	\$103,439.43	\$62,692.53
Australia	12-2006	\$205,953.11	\$120,297.24

Figure 12-8: Creating a table

You can change the size of the table by simply resizing the table within the report area. You can also change the order of fields in the table by clicking a column heading.

To begin creating a new table in the report, click in the blank area of the report and begin selecting new fields from the Fields List. If you want to add an additional field to an existing table, simply select the table and begin selecting fields.

Converting Visualizations

Once you've created a table, you can convert it to many other types of visualizations. To convert your table to a chart, select the table. In the Design ribbon, expand the Visualization window and select the type of visualization you want to use, as shown in Figure 12-9.

The type of visualization you can use is restricted to the data in the table. For example, if the table does not contain a numeric-type field, you cannot convert the table to a bar or column chart. Also, if the table has more than one column, the table cannot be converted to a slicer. Based on the fields in the selected table, the Design ribbon will automatically enable and disable the visualizations that can be used.

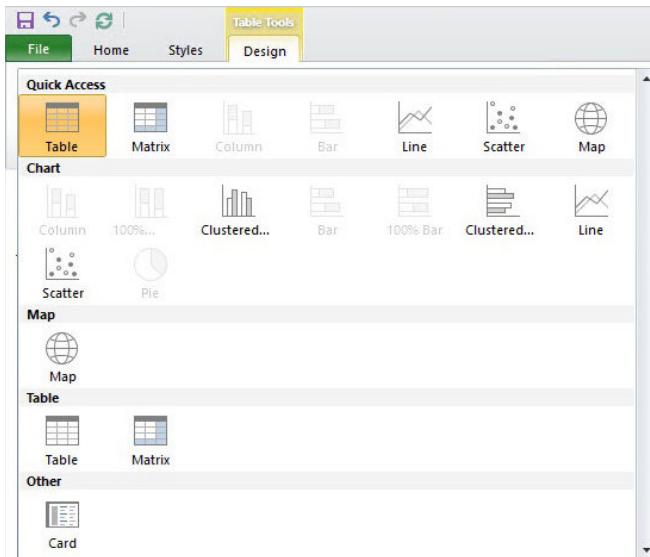


Figure 12-9: Converting visualizations

Creating Matrices

A matrix visualization is similar to a table but there are some key characteristics unique to a matrix:

- You can display fields along the top of the report as column groups.
- You can display totals and subtotals by rows and columns.

If you want to keep totals and subtotals from displaying, select the matrix visualization and use the Totals option as shown in Figure 12-10.

The screenshot shows a Microsoft Power View ribbon with the 'Matrix Tools' tab selected. Below the ribbon, a matrix visualization is displayed with a table of data. The data table has columns for Year, Australia, Canada, France, Germany, United..., and United States. The 'Totals' button in the ribbon is highlighted. A dropdown menu is open next to it, showing four options: 'None' (selected), 'Rows' (Show subtotals and totals only for rows), 'Columns' (Show subtotals and totals only for columns), and 'Both' (Show subtotals and totals for rows and columns).

Year	Australia	Canada	France	Germany	United...	United States
2005	\$1,309,047.20	\$146,829.81	\$180,571.69	\$237,784.99	\$291,590.52	\$1,100,549
2006	\$2,154,284.88	\$621,602.38	\$514,042.01	\$521,230.85	\$591,586.85	\$2,126,696
2007	\$3,039,784.21	\$535,784.46	\$1,026,324.97	\$1,058,405.73	\$1,298,248.57	\$2,838,512
2008	\$2,563,884.29	\$673,638.21	\$922,179.04	\$1,076,890.77	\$1,210,286.27	\$3,324,031
Total	\$9,061,000.58	\$1,977,844.86	\$2,644,017.73	\$2,894,312.34	\$3,391,712.21	\$9,389,789

Figure 12-10: Showing and hiding totals in a matrix visualization

You use a matrix visualization when you need to group your data in a list format or need to aggregate data along rows and columns at the same time.

Creating Charts

With Power View, you can easily create many different types of charts. Each chart type has its own unique characteristics that should be considered when designing your reports. The following sections discuss each chart type in detail.

Bar Charts

You can create a bar chart from a table as long as a numeric field is in the table. When more than one numeric field is present, a stacked bar chart, clustered bar chart, or 100% stacked bar chart can be created. The numeric values are plotted along the x-axis and the categories are plotted along the y-axis.

A bar chart is a great visualization type to use because the user can easily compare the values of one category to another. Use a bar chart when you have a limited number of categories and multiple measures you want to compare.

Selecting a bar chart visualization activates the Layout ribbon, as shown in Figure 12-11. In the Layout ribbon, you can show or hide the chart title, modify the position of the legend, add or remove data labels, or specify the data on the axis as continuous or discrete.

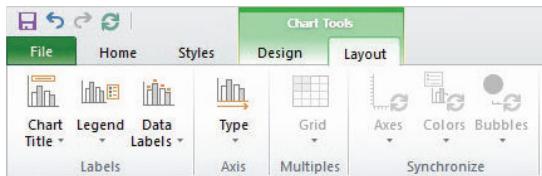


Figure 12-11: Using the Layout ribbon to customize the chart

When multiple fields are added to the axis, users will then have the ability to double-click a bar within the chart and drill down to a specified lower level. For example, Figure 12-12 shows the addition of the Country field beneath Year; when a user double-clicks the bar for the year 2007, the chart automatically displays the metrics by Country for the year 2007.

Column Charts

A column chart is very similar to a bar chart except the numeric values are plotted along the y-axis and the categories are plotted along the x-axis. A column chart can be created as a stacked column chart, clustered column chart (as shown in Figure 12-13), or 100% stacked column chart.

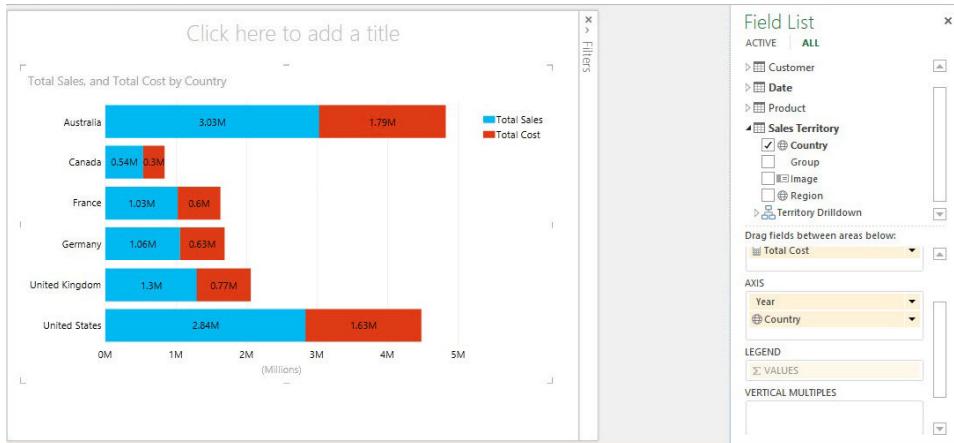


Figure 12-12: Drilling down in a bar chart

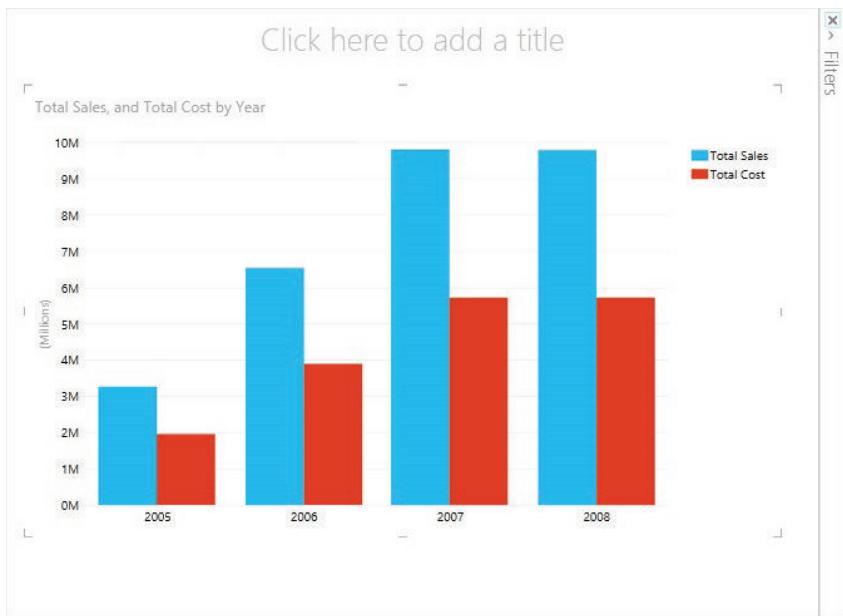


Figure 12-13: Creating a clustered column chart

You use a column chart when you want to analyze one or multiple measures for a limited number of categories. Column charts are also great for analyzing data across time because date fields can be displayed along the x-axis as in a timeline.

Line Charts

A line chart (shown in Figure 12-14) is the perfect chart for displaying numeric values across time. Although you can use any field along the x-axis of a line chart, you traditionally use line charts to display trends in your data across a range of time or date periods.



Figure 12-14: Creating a line chart

Pie Charts

You can easily create pie charts when you want to view a metric by categories as parts of a total. In Power View, however, pie charts are simple but powerful. Pie charts can be drilled into similarly to a bar chart or column chart. Also, pie charts respect the selections made in other charts as a cross-filter. For example, if you want to view the total sales by a country, you can create a pie chart. But if you select a category in a separate chart in the same report, the pie chart will display the filtered data as highlighted while the rest of the pie chart is lightened, as seen in Figure 12-15.

You use a pie chart when you need to compare metrics for a limited number of categories as part of a total. Pie charts are great when you need to easily identify which region had the most products sold, for example.

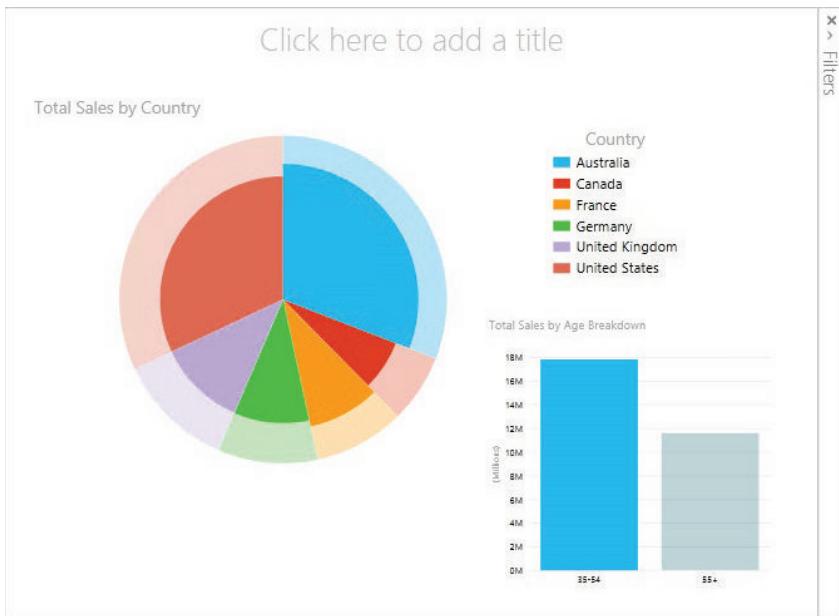


Figure 12-15: Pie charts are dynamic and interactive.

Scatter Charts

A scatter chart is an excellent visualization to use when you need to analyze the correlation between two or three metrics at once across time. It can plot numeric values along the x-axis, y-axis, and as the size of the bubble.

You can view trends and changes in the data over time by adding a time or date field to the Play axis, as shown in Figure 12-16. Now when a user clicks the play button near the bottom of the visualization, the bubbles will animate, growing in size, and moving up and down along the x- and y-axes.

A user can also focus on a particular category by clicking the desired bubble in the scatter chart. Clicking a bubble highlights the selected bubble and grays

out the other bubbles. Focusing on a bubble allows you to see the history of a category when using the play axis, as shown in Figure 12-17.

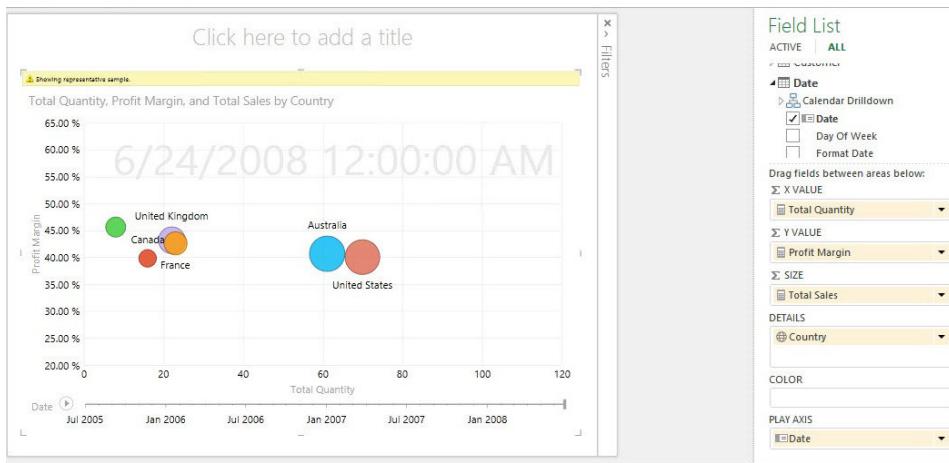


Figure 12-16: Analyzing multiple metrics across time with a scatter chart

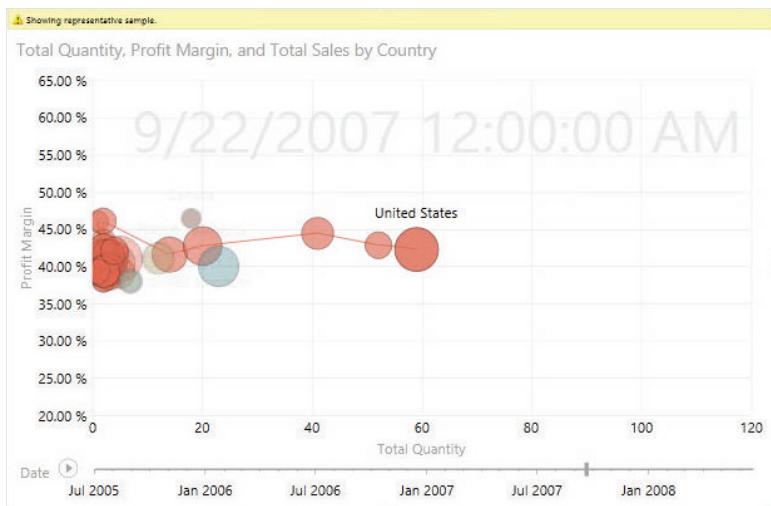


Figure 12-17: Focusing on a bubble in a scatter chart

You use a scatter chart when you need to visualize the relationship between multiple measures at the same time for a certain category in your data.

Creating Multiples

In Power View, you have the ability to create multiples of the same chart. *Multiples* are small repeating charts of the same time featuring the same x-axis and y-axis for a series of different category values.

To create multiples of a chart, first create the type of chart you want to multiply. Then add the field you want to use to multiply the chart to either the Vertical Multiples section or the Horizontal Multiples. This action creates the chart multiples, as shown in Figure 12-18.

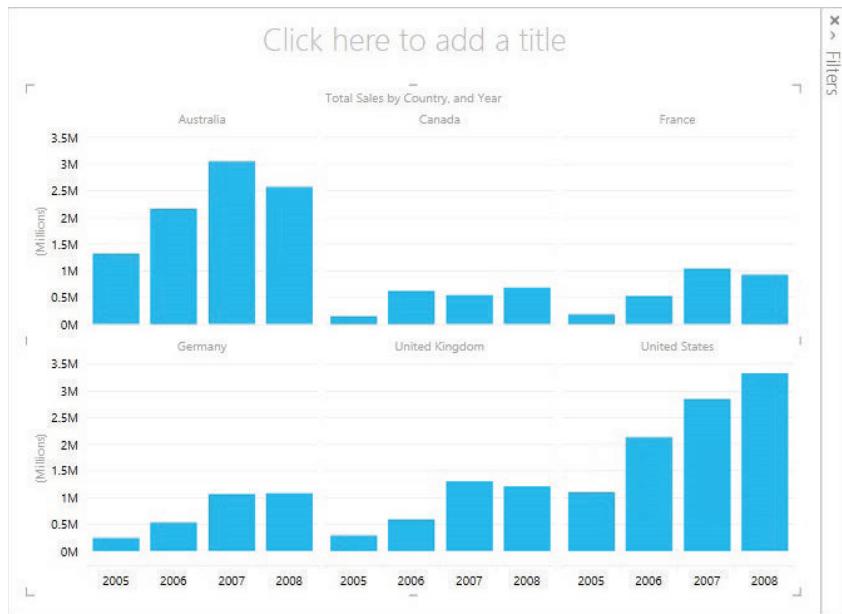


Figure 12-18: Creating multiples of a column chart

You can create multiples with column charts, bar charts, line charts, scatter charts (without the play axis), pie charts, or maps. Consider using multiples when you need to compare many different values for multiple categories at the same time across identical axes.

Creating Cards

Cards are a type of visualization unique to Power View that enables a user to quickly view data in a format not unlike that of a baseball card or index card.

A card visualization displays the default label as a header with the default image beneath. See Figure 12-19 as an example.

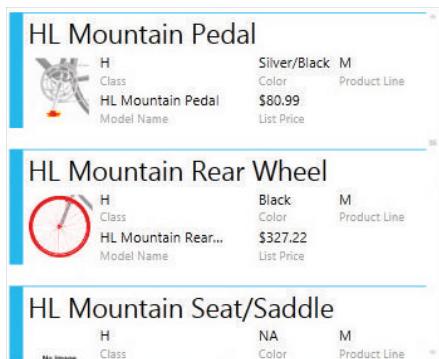


Figure 12-19: View data as a card visualization

To enable Power View to correctly display your data in a card visualization, the designer of the data model should correctly set the Default Label property for the field to be used as the card heading. Also, the Default Image property should be set on the image field in the data model.

Creating Maps

Maps provide your users with a unique way to view your organization's geographic data in a highly interactive map visualization. Power View map visualizations leverage Bing maps, giving users the ability to pan, zoom in, and zoom out just like they would if they were using Bing maps at Bing.com.

This also means that your user will need access to the Internet to use the Bing map layer. When creating a map visualization, you are prompted to enable content because Power View needs to send the data through a secured web connection to Bing for geocoding.

To create a map report, add a geographic-type field to your report, such as Postal Code, City, State, Country, or Latitude and Longitude. Then select a numeric field for the table, such as Profit or Product Cost. On the Design ribbon, click Map to convert the table to a map visualization.

The size of the dots displayed in the map represents the numeric value. To convert the dots to pie charts, as shown in Figure 12-20, add a field to the Color section of the Fields List.

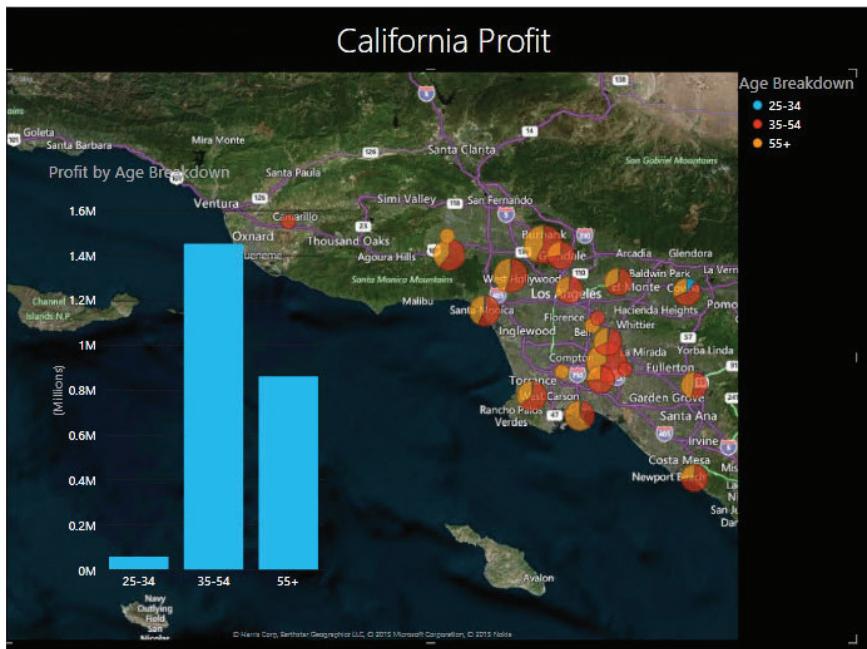


Figure 12-20: Creating a map visualization in Power View

Using Excel to Create Power View Reports

You may choose to create Power View reports in an Excel workbook rather than in SharePoint for the purposes of data exploration, quick ad hoc querying of your data, or when the same report may not necessarily be useful to other members of your team or organization. If your organization does not use SharePoint, Excel is your only option for developing Power View reports. Power View reports developed within Excel can be exported as a PDF or XPS document.

Creating Power View reports in Excel is very similar to SharePoint, except in Excel a new Power View report is created by navigating to the Insert ribbon and clicking the Power View button, as shown in Figure 12-21. A blank Power View report appears within its own sheet in the Excel workbook. Any new Power View reports created will appear on a new sheet.

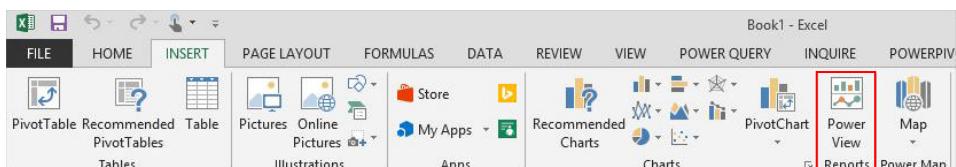


Figure 12-21: Creating Power View reports in Excel

To switch a data region between report visualization types, simply select the visualization you want to change, open the Design ribbon, and select the desired Visualization type, as seen in Figure 12-22. As in the SharePoint Power View designer, some visualization types may or may not be enabled depending on the data in the report.

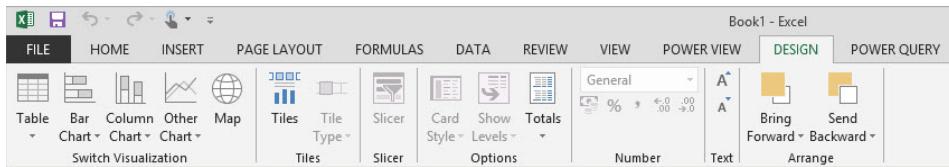


Figure 12-22: Changing visualizations in Excel

Filtering Data with Power View

In Power View, *filters* serve the purpose of displaying only the data you want to view within your visualizations. When you apply a filter to a report, Power View leverages the underlying data model to related visualizations within the same report.

Using Power View means you can apply filters to the data using a variety of features. You can apply filters to your reports using the Filter Pane, slicers, tiles, or even other visualizations in the report using cross-filters. In this section, you'll see how to use each filter type to narrow the scope of the data in your visualizations.

Adding Filters

Sometimes when developing a Power View report you may need to filter a single view, an entire report, or just an individual visualization within the report. You can accomplish each of these tasks by using the Filters Area pane. The Filter Area can be accessed by expanding the Filters Area pane at the right side of the report, as shown in Figure 12-23.

- **To show or hide the Filters Area in Excel:** Navigate to the Power View ribbon and click the Filters Area button.
- **To show or hide the Filters Area in SharePoint:** Navigate to the Home ribbon while editing the Power View report and click the Filters button.
- **To add a filter to the entire sheet or view:** Click View in the Filters Area. Then simply drag and drop fields from the Field List Area. Filters added

at the view or sheet level affect all visualizations within the view or sheet. Filters added at the view or sheet level are applied to the data at the most granular level.

- **To apply a filter to all views within a report:** Click the Pin filter icon in the Filters Area. By clicking the Pin filter icon, all views include the specified filter (see Figure 12-24).



Figure 12-23: Accessing the Filters Area

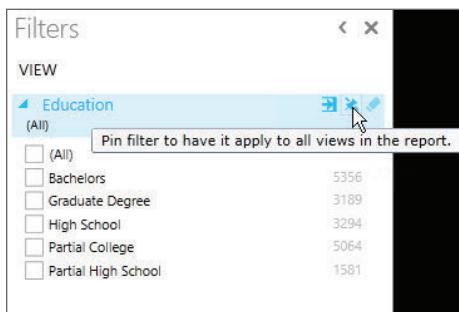


Figure 12-24: Pinning a filter to apply the filter to all views

- **To apply a filter to an individual table, matrix, card, or chart visualization:** Hover over the visualization and click the filter icon near the top right of the visualization. This opens the Filters Area for the selected visualization. Fields that do not appear in the visualization can be added

by simply dragging the field from the Fields List into the Filters Area for the visualization (see Figure 12-25).

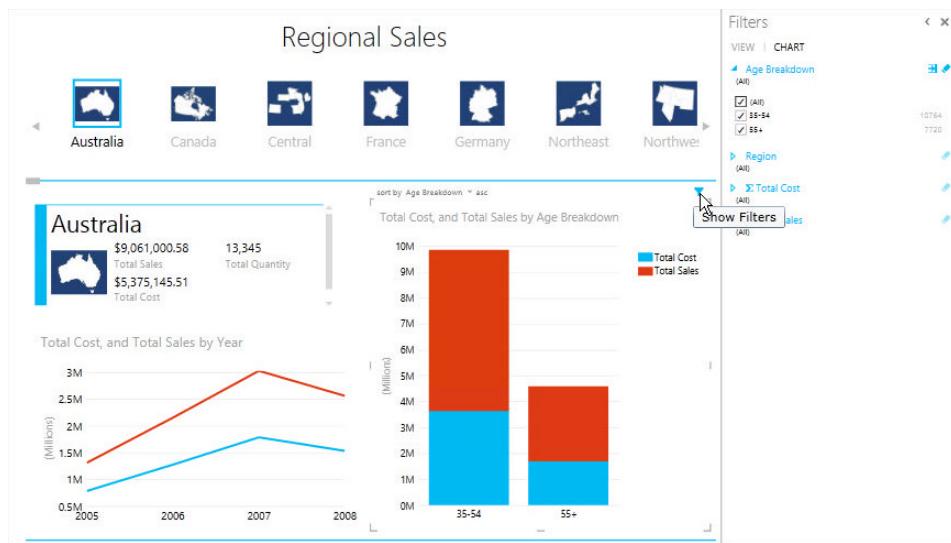


Figure 12-25: Adding filters to a visualization

- **To delete a filter:** Just click the “x” next to the field in the Filters Area. If you need to clear a filter without deleting the filter, click the Clear filter button next to the field in the Filters Area.

Any filters applied to your Power View report when the report is saved are saved as part of the report.

Using Advanced Filters

To switch between using basic and advanced filters, click the Advanced Filter Mode icon next to the field you want to use in the Filters Area (see Figure 12-26).

The Advanced Filter functionality gives you the ability to type in specific values that you want to use to filter your report. For example, you can filter numeric fields for values greater than or less than a specified value, or a text value that contains or doesn't contain a specified value. You can also use an Advanced Filter to filter for dates that occurred before or after a date you specify.

Adding Slicers

Slicers are very similar to filters because they are graphical filters that appear in the report area that affect the entire view or sheet.

To add a slicer to a report, click in the blank area of the report and select the field from the Fields List that you want to use as the slicer, which initially

creates a table visualization in the report. To convert the table to a slicer, select the table visualization, navigate to the Design ribbon, and click the Slicer button. To select multiple values in a slicer, hold Ctrl while selecting values in the slicer (see Figure 12-27).

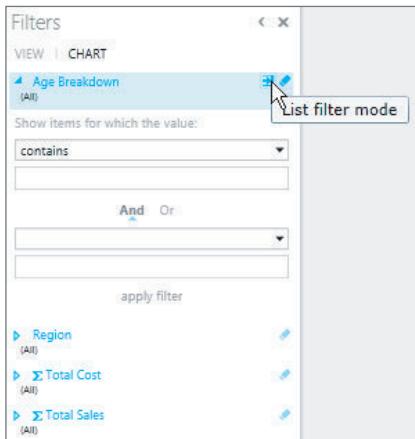


Figure 12-26: Using Advanced Filters in Power View



Figure 12-27: Using Slicers in Power View

To convert a table to a slicer, the table must have only one field. If the table has more than one field in the visualization, you will not be able to convert the table to a slicer.

NOTE Slicers are ideal for Power View reports that users will access utilizing touchscreen or mobile devices due to the ease of selecting items in a slicer by hand.

Invoking Cross-Filters

One of the great features included in Power View is the ability to cross-filter reports using visualizations within a view or sheet. A single visualization can act as a filter for the entire view.

To invoke the cross-filtering behavior in a report, simply select a category or field within a report. To select multiple categories, hold the Ctrl key while selecting. By selecting the category, you'll notice that the value is highlighted, and the other visualizations with the report are respecting the selected value as a filter. In the report shown in Figure 12-28, you can see that the column representing the United Kingdom is selected in the top visualization, which has applied a filter to the lower visualization.

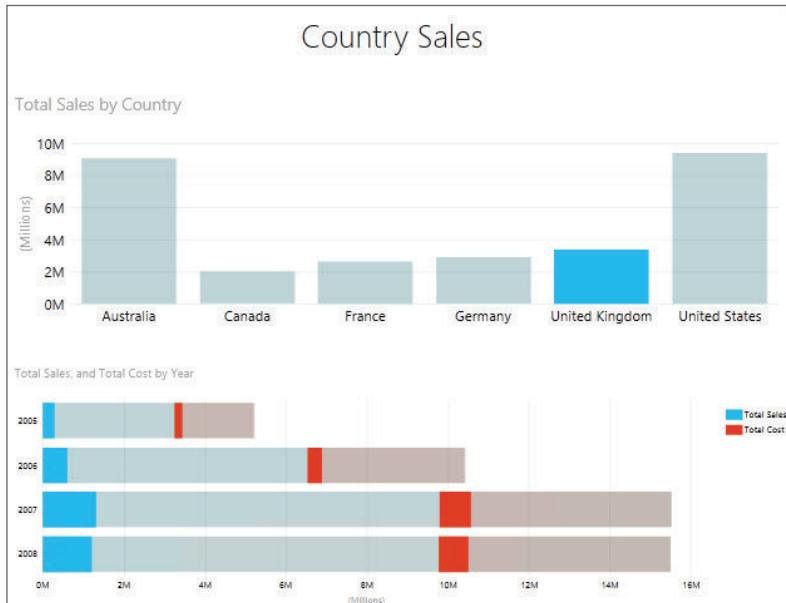


Figure 12-28: Invoking the cross-filtering behavior in Power View

Adding Tiles

In Power View, *tiles* act as interactive filters for visualizations contained within a tile visualization. A tile visualization appears as a carousel-like visualization in the report that can slide left and right, allowing the user to navigate through the unique values in the data. Images can also be used in a tile visualization.

To add tiles to a visualization, select the visualization you want to modify and add a field to the Tile By section of the Field List. This creates an interactive tile filter, as shown in Figure 12-29.

After you have created a tile visualization, other visualizations can be added to the report within the tile area. In the report shown in Figure 12-30, a line chart, column chart, and card visualization have been added to the tile area.

Now when a user selects a sales region from the tile, the visualizations change according to the selected sales region.

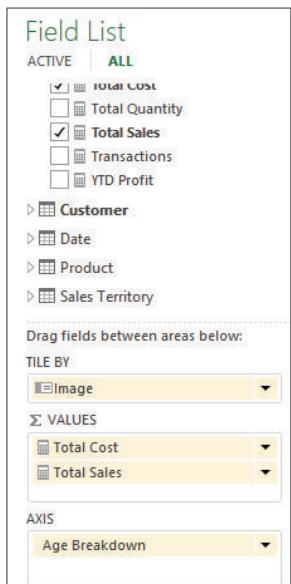


Figure 12-29: Creating a tile visualization



Figure 12-30: Using tiles to filter multiple visualizations

Consider using tiles when you need to provide your user with an intuitive and graphic way to filter the data for multiple visualizations within your Power View report.

NOTE Just like slicers, tiles are also great for Power BI solutions that may be accessed from a mobile device, because the large tiles are easy to select via touchscreen.

Adding Filters to a Report URL

You can also pass filters to a Power View report in SharePoint through the URL. To pass a filter in the URL for a Power View report, use the following syntax:

Table 12-1: URL Filters for Power View Reports

FILTER	SYNTAX
String	&rf=[table name].[field name] eq 'my value'
Numeric	&rf=[table name].[field name] eq 2015 &rf=[table name].[field name] eq 3.14 &rf=[table name].[field name] eq 'my value'
Date	&rf=[table name].[field name] eq datetime '2015-01-01' &rf=[table name].[field name] eq datetime '2015-01-01T12:00' &rf=[table name].[field name] eq datetime '2015-01-01T12:00:00.0001'
Boolean	&rf=[table name].[field name] eq true
Blank or null	&rf=[table name].[field name] eq null

For example, if you needed to pass a filter for Year and Country to your Power View report through the URL, you would use the following link:
[http://sqlbi/sites/BI%20Center%20Test/_layouts/15/ReportServer/AdHocReportDesigner.aspx?RelativeReportUrl=%2fsites%2fBI%2520Center%2520Test%2fPowerPivot%2520Gallery%2fRegional%2520Sales.rdlx&ViewMode=Presentation&ReportSection=ReportSection&rf=\[Sales Territory\].\[Country\] eq 'France'&rf=\[Date\].\[Year\] eq 2007.](http://sqlbi/sites/BI%20Center%20Test/_layouts/15/ReportServer/AdHocReportDesigner.aspx?RelativeReportUrl=%2fsites%2fBI%2520Center%2520Test%2fPowerPivot%2520Gallery%2fRegional%2520Sales.rdlx&ViewMode=Presentation&ReportSection=ReportSection&rf=[Sales Territory].[Country] eq 'France'&rf=[Date].[Year] eq 2007.)

NOTE Passing filters to a Power View report through a URL is a great way to enable a user to drill through to a Power View report from a SQL Server Analysis Services cube.

Exporting Power View Reports

Although Power View reports are intended to be highly interactive from the user's computer screen, multiple export options are available for your Power View reports.

You can export Power View reports developed in Excel as a PDF or XPS document. To export a Power View report, click File and select Export. Click Create PDF/XPS Document to export the Power View report. It's important to note that you will lose the report's interactivity by exporting the Power View report. If your report is designed to allow a user to scroll through multiples of a chart, the functionality to scroll disappears after the report is exported. Also, Excel provides the ability to print a Power View report if necessary.

Power View reports developed in SharePoint can be printed, too, but where Power View reports developed in SharePoint have an advantage over those developed in Excel is in their ability to export to PowerPoint.

To export a Power View report from SharePoint to PowerPoint, open the Power View report in your browser. Click File and select Export to PowerPoint. Power View reports exported to PowerPoint maintain their interactivity. Each view within the Power View report becomes an individual slide within the PowerPoint slide deck (see Figure 12-31).

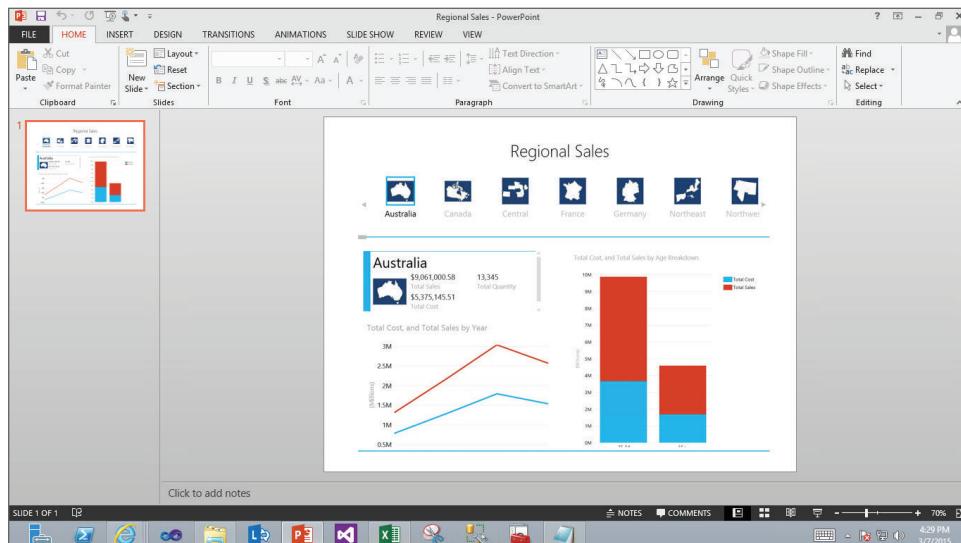


Figure 12-31: Exporting your Power View reports to PowerPoint

After exporting your Power View report to PowerPoint, simply open the PowerPoint slide deck and start the slide show. On the slide with your Power View report, click the Interact button at the bottom right of the slide while presenting to interact with the report. Now you can change filters, invoke cross-filtering, sort fields, and interact with the report any way you normally would.

It's worth noting that for the Power View report to remain interactive within the Power Point slide deck, the Power Point slide deck must have access to the Power View report in SharePoint. If the Power View report cannot access the Power View report on the SharePoint site, the Power View report will not be interactive.

Summary

Power View is a powerfully interactive reporting technology that can allow your organization's users to explore and perform ad hoc analysis on their data quickly and easily. Power View is designed to be easy and interactive with no requirement for coding or writing expressions.

This chapter discussed the potential data sources for Power View, the Power View design interfaces, the types of visualizations you can use in Power View, how to apply filters to Power View reports through a variety of methods, and how to export a Power View report.

After reading through this chapter and examining the examples presented here, you should have a deeper and more complete understanding of the capabilities of Power View, and how you and your organization can develop comprehensive and interactive reports to meet the needs of your team.

Exploring Geographic and Temporal Data with Power Map

Many organizations perform huge amount of analyses in geographical or temporal contexts. Analyzing sales is a very typical example. Business analysts frequently request sales comparisons in different regions and over time. Most of the international statistical information, like income per capita, birth and death rates, and similar categories, include statistics over countries or over time. Showing such data on map might be much more informational than showing it in a tabular format. Microsoft Power Map is an add-in for Excel that enables you to plot *geographic data* on three- or two-dimensional (3-D or 2-D) maps. In addition, you can show *temporal data* dynamically in a tour that shows a time-lapsed animation of the data.

This chapter describes Power Map and teaches you how to create dynamic *tours*. You learn how Power Map reports fit into your enterprise and personal reporting solution. Power Map is not the only tool in Microsoft BI suite that can show geographic and temporal data. A feature comparison of Power Map with Power View and SQL Server Reporting Services (SSRS) helps you decide which tool to use for different needs and scenarios. Through examples you learn in detail how to prepare the data and create Power Map reports.

How Power Map Fits into Reporting Solutions

The number of different analysis and reporting tools in Microsoft BI suite is simply overwhelming. One of the most important decisions you have to make when developing a business intelligence solution is which tool to use in which scenario. This section describes Power Map features, advantages, and requirements, and compares Power Map with other tools that can present geographic and temporal data. This knowledge should help you in choosing the right tool for your specific tasks.

If you want to create your own reports following the instructions in this chapter, you need to have Power Map installed; you will learn about Power Map requirements later in this section. In addition, Power Map reports are compared with Power View and SSRS reports. Therefore, if you want to run these reports from the companion code of the book, you also need to have SSRS and Power View available. The SSRS demo report uses SQL Server AdventureWorksDW demo database. Power View and Power Map reports use Power Pivot for Excel database. Of course, this means you also need to have Power Pivot available. The Power Pivot database is extracted from the AdventureWorksDW SQL Server database. Versions 2012 and 2014 of the AdventureWorksDW demo database are both good to create or run the examples. In addition, one table, the list of countries by population, is extracted from Wikipedia with Power Query. If you want to extract the table yourself, you also need Power Query installed. However, you can also use the Excel 2013 file, c13 - Starter.xlsx, provided with the companion content for this book. This file already includes the complete Power Pivot model, including the countries by population table.

Understanding Power Map Features and Advantages

Power Map is an add-in for Excel you can use to plot geographic data on a 3-D globe or custom map and use animations to explore temporal data visually. For example, you can use Power Map to compare sales volume by country, state, or city, and show data aggregations over time. Power Map visualizations use data from an Excel worksheet or from a Power Pivot tabular data model in an Excel workbook. With Power Map, you can:

- **Map data:** Plot more than a million rows of data visually on Bing 3-D or flattened (2-D) maps.
- **Discover insights:** Discover new patterns inside your dataset by analyzing your data in geographic space and by visually checking changes over time.
- **Create and share stories:** Capture screenshots and build video tours and share them to a broad audience.

Comparing Power Map to Other SQL Server Geospatial Reporting Tools

SSRS reports can include the Map data region. You can add a map that displays locations only; a bubble map where the bubble size or the bubble color depends on some analytical aggregated data; a marker map where you can use a marker style based on some analytical data; or a line map to display routes between points on a map. You can use a map from the following sources:

- **Map gallery:** A set of installed maps that include U.S. maps only.
- **ESRI shapefile:** ESRI shapefiles contain data that complies with the Environmental Systems Research Institute, Inc. (ESRI) shapefile spatial data format. You can upload the shapefiles to your report server.
- **SQL Server spatial data stored in a database:** You can use a query that specifies SQL Server Geometry or SQL Server Geography data types from a SQL Server relational database.
- **Bing maps:** You can add a Bing map layer to show the road, aerial, or hybrid view on your map.

You can embed spatial data in the report definition. You actually embed map elements that are used in the map layer. By embedding maps you increase the size of the report definition. However, this way you can ensure that the spatial data is always available when the report runs, even if you are not connected to the web.

Figure 13-1 shows a SSRS report that uses the Map data region. It shows the U.S. sales of the AdventureWorks company, broken down by state. States are colored based on the aggregated sales amount for each state. In addition, the report uses a bubble with a label to show the top customer in each state. The size of the bubble depends on the sales amount for the customer.

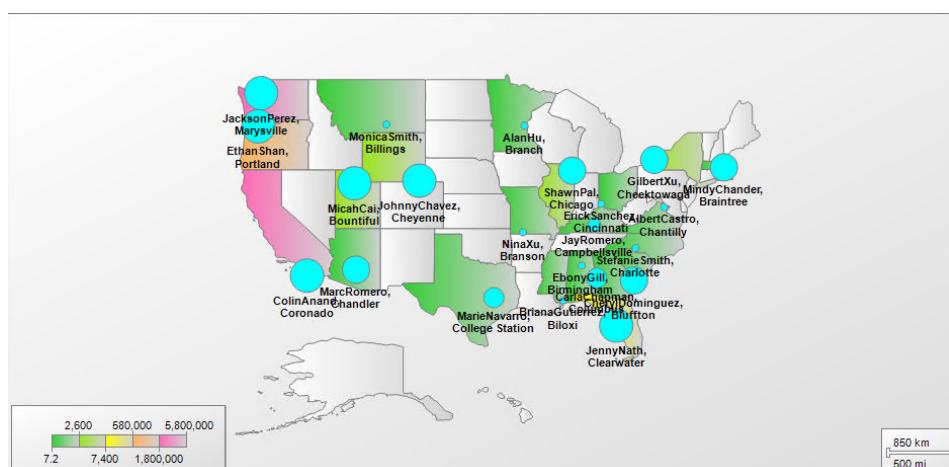


Figure 13-1: SSRS report with a map

The following query was used in the context of the AdventureWorksDW database to retrieve the data needed to show the sales by state:

```

SELECT cu1.CustomerKey
    ,cu1.FirstName + cu1.LastName AS FullName
    ,ge.StateProvinceName AS StateProvince
    ,fis.SalesAmount
FROM dbo.DimCustomer AS cu1
INNER JOIN dbo.DimGeography AS ge
    ON cu1.GeographyKey = ge.GeographyKey
INNER JOIN dbo.FactInternetSales AS fis
    ON fis.CustomerKey = cu1.CustomerKey
WHERE ge.EnglishCountryRegionName = 'United States';

```

The StateProvince column from the query gives the geographical context. The following query was used to find the top customer in each state:

```

WITH rnCTE AS
(
SELECT cu1.CustomerKey
    ,cu1.FirstName + ' ' + cu1.LastName AS FullName
    ,ad.City
    ,ad.SpatialLocation
    ,ge.StateProvinceName AS StateProvince
    ,fis.SalesAmount
    ,ROW_NUMBER() OVER (PARTITION BY ge.StateProvinceName
                           ORDER BY fis.SalesAmount DESC)
    AS rn
FROM dbo.DimCustomer AS cu1
INNER JOIN AdventureWorks2012.Sales.Customer AS cu2
    ON cu2.CustomerID = cu1.CustomerKey
INNER JOIN AdventureWorks2012.Person.BusinessEntityAddress AS bea
    ON cu2.PersonID = bea.BusinessEntityID AND
        bea.AddressTypeID = 2
INNER JOIN AdventureWorks2012.Person.Address AS ad
    ON bea.AddressID = ad.AddressID
INNER JOIN dbo.DimGeography AS ge
    ON cu1.GeographyKey = ge.GeographyKey
INNER JOIN dbo.FactInternetSales AS fis
    ON fis.CustomerKey = cu1.CustomerKey
WHERE ge.EnglishCountryRegionName = 'United States'
)
SELECT *
FROM rnCTE
WHERE rn = 1;

```

The SpatialLocation column is a SQL Server Geography data type column that gives the latitude and longitude information for the customer.

Figure 13-2 shows a Power View report with geographic data. It shows bike sales by year and by color of the bike in different countries. You can see that it uses a multi-panel of small, similar maps, one map for each year. Such a multi-panel chart is called a *trellis chart*. The report is very narrative; you can easily spot how the fashion has changed over time. For example, red bikes were popular in 2006 and 2007; however, the popularity shifted to black, silver, and yellow bikes in 2007 and 2008. With a trellis chart, you can also show the temporal context.

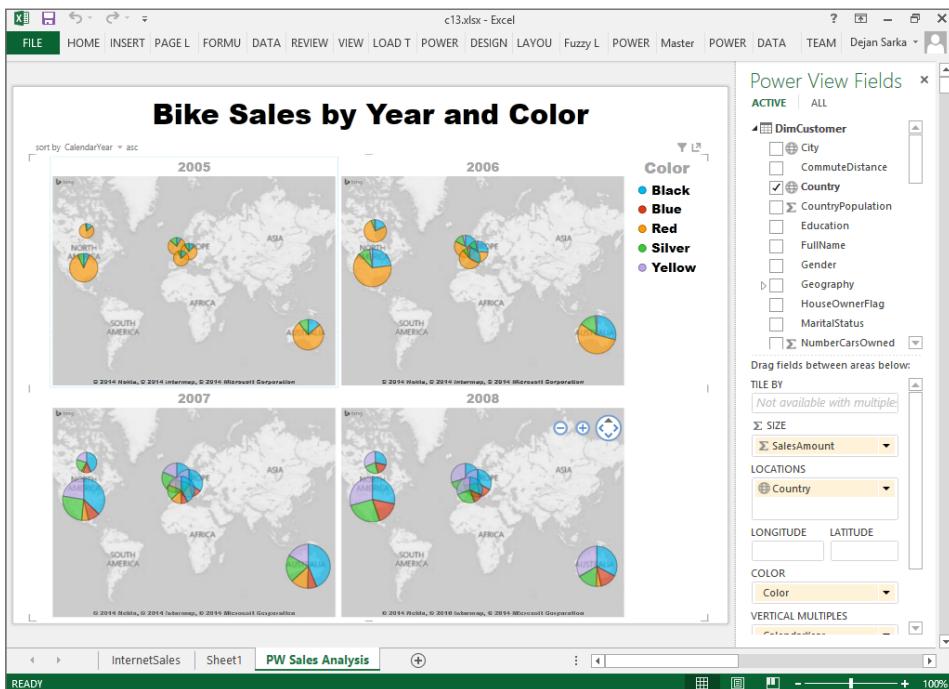


Figure 13-2: Power View trellis report

The data model used for the Power View report is the same used for a Power Map report later in this chapter. Figure 13-3 shows the diagram of this data model.

You should already be familiar with Power Pivot. For now, you will focus on the CountriesPopulation table, highlighted in the bottom right corner of Figure 13-3. This table was imported from Wikipedia with Power Query. The following code shows the Power Query Formula Language (also known as M) used to get this data:

```
let
    Source = OData.Feed("https://publicdata.clouddatahub.net/Web/Tables/
        accc2f5d95c8482dbc3792f0602dcdd8/V1/Data"),
    #"Renamed Columns" = Table.RenameColumns(Source,
        {{"Country (or dependent territory)", "Country"}}),
    ...
```

```

#"Removed Columns" = Table.RemoveColumns(#"Renamed Columns",
    {"Date", "% of world population", "Source", "Key", "Rank"}),
#"Replaced Value" = Table.ReplaceValue(#"Removed Columns","9 467,000",
    "9,467,000",Replacer.ReplaceText,{"Population"}),
#"Changed Type" = Table.TransformColumnTypes(#"Replaced Value",
    {"Population", Int64.Type})
in
#"Changed Type"

```

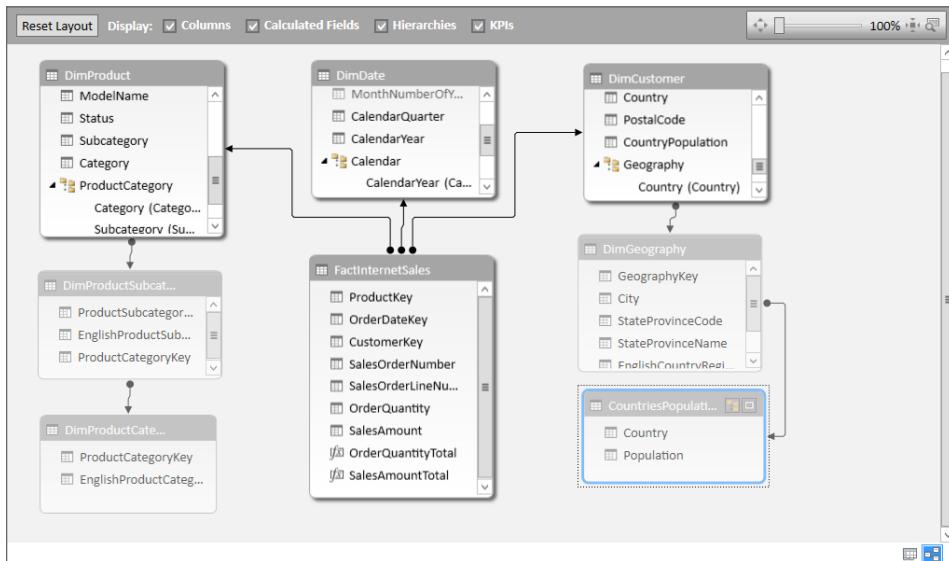


Figure 13-3: The data model for the Power View and Power Map reports

NOTE The URL in the Source = Odata.Feed("https://publicdata...") part of the code should be written in a single line. The line is broken and spaces are added here for better readability.

You have seen the SSRS and Power View reports with geographic data. The following list summarizes some of the advantages and disadvantages of SSRS, Power View, and Power Map reports.

- **SSRS:** You are limited to 2-D maps. Besides Bing maps, you can also use ESRI shapefiles. The Map gallery already includes U.S. maps. You can embed maps in your reports and make them available in offline mode as well. You can create a very complex layout and design every property of each part of the report in detail. You can even use an expression for each

property. You can export the report to various formats, including Excel, Word, PDF, XML, MHTML, CSV, and TIFF. However, developing an SSRS report might require a lot of effort and time. SSRS reports are typically authored by business intelligence developers and not by end users. If you want to create a trellis report, you have to do it manually by adding a chart inside a matrix. SSRS reports do not include time-lapsed views. You can use data from nearly any source. You can use SSRS reports in native and SharePoint integrated mode.

- **Power View:** You can use Bing maps only, meaning that you can't create or view the geographic report in offline mode. You are limited to 2-D maps. Trellis charts are supported out of the box. To view changes in data over time, you can use scatter and bubble charts, and add a time dimension to them with a play axis. You cannot add the play axis to the map report, meaning you can't show data over geography and time simultaneously. However, as you could see from the example in Figure 13-2, you can use trellis charts to add a static time dimension to maps. There are two versions of Power View: Power View in Excel and Power View in SharePoint. You can only export to PowerPoint from Power View in SharePoint; export from Power View in Excel is not available. You cannot export reports to other formats. Power View is extremely simple to use and is thus very suitable for end users. You can use data from the Power Pivot, SSAS tabular, or SSAS multidimensional data model.
- **Power Map:** 3-D maps are supported and are actually the default presentation for geographic data. You can use Bing maps only, meaning that you can't create or view the geographic report in offline mode. Time-lapsed animations are fully supported. You can export screenshots, or even create MP4 video files from your Power Map tours. You cannot export reports to other formats. Trellis charts are not supported. Power Map is also simple to use and thus suitable for end users. You can use data from a Power Pivot data model or Excel worksheets only. There is only the Excel version of Power Map.

Understanding Power Map Requirements

Before deciding to use Power Map, you should always check its requirements. The Power Map add-in is supported only in editions of Microsoft Excel that have been installed from a Microsoft Office 365 subscription. You can also download from Microsoft a free Power Map preview for Excel 2013. However, this is an unsupported add-in. It does not include all the functionality available in the Power Map add-in provided with the Office 365 version of Excel. In addition, if

you want to export the tours to MP4 videos, you need to have Microsoft Media Foundation installed. Microsoft Media Foundation enables the development of applications and components for using digital media on Windows Vista and later.

Power Map for Excel is a part of the self-service business intelligence tools in Office 365, or part of Power BI. However, Power BI tools in Office 365 do not come for free; you need to have a valid subscription to use them. Because it is not possible to expect that every reader of this book would have a Power BI subscription, all examples in this chapter use Excel 2013 with the free Power Map preview. Please note that Microsoft updates Power BI suite quite frequently; therefore, your version of Power Map could have slightly different options and user interface.

Optimizing Your Data Model for Power Map

First step in creating a Power Map visualization is data preparation. You already learned from previous chapters in this book how to create a Power Pivot data model and how to optimize it for Power View reports. There is not much to add about specific data preparation for Power Map. However, before learning about Power Map data preparation specifics, you need to understand the basic building blocks of a Power Map report.

Using Tours, Scenes, and Layers in Power Map

The basic building block of a Power Map visualization is a *tour*. You can create multiple tours in a single Excel workbook. You can have static tours, showing data aggregations for a single point in time, or dynamic tours with animations over time. You can create an MP4 video from a single tour. The only property of a tour you configure is the name of the tour.

Each tour has one or more *scenes*. For example, you can create a tour that visualizes sales data. You can use multiple scenes that play sequentially to show different aspects of this data—for example, sales amount, market coverage, number of distinct customers in different regions, profitability, and more. For each scene, you can configure the following properties:

- **Duration:** The number of seconds for the scene animation.
- **Name:** The name of the scene.
- **Transition duration:** The amount of time for transition from one scene to another.

- **Effect:** The animated movement pattern over the map when viewing the scene.
- **Effect distance:** The magnitude or speed of the scene effect.
- **Start date:** The start time for data animation in this scene. Default is the earliest time point in your data.
- **End date:** The end time for data animation in this scene. Default is the latest time point in your data.
- **Speed:** The speed at which the animation moves.

In addition, in each scene you can customize how the map appears through the following options:

- **Themes:** Define the colors of the map. Can also have satellite images for the globe.
- **Map labels:** Show or hide the labels on the map.
- **Flat map:** Switch between globe (3-D) and flat map (2-D) format.
- **Find location:** Go to address, city, country, or other geographical location on the map.

Each scene can have one or more *layers*. Besides sales data, you might want to show other data related to the geographic locations you are showing. For example, you can show the sales amount and population density for each region on separate layers. For each layer, you can configure the following options:

- **Formatting:** Depending on the type of the visualization, you can change different formatting options. For example, for clustered columns, you can define opacity, height, and thickness. For region coloring, you can change the opacity and color scale. You can always define colors for different categories of data.
- **Data:** You have options to show or hide zeroes, negatives, or nulls.

When you start Power Map from an Excel workbook that does not contain a tour yet, a new tour with a single scene with a single layer is automatically created. If a workbook already contains one or more Power Map tours, you can select an existing tour or create a new one when you launch Power Map. You launch Power Map from the Insert menu. Figure 13-4 shows the Launch Power Map pop-up window for a worksheet that already contains a tour.

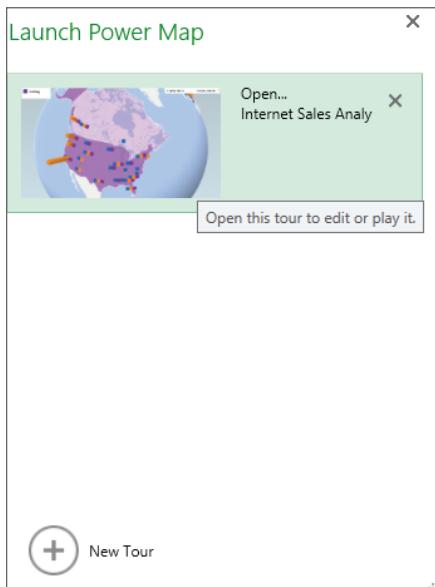


Figure 13-4: The Launch Power Map window with an existing tour

Defining Geography Fields in Your Data Model

Each layer in a scene must have at least one column with a geographic data value that plots the data on the map. Power Map supports several geographic formats and levels, including:

- Latitude/longitude data formatted as two decimal columns in a table
- Street
- Address
- City
- County
- State/Province
- Zip Code/Postal Code
- Country/Region

Depending on the type of geographic data, you might have problems with ambiguity when matching this data to geographic locations. Longitude and latitude are, of course, always resolved to a geographic location without ambiguity. If you use only cities to define geographic locations, you can have a lot of ambiguity, and Power Map cannot resolve all the cities properly. For example, besides Paris in France, there are over twenty cities named Paris in the U.S., and there are also cities with this name in Canada, Denmark, Kiribati, and even in Russia. However, if you also specify country, state, and zip columns, then Power Map can use this information to resolve the city Paris from a specific row in a table to a correct, unambiguous geographic location. Figure 13-5 shows customer data with city, state, country, and postal code geographic columns for the DimCustomers table in the Power Pivot data model. The data in the table comes from the dbo.DimCustomers table from the AdventureWorksDW demo database.

[FullName]		=DimCustomer[FirstName] & " " & DimCustomer[LastName]	City	StateProvince	Country	PostalCode	CountryPopulation
Wyatt Hill		Imperial Beach	California	United States	91932	318081000	
Ethan Zhang		Bellingham	Washington	United States	98225	318081000	
Chase Cox		La Jolla	California	United States	92806	318081000	
Gabriel Wang		Spring Valley	California	United States	91977	318081000	
Hunter Clark		Redmond	Washington	United States	98052	318081000	
Richard Torres		San Diego	California	United States	92102	318081000	
Carson Patterson		Fremont	California	United States	94536	318081000	
Xavier Alexander		Concord	California	United States	94519	318081000	
Dalton Thompson		Lynnwood	Washington	United States	98036	318081000	
Noah Lopez		Daly City	California	United States	94015	318081000	
Jack Hayes		Sedro Woolley	Washington	United States	98284	318081000	
Ian King		Lebanon	Oregon	United States	97355	318081000	
Brandon Alexander		Lincoln Acres	California	United States	91950	318081000	
Alexander Johnson		Long Beach	California	United States	90802	318081000	
Eric Collins		Santa Monica	California	United States	90401	318081000	
Brandon Russell		San Carlos	California	United States	94070	318081000	
Alfredo Jiménez		Burien	Washington	United States	98168	318081000	

Figure 13-5: Geographic columns in the DimCustomers table

Defining Date and Time Fields in Your Data Model

For time-lapsed animations, you need to have a column of the date data type column in a Power Pivot data model table. Figure 13-6 shows such a column from the DimDate table in the Power Pivot data model. The data in the table comes from the dbo.DimDate table from the AdventureWorksDW demo database.

Date...	Month	Add Column				
Day	MonthNumberOfYear	CalendarQuarter	CalendarYear	Date	Month	Add Column
20050701	7	3	2005	7/1/2005 12:00:00 AM	July	
20050702	7	3	2005	7/2/2005 12:00:00 AM	July	
20050703	7	3	2005	7/3/2005 12:00:00 AM	July	
20050704	7	3	2005	7/4/2005 12:00:00 AM	July	
20050705	7	3	2005	7/5/2005 12:00:00 AM	July	
20050706	7	3	2005	7/6/2005 12:00:00 AM	July	
20050707	7	3	2005	7/7/2005 12:00:00 AM	July	
20050708	7	3	2005	7/8/2005 12:00:00 AM	July	
20050709	7	3	2005	7/9/2005 12:00:00 AM	July	
20050710	7	3	2005	7/10/2005 12:00:00 AM	July	
20050711	7	3	2005	7/11/2005 12:00:00 AM	July	
20050712	7	3	2005	7/12/2005 12:00:00 AM	July	
20050713	7	3	2005	7/13/2005 12:00:00 AM	July	
20050714	7	3	2005	7/14/2005 12:00:00 AM	July	
20050715	7	3	2005	7/15/2005 12:00:00 AM	July	
20050716	7	3	2005	7/16/2005 12:00:00 AM	July	
20050717	7	3	2005	7/17/2005 12:00:00 AM	July	
20050718	7	3	2005	7/18/2005 12:00:00 AM	July	
20050719	7	3	2005	7/19/2005 12:00:00 AM	July	

Figure 13-6: Date column in the DimDate table

Working with Geospatial and Temporal Data

After the data is prepared, it is time to create a Power Map tour. In this section, you learn about data aggregation visualizations and how to create tours with scenes and layers and with time-lapsed animation.

Visualizing Data Aggregation

Analytical data is typically aggregated over geographic and temporal data. You need to specify the columns with the analytical data, and which aggregation function and visualization to use. Power Map supports the following aggregate functions:

- Sum
- Average
- Count (Not Blank)
- Count (Distinct)
- Maximum
- Minimum
- No Aggregation

For numeric data, Sum is the default aggregate function. For non-numeric data, Count (Not Blank) is the default aggregate function.

Besides geographic and temporal aggregations, you can also categorize your analytical data. You can specify a single column for the data categorization. For example, you can explore sales over geography, time, and product categories. You can use the following visualizations for your analytical data:

- **Stacked Column:** Shows aggregated values as columns on the surface of the map. If you use the category field, then the columns are stacked.
- **Clustered Column:** Shows aggregated values as columns on the surface of the map. Zero values or nulls are shown as a flat square, and negative values are shown as columns that appear to be under the surface of the map. Without a category field you get a single column; with a category field you get clusters of columns with each category visualized with a different color.
- **Bubble:** Shows aggregated values as colored circles on the map. A bigger circle means a higher value than denoted with a smaller circle. If you categorize your analytical aggregations, the circles are multicolored.
- **Heat Map:** Shows aggregated values as color-coded circles on the map. More intense color of a circle means a higher value.
- **Region:** Shows aggregated values as colored geographic areas like countries or states. A more intense color means a higher value. If you categorize your analytical aggregations, then colors indicate specific categories of the values.

Creating a Power Map Tour

You launch Power Map in an Excel workbook by clicking the Map icon on the Insert tab on the ribbon. You edit Power Map tours in a separate Power Map window. This window has by default three panes:

- The globe in the middle pane.
- Tour Editor pane in the left side, where you specify a name for the tour and view a list of the scenes in the tour. By default, a tour includes one scene. You can add more scenes later.
- Layer pane in the right side, where you define geographic, analytical, categorical, and temporal data.

The next example shows the steps to create a new tour. You can test these examples using your own worksheet, or the c13 - Starter.xlsx workbook provided with the companion content for the book. The examples assume that you are using the workbook mentioned.

1. In an Excel workbook with Power Pivot data model, click Insert \Rightarrow Map \Rightarrow Launch Power Map. If there is no Power Map tour in the workbook yet, a new tour is created. Otherwise, click New Tour in the pop-up window.
2. In the Tour Editor, rename the default tour name, Tour 1, to Internet Sales Analysis.
3. Rename the layer from the default name to some more meaningful name. Use the stencil icon to the right of the layer name to rename the layer.
4. Select geographic fields. In this example, Country, StateProvince, City, and PostalCode fields are used. Match your fields to appropriate geographic locations: Country/Region, State/Province, City, and Zip. Select the State/Province map level. Figure 13-7 shows the first layer renamed to Sales by Category with fields mapped to geographic locations and with State/Province map level selected. In addition, the Rename this layer stencil is selected, to help you in locating it.

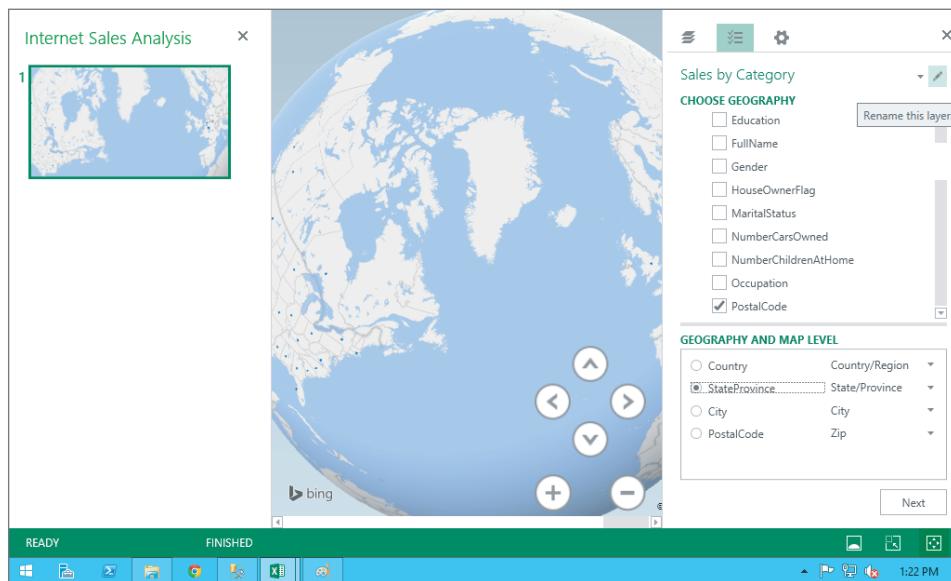


Figure 13-7: Sales by Category layer geographic data

5. In the Layer pane, click Next. You see the next set of settings for the layer.
6. Select the Clustered Column visualization. Select the SalesAmount column from the FactInternetSales table for the Height.

7. **Select the Category column from the DimProduct table for Category.**
In the Globe pane, a legend with category labels immediately appears. Resize it appropriately and move it to the top left corner.
8. **Click the Settings button (the third icon above the layer name).** Change the formatting options. Raise the Height to 200 percent and Thickness to 400 percent. Figure 13-8 shows the Sales by Category layer after the selection of the fields and after changes in the formatting of the layer. In addition, the Tour Editor pane is hidden in order to have more space for the Globe panel. Finally, the Layer Manager button is highlighted. You will need it to add another layer.



Figure 13-8: Sales by Category layer finished

In the second layer, you color countries based on the country population. Use the following steps as an orientation on how to add the second layer.

1. **Click the Layer Manager button.** Click the Add another data layer to the selected scene button.
2. **Rename the layer to Country Population.** Select the Country field for geographic data and Country/Region map level.

3. **Select the Region data visualization.** Select the CountryPopulation column for the Value field. Use the Average aggregation function. Resize the Country Population legend and move it to the top right corner.
4. **Format the layer by changing Opacity to 20 percent, and changing the color for the CountryPopulation (Average) aggregate to Purple, Lighter 40 percent.** Select the State/Province map level. Figure 13-9 shows the Country Population layer defined and formatted.

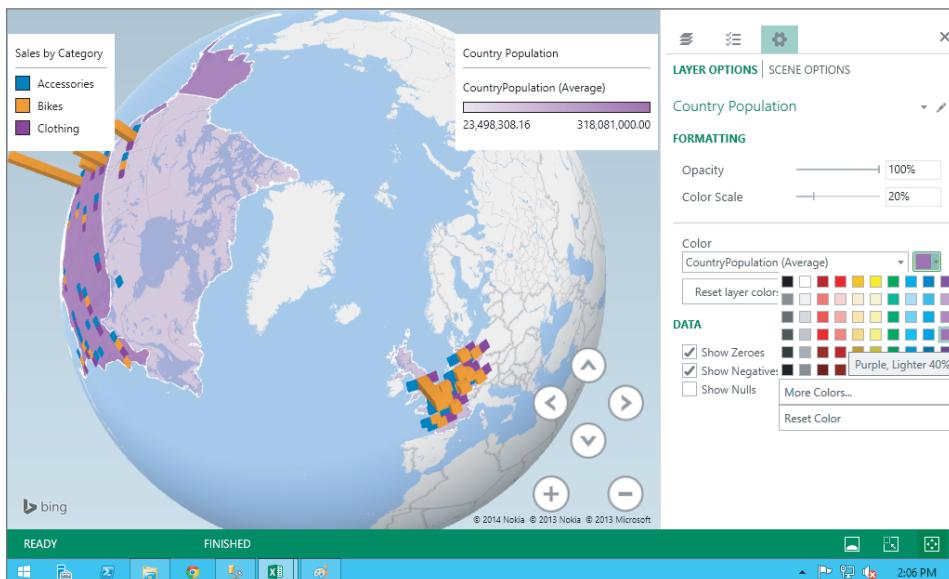


Figure 13-9: Country Population layer

Note that Power Map has no Save button. All tours and scenes are preserved automatically in the state in which you close the window. When you save your workbook, Power Map tours and scenes are saved with it. Changes that you make to a scene when you play the tour—for example, resizing a legend for better visibility of some locations in the globe—are not saved.

Visualizing Data Over Time with Rich Animations

Power Map scenes created so far show static data in geographic locations. This might already be a useful view. However, you can make the scenes dynamic and add time-lapsed animations by adding the temporal data. You can use any date column from your Power Pivot model to add the animations. Power Map

animates values in chronological order. You can specify a time period for an animation. This way, the data visualizations change on a regular interval. You can select the following intervals:

- None
- Day
- Month
- Quarter
- Year

You also need to specify how data values change when the time period is incremented. You can select one of the following options:

- **Data shows for an instant.** Each data point appears only for the duration of the time increment and is redrawn for each time interval.
- **Data accumulates over time.** The values for each data point are accumulated to show how the value grows over time.
- **Data stays until replaced.** Data visualizations remain on the map until the data value for that visualization changes.

When you add a date column to the Power Map layer Time field, a floating window, called the *time decorator*, appears. It shows the time interval values during the animation. You can format the time interval shown, or you can remove the window from the scene. In addition, the Time Line control appears at the bottom of the scene.

Follow these steps as an orientation on how to add the time animation:

1. **Edit the Sales by Category layer.** Add the Date column from the DimDate table to the Time field. Select the Month interval in the drop-down list in this field.
2. **Move the time decorator window to the bottom left corner of the scene.** Right-click the time decorator and select the Edit option to change the time format and font. Change the time format to *MonthName YYYY*, like August 2008. Change the default font to a smaller one. Resize the window appropriately.
3. **Change how the time animation is shown for this layer.** Use the clock icon at the top right side of the Time field to open the drop-down list, and select the Data accumulates over time option.
4. **Define the scene options.** You can click the Settings button for the layer and then select the Scene Options tab, or you can click the Set time options for the current scheme button on the Time Line control. You can also use

the Change scene options button in the scene icon in the Tour Editor pane for this task.

5. **Change the scene name to Sales over Months.** Change the scene duration to 30 seconds and transition duration to 2 seconds. Change the effect to Dolly, and raise the effect distance to the maximal possible value. Figure 13-10 shows the Sales by Category layer animation defined and formatted, with the Change how the time animation is shown for this layer drop-down list opened.

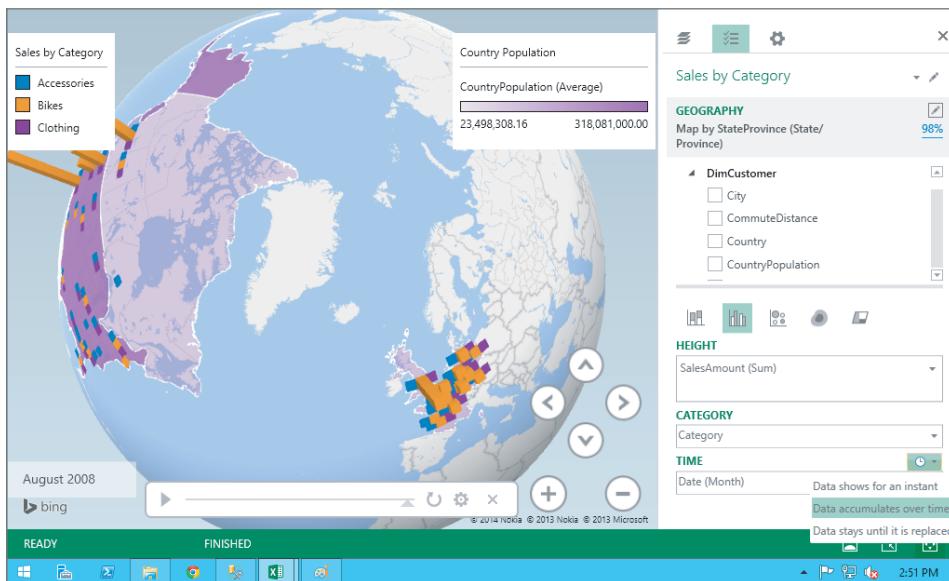


Figure 13-10: Sales by Category layer time animation defined

The Power Map tour is finished. You can play the tour to see how the animation works. You can also pause the animation and then double-click somewhere in the map to zoom in and explore the data.

Deploying and Sharing Power Map Visualizations

Before sharing your Power Map visualizations, you can add a couple of embellishments:

- **Legends:** You can add a legend for each layer to indicate the colors and scales that are used to visualize the data.

- **Text Boxes:** You can provide additional information about the scene in one or more text boxes. A text box has a heading and content.
- **Annotations:** You can add labels to specific data points on the map. These labels are called *annotations*. You can use annotations to add additional explanation for the value in a data field.
- **Charts:** You can add a two-dimensional chart that shows the top or bottom 100 data values. This can be useful to focus on the most important values when a map contains many data values.

Sharing Power Map Tours

You can share your Power Map tours in many different ways. The simplest way to share them is to share the Excel workbook that contains the Power Map tours.

You can capture one or more screens when you edit a Power Map visualization. Simply use the Capture Screen button in the Power Map editor ribbon. The screen is captured to the clipboard. You can then paste it to any application—for example, to Microsoft Paint for further editing, or to PowerPoint to create rich presentations.

You can also export a Power Map tour as a video. You can add a soundtrack from an audio file to your video or add narration or music to accompany the tour. Note that you have to create the audio file separately; you can't create audio files from Power Map. Please refer for details to the official Office support site at <https://support.office.com/en-sg/article/Export-a-Power-Map-tour-as-a-video-0082dc6b-3ed6-4690-96f7-e0579d1c1eec>.

To create a video, use the Create Video button in the Power Map editor ribbon. Pick a video quality setting based on the device you expect your audience to use. You can create a separate video for each available quality:

- 1080p for a high-definition video display
- 720p for a desktop, notebook, or tablet
- 360p for a mobile device or a small tablet

To attach a soundtrack, click the Soundtrack Options button in the Create Video window.

Enhancing Power Map Deployment and Configurations in Office 365

Power Map in Office 365 offers a couple of additional options you can use before deploying your tours. You can use filters to see individual portions of your dataset and compare how different factors can alter your data. You have three different filtering options:

- **List filter:** This allows you to select or exclude individual categories and is especially useful for filtering based on a column with discrete values.
- **Range filter:** With this, you can filter a numerical field between the minimum and maximum values, including the minimal and maximal value. You can filter on aggregated values and even change the aggregation to choose whether to filter based on the sum, average, count, minimum, or maximum for each point on the map.
- **Advanced filter:** For this, you can use predicate statements to filter your data. The predicate statements can include multiple conditions, connected with logical operators. For example, you could filter data to include only sales with a profit margin lower than 5 percent and a sales date after 8/01/2014.

Filters operate on the layer level. That means the filters in one scene are independent from the filters in another scene, and filters in one layer of a scene are independent from the filters in other layers of the same scene.

In Power Map for Office 365, you can also use custom maps. Custom Maps use x and y coordinates in your data to plot the data across any image file. This means you can adjust the data to fit your image file as needed. You can expand and collapse the data across the image with the Scale option. You can shift the data in any direction with the Offset option. When you finish editing a custom map, you can use it in multiple scenes and in multiple tours. You can use custom maps for any kind of map, not just for the globe. For example, you can use a custom map to visualize analytical data on a building floorplan.

After you enhance your Power Map visualization with Office 365 options, you can deploy and share it just like you do from Excel 2013. You can share a workbook, capture screens, and create videos.

Summary

Power Map is an important part of the Power BI suite. You can create rich geographic visualizations and time-lapsed animations with minimal effort. Of course, you need to understand how Power Map visualizes the geographic, temporal, and analytic data to prepare the datasets appropriately. Nevertheless, preparing the data for Power Map is not a complex task. Typically, when your Power Pivot data model is optimized for Power View reports, you can use it for Power Map tours immediately.

Monitoring Your Business with PerformancePoint Services

If you're reading this book, then it is a safe bet that you care about and are interested in business intelligence. You care about designing, creating, and providing actionable intelligence and information to managers, executives, or other users that will allow them to gain valuable insights into their business as easily as possible. Providing this kind of reporting is not always easy, but it is a task that PerformancePoint Services (PPS) was designed to accomplish.

This chapter takes an in-depth look at some of the powerful and interactive features of PerformancePoint Services; where PerformancePoint Services fits within your organization's reporting solution; how you can create dynamic and useful reports; and finally how you can deploy and secure your PerformancePoint Services solution. This chapter is about more than just combining a bunch of KPIs, charts, and grids on a dashboard. The concepts and features discussed here give you a more complete understanding of how to build upon your basic knowledge of PerformancePoint Services, so you and your team can better leverage it to give your organization unique insights into the business.

Where Does PerformancePoint Services Fit with Your Reporting Solution?

To get the best results and value from your PerformancePoint Services reports, it's important to understand the purpose of the technology. Using PPS, you can develop many different report types that are discussed in this section. It's important to remember, however, that the goal of PerformancePoint Services is to develop individual reports and assemble those reports together into a single dashboard report. For example, you may develop an Analytic Chart, Analytic Grid, and a KPI Scorecard and display those reports within a dashboard report, thus giving users the complete overview of the organization's top metrics in a single report, as you can see in Figure 14-1.

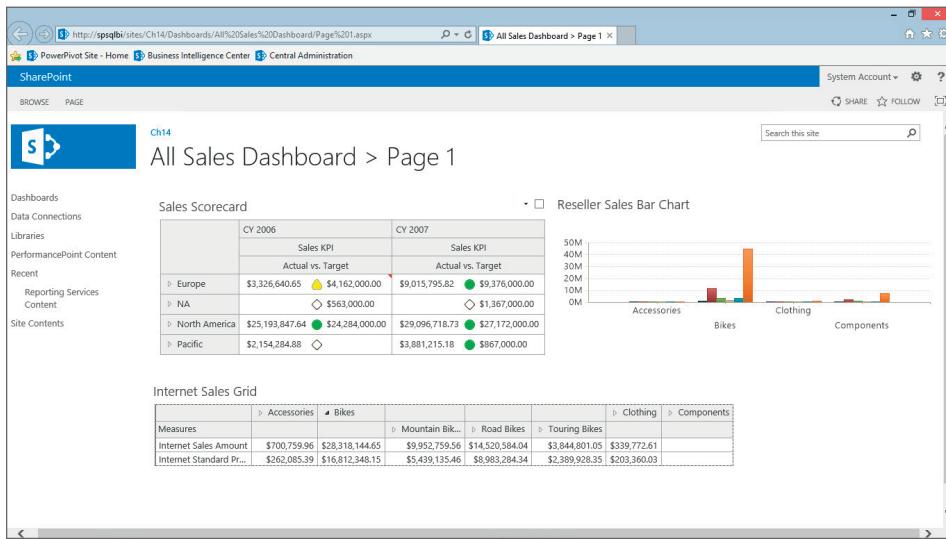


Figure 14-1: A Scorecard, Analytic Chart, and Analytic Grid in a dashboard

That's where PerformancePoint Services shines: creating flexible yet powerful dashboard-style reports. The dashboards are very interactive and allow the user to modify and explore the data unique to PerformancePoint Services. As you read through the following sections, think about how you could use the report types, features, and concepts discussed together as individual pieces of a larger dashboard-style report for your organization. By keeping this in mind, your organization can realize great value from the dashboard reports you'll develop with PerformancePoint Services.

Understanding PPS Features

PerformancePoint Services is stacked full of features and capabilities that really give developers an impressive toolbox—enabling them, courtesy of the Dashboard Designer, to create powerful, interactive, and multifunctional reports, specifically dashboards.

Analytic Charts

Analytic Charts are probably the most commonly used report type you will find in PerformancePoint Services, and for a good reason. The Analytic Chart is one of the most dynamic, interactive, and visually powerful report types for the web. Notice the example in Figure 14-2.



Figure 14-2: PPS Analytic Chart

With the Analytic Chart, users of your dashboard have the ability to interact with the report by changing the metrics being analyzed, altering the chart type, changing the layout, and drilling into and through the data, thus exploring the data in a way that was never before possible.

Analytic Chart reports are developed using the Dashboard Designer. The chart types available include a bar chart, stacked bar chart, line chart, and pie chart.

Analytic Grids

The *Analytic Grid* report type displays the data in a grid format rather than on a graphical chart. This type of report is also designed using the Dashboard Designer. Analytic Grid reports are very useful for displaying Key Performance Indicators or other numeric data alongside an Analytic Chart. Notice the example in Figure 14-3.

	Accessories	Bikes	Mountain Bikes	Road Bikes	Touring Bikes	Clothing	Components
Measures	\$571,297.93	\$66,302,381.56	\$26,492,684.38	\$29,358,206.96	\$10,451,490.22	\$1,777,840.84	\$11,799,07
Reseller Sales Amount	\$571,297.93	\$66,302,381.56	\$26,492,684.38	\$29,358,206.96	\$10,451,490.22	\$1,777,840.84	\$11,799,07
Reseller Standard Pr...	\$71,122.89	\$22,039,062.39	\$7,643,163.79	\$10,330,875.35	\$4,065,023.25	\$294,511.26	\$4,289,13

Figure 14-3: PPS Analytic Grid

Analytic Grids are also very flexible. A user has the ability to pivot the grid, filter out empty items, drill down through the data, or sort the data in different ways.

Excel Services Workbooks

Excel Services reports that have been deployed to a trusted SharePoint document library can be reused in part or as a whole. Then the Excel Services report can be published as a PerformancePoint Web Part.

Using an existing Excel Services Workbook in your PerformancePoint Services reports has several benefits. The Excel workbooks are interactive, so your users can sort and filter the data in the Excel workbook within their web browser or by opening the Excel workbook in Excel. Also, using an existing Excel workbook in your PerformancePoint Services report saves you development time by not having to develop a new report to display the data. Finally, in Excel you

have the ability to create reports leveraging data sources not available to the Dashboard Designer.

KPI Details

KPI Details reports are a useful report type to include in your dashboard because it allows you to display additional information regarding your scorecard metrics without taking up additional space within the report. To access the KPI details from within your dashboard, simply click a KPI on your scorecard. This action displays the KPI Details report, which shows how performance is calculated, the type of indicator used, and banding settings.

Reporting Services

SQL Server *Reporting Services* reports are highly flexible and offer many different possibilities regarding the report layout and the types of reports you can use. You can deploy Reporting Services reports directly to SharePoint and display them in your PerformancePoint dashboard solution as a Web Part.

Reporting Services reports can be very interactive, allowing users to sort and filter the data as well as exporting the report in many different formats, such as image, Adobe PDF, web, or Excel files.

Strategy Map

A *Strategy Map* report is a type of report created using a Scorecard as a data source and a Visio diagram as the report. The Scorecard metrics are overlaid on the Visio diagram, creating a Strategy Map that not only displays relationships between concepts, objectives, or other organized items, but also displays the color of the key performance indicator on the Strategy Map.

Web Page

You can display a *Web Page* report within a Web Part, which can be displayed within your PerformancePoint solution. This is especially useful if your organization has valuable business information displayed on various web pages through your organization. For example, you may use a Web Page report to display certain textual details about your PerformancePoint dashboards within the dashboard. You could also use a Web Page report to integrate report types with your PerformancePoint dashboards that cannot be created using Dashboard Designer.

Decomposition Tree

A *Decomposition Tree* report is a very powerful yet flexible report that allows a user to drill through and decompose an individual member into the lower level of the specified member. Figure 14-4 shows an example of a Decomposition Tree report.

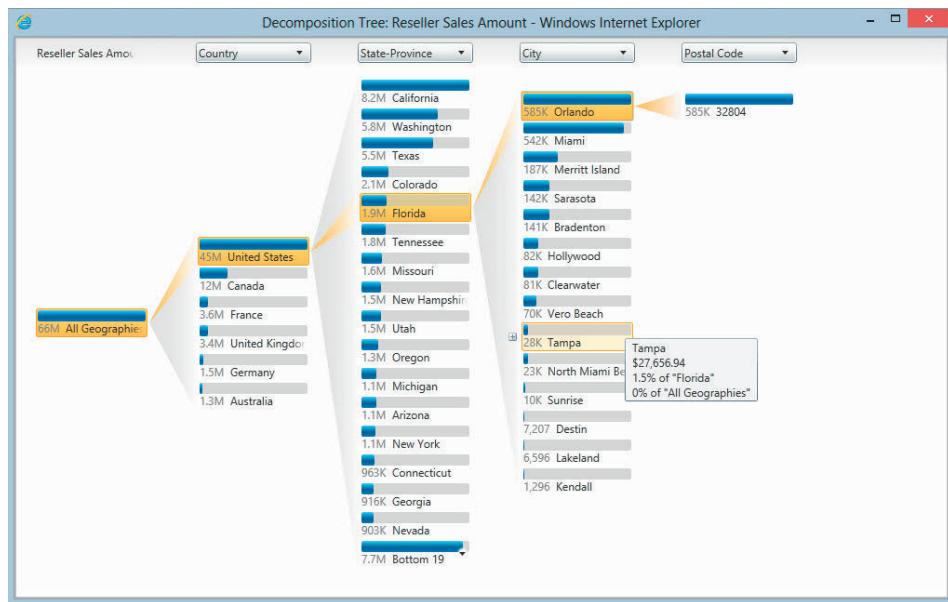


Figure 14-4: Decomposition Tree

The Decomposition Tree report type is designed to give the user the ability to conduct an ad-hoc analysis of individual items in a report in an attempt to discover where a value originates. With a Decomposition Tree report, a user can sort the data or create an ad-hoc Analytic Chart or Analytic Grid providing further insights into the data.

Scorecards

A *Scorecard* report is a special type of report that can display a collection of KPI's and is designed to provide insights into how your organization is achieving its goals. Scorecard reports can display Key Performance Indicators, which indicate progress to reaching a specific business objective or target. These reports are designed to provide business intelligence at a quick glance and are typically easy and quick to consume by a user.

When Is PPS the Right Choice?

The optimal reporting solution for your organization will most likely include the use of multiple types of reporting technologies. To see where PerformancePoint

Services fits within your organization's reporting solution, you need to understand where PerformancePoint Services excels and what it was intended for.

After working with PerformancePoint Services for only a short time, you begin to realize that the main goal of the tool is to create dynamic and interactive dashboards. PPS gives the developer the ability to design Analytic Grids, Analytic Charts, KPIs, Strategy Maps, and other report components, but the purpose of creating those components in the first place is to bring them together into a larger and more comprehensive dashboard. Each individual component is developed first and then brought together with other components into a single dashboard. In Figure 14-5 you can see an example of a dashboard report that combines a KPI Scorecard, an Analytic Chart (bar chart), and an Analytic Grid. If you are looking to develop comprehensive and multifunctional dashboards, PPS is the right tool for the job.

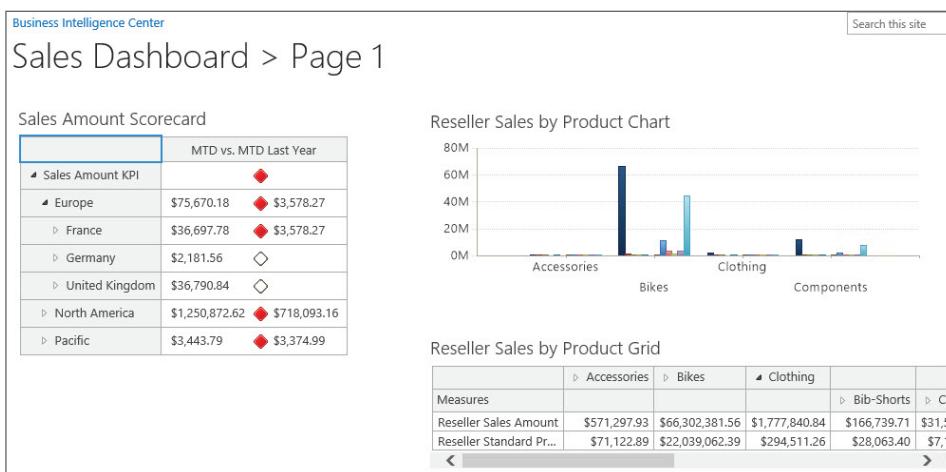


Figure 14-5: PPS dashboard

PerformancePoint Services has many functions and strengths that make it an ideal tool for developing your dashboards and other high-level reports. First, each component of a dashboard in PPS can be modified. For example, Figure 14-6 shows how just a single right-click easily transforms a bar chart into a stacked bar chart. In the same manner, you can even change which measures or attributes you want to view in the report. The Decomposition Tree can also be accessed simply by right-clicking a report region and selecting Decomposition Tree from the menu, giving the user the ability to decompose the measure to further explore the data. When your users need flexible dashboards that can be tweaked, explored, and drilled down in just a few clicks, PPS accomplishes these goals beautifully.

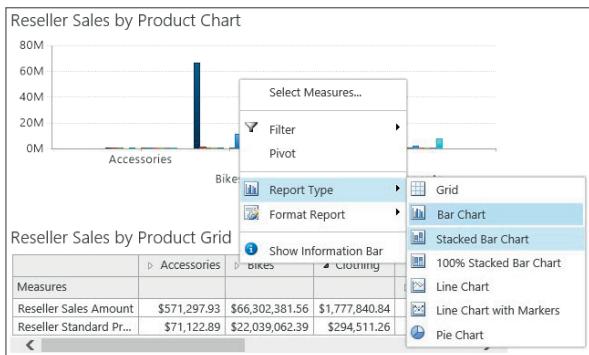


Figure 14-6: Changing chart types in Performance Point

Implementing PPS Requirements for SharePoint

To develop PerformancePoint Services content, you must install SharePoint 2010 or 2013. Also, you must meet the following requirements to configure PerformancePoint Services:

- Install ADOMD.net Version 11.
- Configure the Secure Store if you want the PerformancePoint Services unattended services account to access your data sources.
- Configure the Kerberos constrained delegation if you want a user's credentials to access a data source.

The easiest way to configure PerformancePoint Services is to create a new SharePoint site using the Business Intelligence Center template. The BI Center site is designed specifically for business intelligence content and thus comes configured for business intelligence.

To create your Business Intelligence Center Site, complete the following steps:

1. **Navigate to Central Administration.** Now select Create site collections underneath Application Management.
2. **Next to Title and Description, give your site a name.** This name will appear on each page of the BI Center site.
3. **Specify the website address.**
4. **Under Template select, click the Enterprise tab.** Now select Business Intelligence Center.
5. **Next to Primary Site Collection Administrator, specify the administrator for the BI Center site collection.**
6. **Click OK.**

After a few minutes, your BI Center site should now be created. You can see a menu item on the left for PerformancePoint content, shown in Figure 14-7.

The screenshot shows a SharePoint site titled "Ch14". On the left, there's a navigation bar with links to Dashboards, Data Connections, Libraries, PerformancePoint Content, and Site Contents. The main content area has a heading "Ch14" and a sub-heading "Empower the people in your organization to gain insights with ease using familiar tools. [Learn More](#)". Below this are three sections: "Explore and Analyze Data" (with a diagram of data tables), "Design Interactive Reports" (with a map and chart), and "Share Dashboards" (with a bar chart).

Figure 14-7: BI Center site

On the PerformancePoint Content page of your site, select the PerformancePoint ribbon near the top of your browser. You can then open the Dashboard Designer to begin developing PerformancePoint content. See Figure 14-8 for an example.

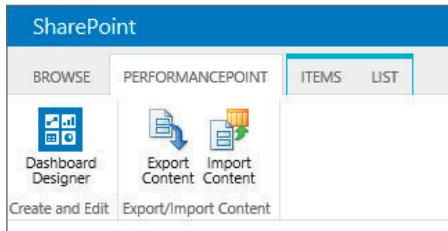


Figure 14-8: Ready to develop PerformancePoint content

Extending PPS Dashboards

Performance Point Services gives your users the ability to interact with the data and reports in unique ways, allowing the users to quickly gain insights into the status of the business. Users have the ability to perform time-based analytics, drill down and through reports, change measures, add comments, and more. This section discusses how to add these advanced features to your Performance Point Services dashboard reports.

Adding PerformancePoint Time Intelligence

Leveraging PerformancePoint Services as your dashboard tool of choice gains you the ability to create advanced, dynamic time intelligence metrics by using a simple yet easy-to-learn expression language referred to as *Simple Time Period*

Specification (STPS). STPS functions are simple and flexible but allow you to create some very powerful calculations. For an example of Time Intelligence formulas using STPS, visit [http://technet.microsoft.com/en-us/library/ff701696\(v=office.15\).aspx](http://technet.microsoft.com/en-us/library/ff701696(v=office.15).aspx).

The following section takes a look at configuring a PPS data source for Time Intelligence, and creating a dynamic Time Intelligence Filter that allows you to apply powerful time filters to a dashboard report. You will be using the SQL Server Analysis Services AdventureWorks multidimensional cube.

Although SQL Server Analysis Services (SSAS) is not the only data source you can connect to within PPS, SSAS provides a much deeper functionality and more flexibility.

1. **On your newly created BI site, open the Dashboard Designer from the PerformancePoint Services ribbon.** This is where you develop any PPS content.
2. **In the Workspace Browser window, right-click the Data Connections folder.** Select New Data Source.
3. **Select Analysis Services.** Click OK.
4. **In the Data Source editor, next to Standard Connection, enter the name of your SQL Server Analysis Services server.** Select the AdventureWorks database from the drop-down list and also select the name of the cube.
5. **Right-click the name of your data source in the Workspace Browser window.** Rename it, as seen in Figure 14-9.

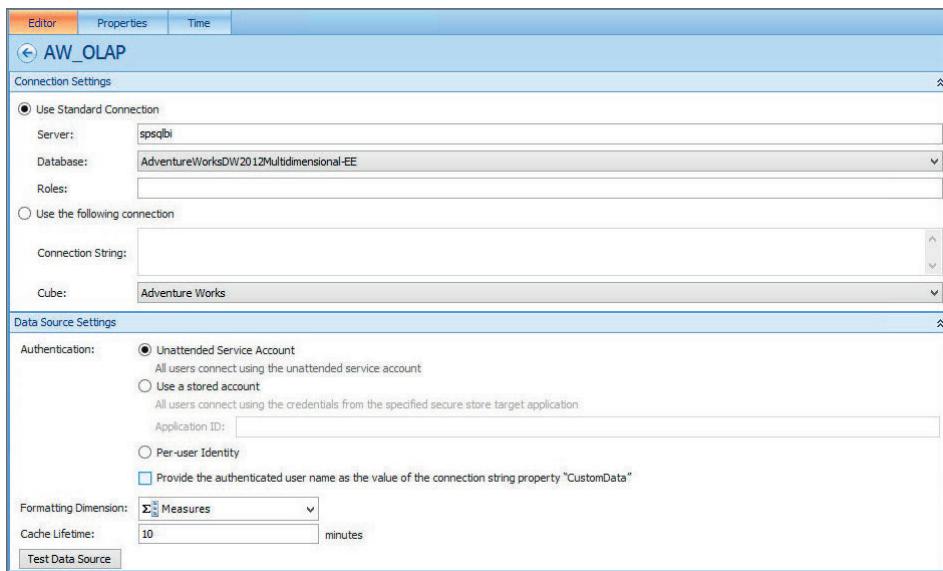


Figure 14-9: Creating your data source

6. **Select the Time tab in the Dashboard Designer.** In the drop-down list underneath Time Dimension, select Date.Date.Calendar. This is the time dimension that you use for your time calculations.
7. **Under Reference Member, select January 1, 2007.** This serves as a reference point between PPS and the Time Dimension. It's important to select the first day of the year. For example, if you're working with a fiscal calendar, select the first day of the fiscal calendar. Also, select Day as the Hierarchy level.
8. **In the Member Associations section, map the Member Levels to the correct Time Aggregations as seen in Figure 14-10.** Click Save.

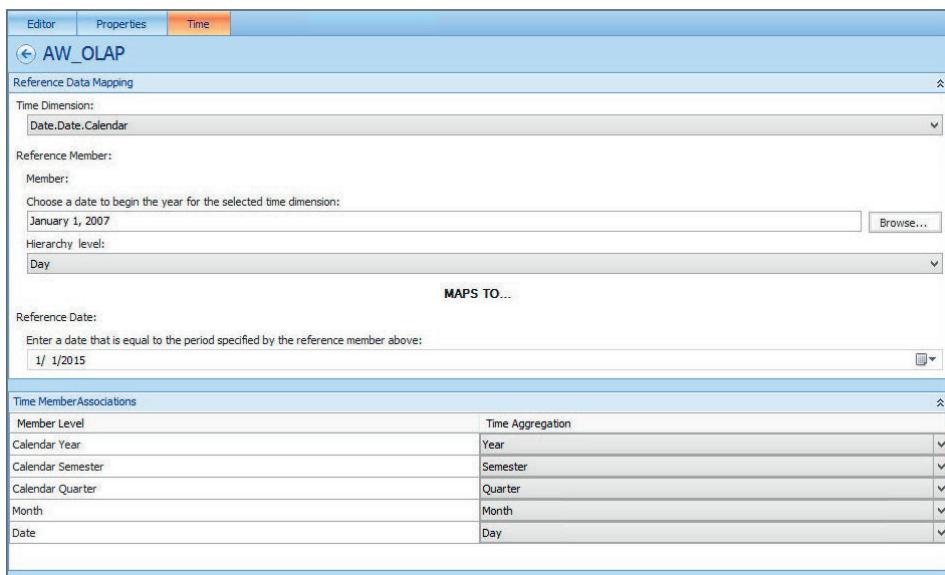


Figure 14-10: Configuring a data source for Time Intelligence

9. **Create the Filter.** In the Workspace Browser, right-click PerformancePoint Content, select New, then select Filter. Highlight Time Intelligence and then click OK.
10. **In the Create a Filter dialog box, click Add Data Source.** Highlight the data source previously created and click OK. Then click Next.
11. **Using Figure 14-11 as a guide, enter the Formulas and Display Names.**

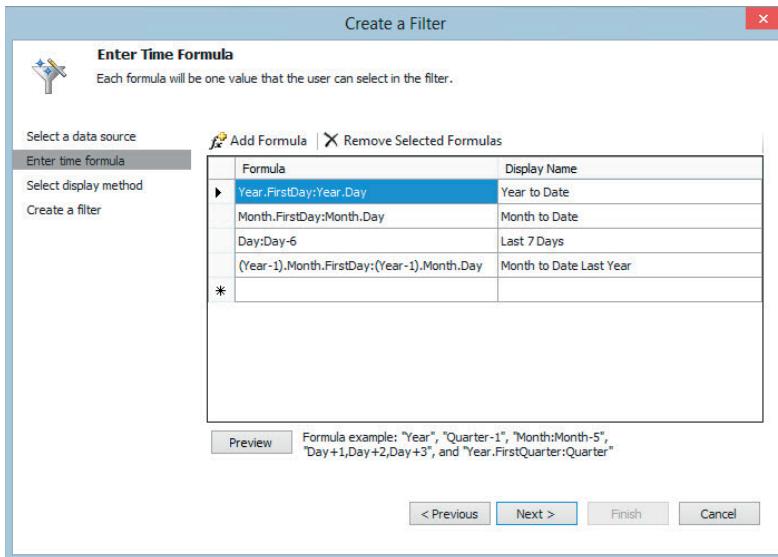


Figure 14-11: Using STPS to create Time Intelligence formulas

12. Click Preview to verify that your formulas are correct. You can see this in Figure 14-12. Click Close.

Time Intelligence Preview	
AW_OLAP	Members
▶ Year.FirstDay:Year.Day	[Date].[Calendar].[Date] & [20070101].[Date].[Calendar].[Date] & [20070102].[Date].[Calendar].[Date]
Month.FirstDay:Month.Day	[Date].[Calendar].[Date] & [20070101].[Date].[Calendar].[Date] & [20070102].[Date].[Calendar].[Date]
Day:Day-6	[Date].[Calendar].[Date] & [20070110].[Date].[Calendar].[Date] & [20070111].[Date].[Calendar].[Date]
(Year-1).Month.FirstDay:(Year-1)	[Date].[Calendar].[Date] & [20060101].[Date].[Calendar].[Date] & [20060102].[Date].[Calendar].[Date]

Figure 14-12: Previewing the Time Intelligence formulas

13. Select List and click Finish. Rename the Filter as Time Intelligence Filter.

With the Time Intelligence Filter created, you are now ready to add the filter to your dashboards, thus enabling users to dynamically filter the dashboards to various custom date ranges.

Using Interactivity Features

Now let's take a look at adding some interactive features to your Performance Point dashboards, which will allow your users to interact with their reports in unique and interesting ways. This section discusses some of the interactive

features of Performance Point, such as drilldown and drillthrough functionality, dynamic measures, and comments, and discuss how we can implement each of these features.

Invoking Drilldown and Drillthrough Functionality in PerformancePoint

The PerformancePoint Services interactive options are accessed via context menus by right-clicking an Analytic Chart, Analytic Grid, or Scorecard report. These context menus give users a variety of ways to explore the data and customize the report to their liking. The area of the report you right-click determines the interactivity options that appear.

By right-clicking the bar or line of an Analytic Chart, the contextual menu shown in Figure 14-13 appears, giving the user the following options:

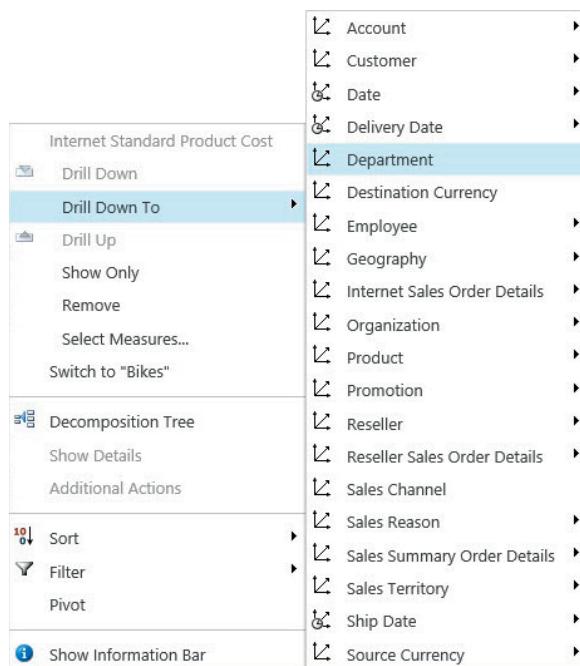


Figure 14-13: Using the contextual menu to customize the report

- Drill down to another dimension attribute.
- Show only the member selected.
- Remove the member selected.
- Select measures to add or remove from the report.
- Switch the context menu to modify another dimension attribute in scope.

- Display a Decomposition Tree report.
- Use additional actions designed in a SQL Server Analysis Services cube.
- Sort the data.
- Filter the data.
- Pivot the report.
- Show the information bar of the report.

To access the second contextual menu of the Analytic Chart, right-click the chart outside of the bar or line. The context menu in Figure 14-14 gives the user the following options:

- Select measures to add or remove from the report.
- Filter the data.
- Pivot the report.
- Change the report type.
- Format the legend of the chart.
- Show the information bar of the report.

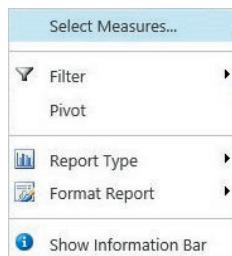


Figure 14-14: This contextual menu allows you to change the report type.

You can also access the contextual menus previously mentioned—with the same options available—from an Analytic Grid report type by right-clicking a detail cell within the grid, a column, or a row header.

Some contextual menu options depend on the design of your SQL Server Analysis Services (SSAS) cube. If no action is defined in the design of the SSAS cube, the Additional Actions option in the contextual menu will be unavailable.

Implementing Dynamic Measures

As you've already seen, PerformancePoint Services gives users multiple ways to interact with the Analytic Grids and Analytic Charts that their dashboard reports may include. By offering the ability to dynamically modify which measures appear in the reports, you can provide a reporting solution that gives users the ability to quickly and easily analyze their important metrics in just a few clicks.

The first way to dynamically modify a report from within the dashboard is to use the previously discussed context menu. Right-click a bar or line in an Analytic Chart or a detail cell in an Analytic Grid and click Select Measures. The Select Measure window appears, allowing the user to easily change which measures to view, as shown in Figure 14-15.

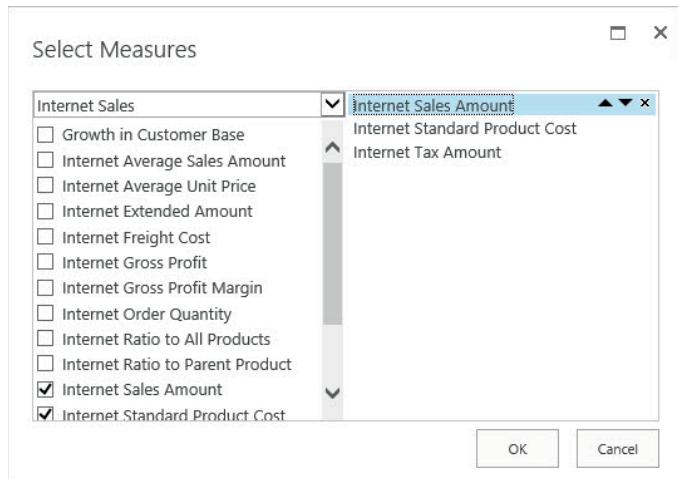


Figure 14-15: Easily modify which measures to view.

The second way to dynamically change which measure appears in a dashboard is to create a connection between a Scorecard and the Analytic Charts and/or Analytic Grids within the dashboard. By creating this connection, a user can click the KPI on a Scorecard and change the measures that appear in the other reports. The following example examines how to link a Scorecard to other reports.

1. **Open the KPI included in the Scorecard you'll be using in your dashboard.** Go to the Properties tab.
2. **Create a custom property.** You can do so by clicking the Create New Property button. Make sure you select Text as the type and click OK.
3. **Type the exact MDX unique name for the measure of the KPI you want to pass to the other reports in your dashboard.** If you make a mistake here, your dynamic measure filtering will not work correctly. See Figure 14-16 as an example. Save the KPI.
4. **Create a connection between the Scorecard report and the other reports in the dashboard.** Open your dashboard. Click the black drop-down arrow in the top right of the dashboard zone that contains the report you want to link to the Scorecard. Click Create Connection (see Figure 14-17).

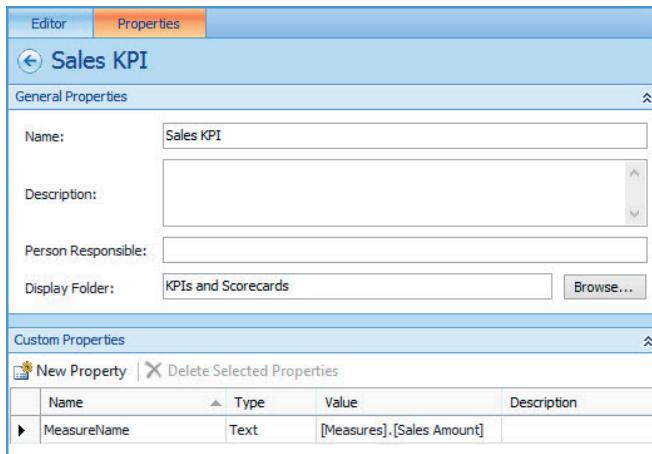


Figure 14-16: Creating the customer property called MeasureName

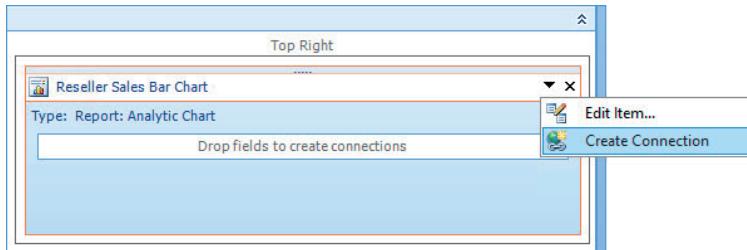


Figure 14-17: Clicking Create Connection to link the Scorecard to the report

5. Select the zone that contains your Scorecard. In the Connection dialog box select the zone from the Get values from drop-down list, as shown in Figure 14-18.

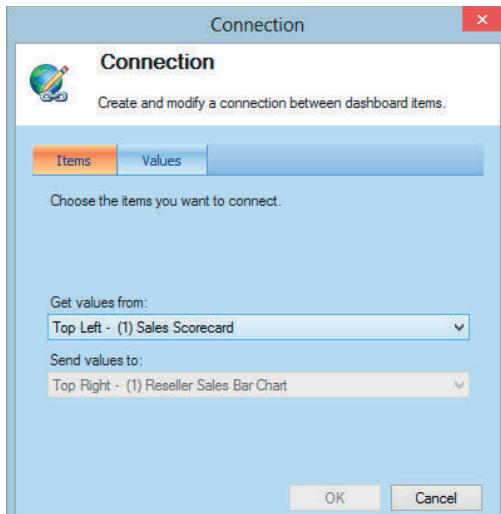


Figure 14-18: Selecting the zone that contains your Scorecard

6. Click the Values tab. Configure it as shown in Figure 14-19, then click OK.

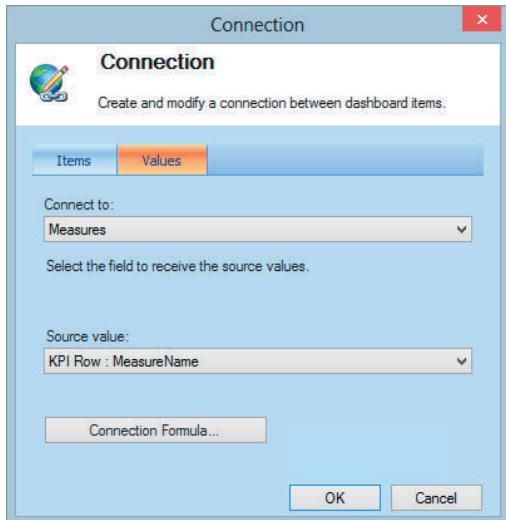


Figure 14-19: Configuring the Values tab

7. Follow Steps 1 to 6 to configure any other reports in your dashboard.
8. Test the functionality. Deploy the dashboard from the PerformancePoint Dashboard Designer by right-clicking your dashboard in the Workspace Browser and clicking Deploy to SharePoint.
9. After your dashboard is deployed, all you have to do is invoke the dynamic measure functionality. You do so by clicking the name of the measure on the Scorecard.

Now your dashboard is configured to allow your users to dynamically modify which measure appears in the reports by simply clicking the name of the KPI in the dashboard's Scorecard. If you've configured the custom property and the connection between reports correctly, you should see the measure in the linked reports dynamically change.

Adding Comments to Scorecards

While viewing a dashboard report, a user has the ability to add a comment to a Scorecard. Imagine a manager is viewing a Scorecard-type report, and he or she has a question regarding a certain KPI. The manager can directly enter the comment into the Scorecard for other team members to review. Team members can then review the manager's comment and respond with another comment, thus answering the manager's question.

You can add a comment to a Scorecard in two ways: from within Dashboard Designer, or while viewing the dashboard report in your browser by using the following steps.

- 1. Add comments to the Scorecard report.** To add a comment, in Dashboard Designer, simply right-click the cell to which you want to add the comment, as shown in Figure 14-20, and select New Comment.

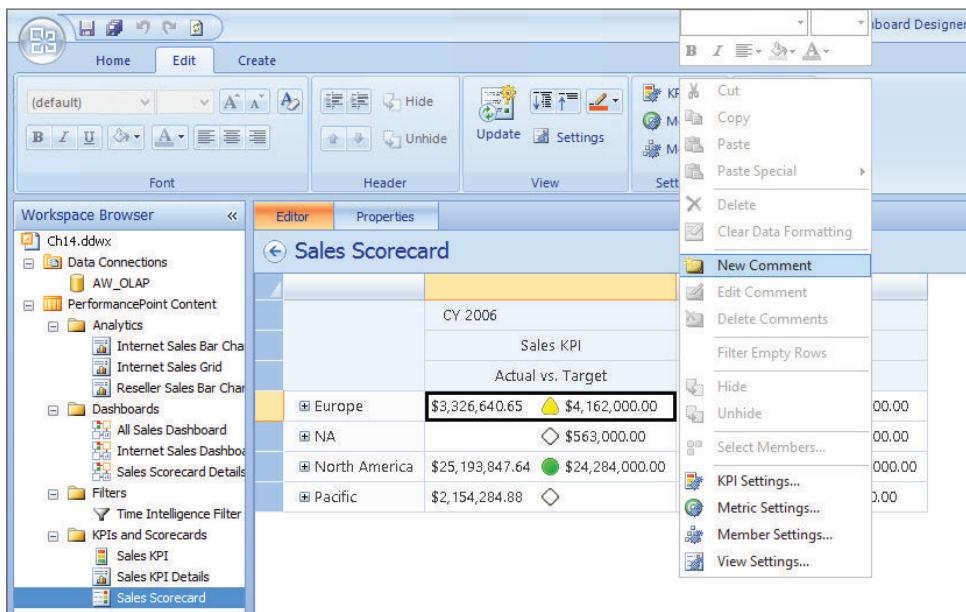


Figure 14-20: Adding comments in Dashboard Designer

- 2. In the Comments dialog box, click New Comment.**
- 3. Give your comment a name and enter your comment into the Description field. Click OK and save your Scorecard. Your comment will now be available for viewing within the dashboard report.**

To add a comment to a Scorecard from within your browser, perform the following steps.

- 1. Right-click the cell to which you want to add the comment. Now select Comments.**
- 2. In the Comments dialog box, click Add Comment.**
- 3. Enter the title of the comment and the comment itself, then click Post.**

After you have added a comment to a Scorecard, a red indicator appears at the top right corner of the affected cell, as shown in Figure 14-21.

Sales Scorecard		
	CY 2006	CY 2007
	Sales KPI	Sales KPI
	Actual vs. Target	Actual vs. Target
▷ Europe	\$3,326,640.65 ▲ \$4,162,000.00	\$9,015,795.82 ● \$9,376,000.00
▷ NA	◇ \$563,000.00	◇ \$1,367,000.00
▷ North America	\$25,193,847.64 ● \$24,284,000.00	\$29,096,718.73 ● \$27,172,000.00
▷ Pacific	\$2,154,284.88 ◇	\$3,881,215.18 ● \$867,000.00

Figure 14-21: The red comment indicator now appears.

The comments are viewable from the Scorecard in the same way that you added the comments to the Scorecard.

Adding Reporting Services Reports to PerformancePoint

SQL Server Reporting Services (SSRS) reports are developed using SQL Server Data Tools or Report Builder and are published to Reporting Services. Reporting Services reports can contain tables, charts, maps, and KPIs, making Reporting Services reports an excellent choice as a reporting technology, whether the report is displaying detail- or dashboard-level data. SSRS reports can also be highly interactive, featuring drilldown actions, drillthrough actions, filters, and images.

Because of their versatility, SSRS reports are an excellent choice for displaying pre-canned data in your PerformancePoint dashboard reports. You can add an existing SSRS report to a PerformancePoint dashboard report using a PerformancePoint Web Part.

In order to add an SSRS report to a dashboard, you must know the name of the report, the server location of the report, the names of any parameters within the report you want to utilize, and the server mode you'll use for the report.

To add an existing Reporting Services report to your dashboard, perform the following steps:

- 1. Open the Dashboard Designer.** In the Workspace Browser, right-click PerformancePoint Content, select New, and click Report.
- 2. Select Reporting Services and click OK.**
- 3. If your SSRS report is deployed to SharePoint, in the Server Mode dropdown list, select SharePoint Integrated.** Select Report Center if your SSRS report is deployed to Reporting Services in Native mode. In this example, the SSRS report is deployed to Reporting Services running in SharePoint Integrated mode.
- 4. In SharePoint Site, enter the URL for the SharePoint site.**

5. In Document Library, select the Document Library that contains your SSRS reports.
6. Next to Report, select your report from the drop-down list. In this example, a report was previously deployed to SharePoint. This report features a single parameter called DateCalendarYear, which appears in the Report parameters window.
7. You can hide the SSRS toolbar when the SSRS report is viewed in the dashboard (see Figure 14-22). To do so, deselect the check box next to Show Toolbar.

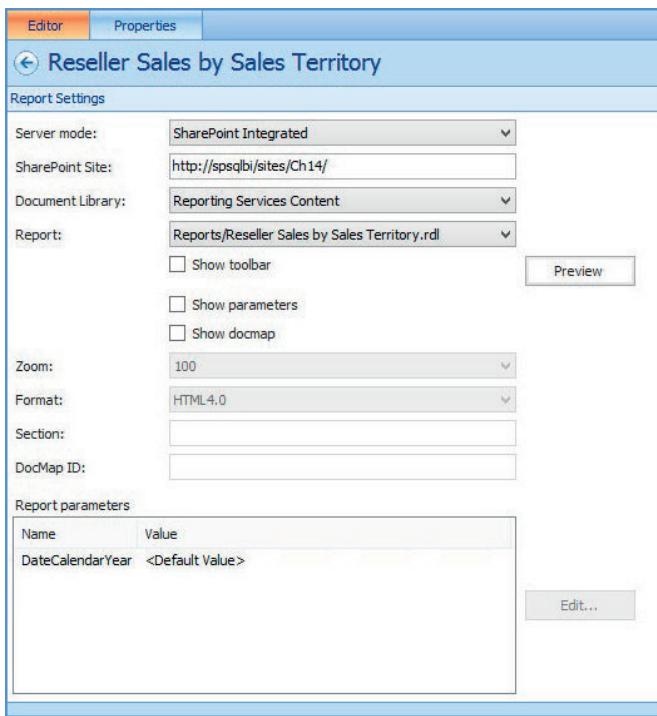


Figure 14-22: Configuring the Reporting Services report

8. Go to the Properties tab. Give your report a meaningful name. Save the report.

Now that the Reporting Services report has been created, you can add the report to a dashboard and even connect the report to other dashboard items such as filters as shown in Figure 14-23.

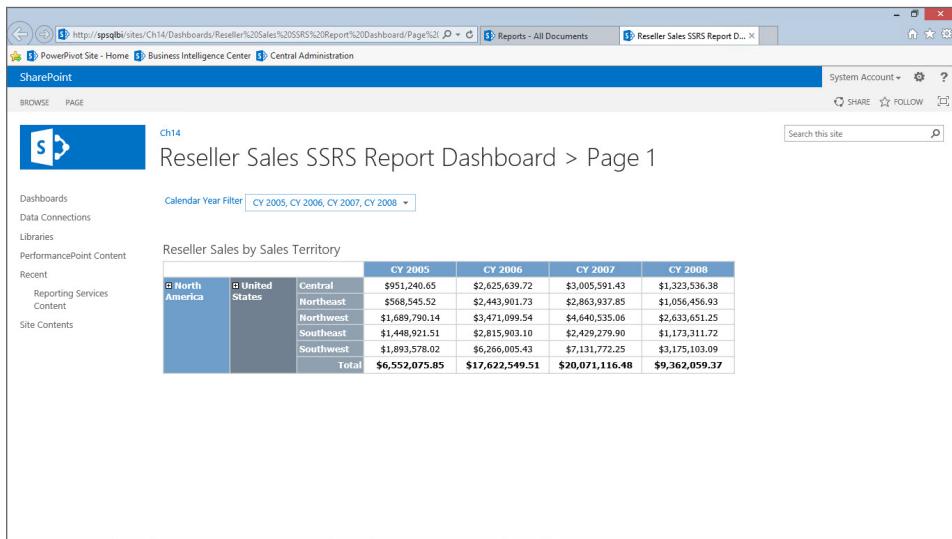


Figure 14-23: Using a Reporting Services report in a PPS dashboard report.

Extending Filters and KPIs

Custom MDX expressions provide the ability to add enhancements and additional functionality to PerformancePoint dashboards that may not otherwise exist within SQL Server Analysis Services cubes. By leveraging the power of MDX—the Analysis Services query and expression language—you can create a custom-named set to use as a filter for your reports. The advantage of using the MDX Query filter is that you can specify exactly the members you want displayed in the filter. The following steps show how to use MDX to create a custom filter using the PerformancePoint Dashboard Designer. In this example, you'll create an MDX Query filter that features members from different levels of the Product Categories hierarchy within the Product dimension.

- Open Dashboard Designer.** In the Workspace Browser, right-click PerformancePoint Content, select New, and click Filter.
- Create a filter list.** Select MDX Query to create a filter list using an MDX query. Click OK.
- Select your data source.** Select the Analysis Services data source you want to use and click Next.
- Enter your MDX set expression.** See Figure 14-24 for an example. Click Next.

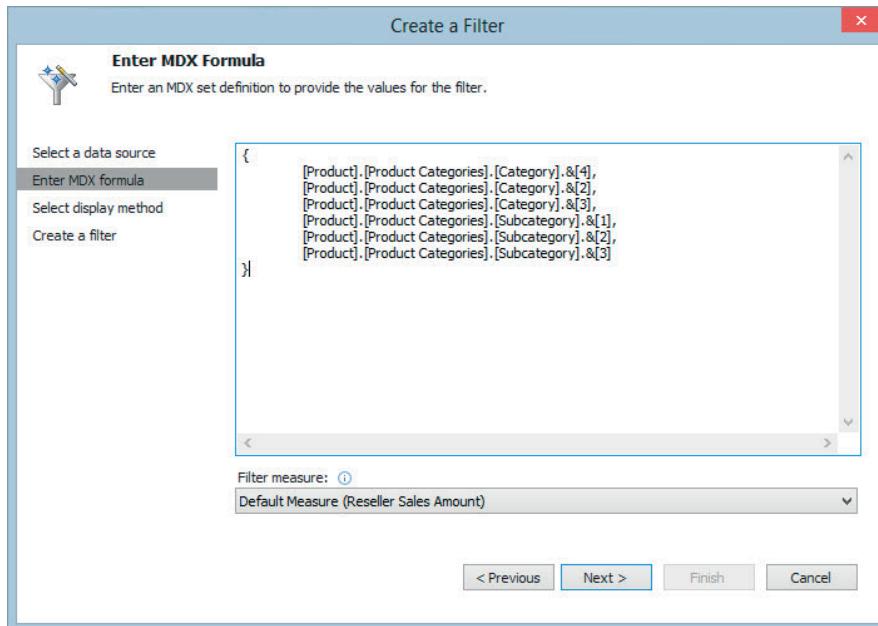


Figure 14-24: A custom MDX set expression

5. Select the Display Method you want to use. Click Finish.

All that's left to do now is to add the filter to a dashboard, and create the connection(s) between the filter and the reports included on the dashboard. In Figure 14-25 you can see how your MDX Query filter appears within the dashboard.

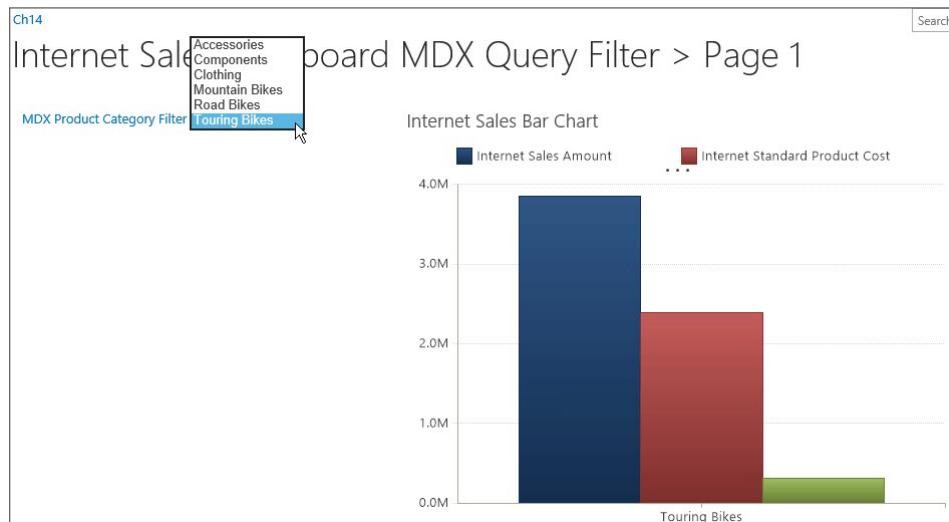


Figure 14-25: The MDX Query filter in action in a dashboard

Occasionally you may encounter a scenario where you need to create a KPI for a metric that does not currently exist within your organization's Analysis Services cube. By using a custom MDX expression, you can create a calculated metric based on measures that already exist in the cube.

In this example, you create a KPI for Internet Sales using an MDX calculation for the Target value.

1. **Create a new KPI.** Open Dashboard Designer. In the Workspace Browser pane, right-click PerformancePoint Content, select New, and click KPI. Select Blank KPI and click OK.
2. **In the Editor tab of the middle pane, click the link under Data Mapping for the Actual value.**
4. **Click Change Source.**
5. **Select your Analysis Services data source pointed at the AdventureWorks cube.** Click OK.
6. **In the Select a measure drop-down list, select Internet Sales Amount.** Click OK.
7. **Click the link under Data Mappings for the Target value.** Then click Change Source.
8. **Click the Calculated Metric tab.** You'll notice you have the option to select from a variety of templates for your calculation, which you can see in Figure 14-26.

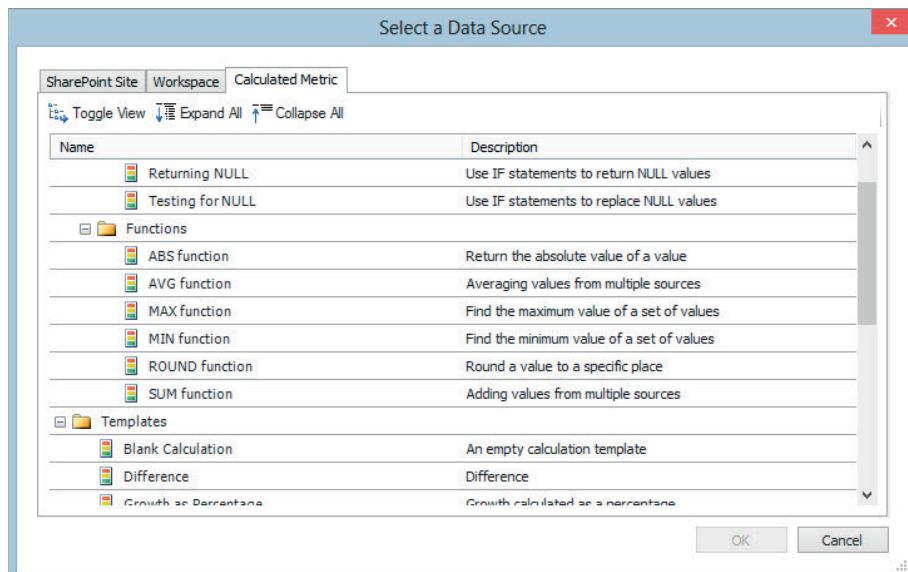
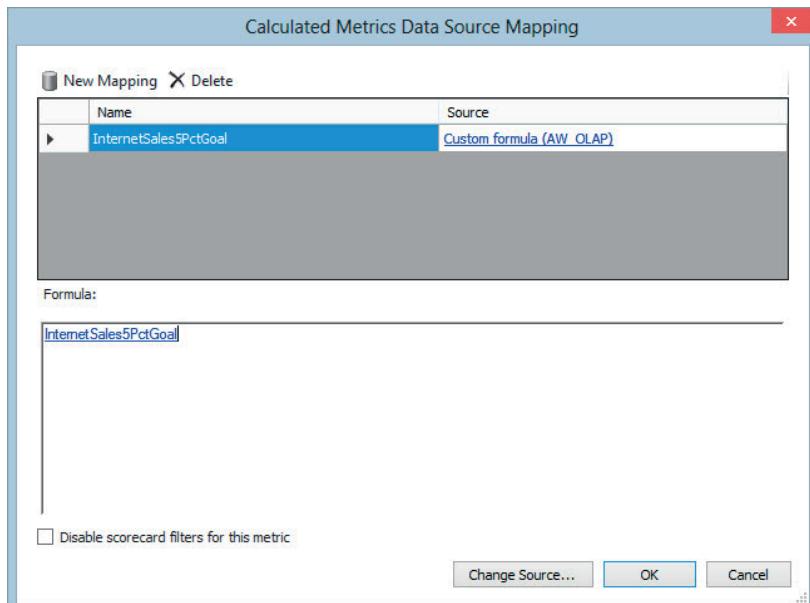


Figure 14-26: Selecting a Calculated Metric template for your data source

9. **Create your own custom calculation and formula.** You do this by selecting Blank Calculation and then click OK.

NOTE In the Calculated Metrics Data Source Mapping, you'll notice two rows in the top section for Value1 and Value2. In this section you can map Measures or calculations to these values. These values are then referenced by name in the Formula section near the bottom of the window.

10. **Highlight Value2 and click Delete.** In this example, you need only one value for your calculation.
 11. **Click the link in the Source column next to Value1.** Then click Change Source.
 12. **Select the Analysis Services Data Connection.** Click OK.
 13. **Deselect the Use MDX tuple formula check box.** Enter the following MDX calculation, which allows you to create a Target that is five percent more than the previous year's Internet Sales Amount:
- ```
(ParallelPeriod([Date].[Calendar].[Calendar Year], 1,
[Date].[Calendar].currentmember),
[Measures].[Internet Sales Amount])*1.05
```
14. **Click OK.**
  15. **Rename Value1 to InternetSales5PctGoal.** In the Formula text box, type the new name for your value, **InternetSales5PctGoal** (see Figure 14-27) and click OK.



**Figure 14-27:** Creating a calculated metric for a KPI

16. In the Editor tab for your KPI, change the Number Format to Currency. You do this by clicking the link next to Target in the Number Format column. Select Currency from the Format drop-down list and click OK.
17. Adjust the thresholds if necessary. Now your KPI is ready to be used in a Scorecard report, as you can see in Figure 14-28.



**Figure 14-28:** Using a Custom MDX KPI in a Scorecard report

## Deployment Best Practices

Following deployment best practices will make development of your PerformancePoint content easier and provide a positive experience for your users. This section reviews the recommended best practices regarding data connections, content libraries, deployments across environments, and the customization of PPS Web Parts.

### Following Best Practices for PerformancePoint Data Connections and Content Libraries

PerformancePoint Services is a powerful and flexible dashboard reporting tool that can give your organization insight into their business processes in some unique and effective ways. To get the most out of PPS, we recommend the following best practices regarding PerformancePoint Data Connections and Content Libraries:

### ***Migrating Required Data to SSAS***

If you do not currently have your data available in a multidimensional format via SQL Server Analysis Services, consider migrating the required data into SSAS. This allows you to take full advantage of the features included in PPS. The format and quality of the data are big considerations, and in our experience, SSAS is the best format to use when designing a PPS reporting solution. SSAS is designed for providing data in an aggregated format and so is PerformancePoint Services.

### ***Properly Formatting Time and Date Dimensions***

When working with SSAS as a data source, make sure that your cube has a properly formatted Time and Date dimension, with members beginning at the first day of a calendar year. This ensures that PPS interacts with the data correctly when using PPS Time Calculations.

### ***Staying at a Grain No Lower Than a Day***

When you are working with time intelligence formulas, we recommend staying at a grain of no lower than a day. You may also choose to report at an hourly level, but any lower than this and you may see performance issues as well as issues with real estate within your dashboard. Remember, PPS is a dashboarding tool and is not typically used for low-level granularity reporting.

### ***Using Friendly Names***

To ensure your users have the best experience possible with your PPS content, make sure that your dimensions, measures, and attributes within your SSAS cube use friendly names. PPS does not provide much ability to customize field and measure names.

### ***Creating Views for Navigation***

As the amount of PerformancePoint Content stored within SharePoint grows, consider creating views to make navigating through your organization's PPS content library easier. For example, you may create a view to display only dashboards, and then set this view as the default view of your PPS content library to prevent users from being overwhelmed by the amount of PPS content. Your users will really appreciate this one. See Figure 14-29 for an example.

The screenshot shows the SharePoint interface for the 'PerformancePoint Content' library. The left navigation bar includes links for Dashboards, Data Connections, Libraries, PerformancePoint Content (selected), Recent, Reporting Services Content, and Site Contents. The main content area displays a list of items with columns for Image, Title, Description, and Display Folder. The items listed are: Internet Sales Dashboard, All Sales Dashboard, Dynamic Measures Dashboard, Sales Scorecard Details, New Dashboard, Reseller Sales SSRS Report Dashboard, and Internet Sales Dashboard MDX Query Filter. A search bar at the top right allows filtering by 'Dashboards Only', 'All Items', or 'By Content Type'.

| Image | Title                                       | Description | Display Folder |
|-------|---------------------------------------------|-------------|----------------|
|       | Internet Sales Dashboard                    | ...         | Dashboards     |
|       | All Sales Dashboard                         | ...         | Dashboards     |
|       | Dynamic Measures Dashboard                  | ...         | Dashboards     |
|       | Sales Scorecard Details                     | ...         | Dashboards     |
|       | New Dashboard                               | ...         | Dashboards     |
|       | Reseller Sales SSRS Report Dashboard        | ...         | Dashboards     |
|       | Internet Sales Dashboard MDX Query Filter * | ...         | Dashboards     |

**Figure 14-29:** Using views to limit the amount of PPS content displayed at one time

### **Grouping Data Connections**

Consider grouping your data connections by connection type in your content library using folders, as shown in Figure 14-30. This becomes especially useful as the amount of data connections your PPS solution uses grows.

The screenshot shows the SharePoint interface for the 'Data Connections' library. The left navigation bar includes links for Dashboards, Data Connections (selected), Libraries, PerformancePoint Content, Recent, Reporting Services Content, and Site Contents. The main content area displays a list of items with columns for Title, Name, and Description. The items listed are: Excel Connections, PowerPivot Connections, and SSAS Connections. A search bar at the top right allows filtering by 'All Items', 'All Info', or 'Approve/Reject'.

| Title | Name                   | Description |
|-------|------------------------|-------------|
|       | Excel Connections      | ...         |
|       | PowerPivot Connections | ...         |
|       | SSAS Connections       | ...         |

**Figure 14-30:** Using folders to make navigating your data connections easier

### **Deploying Dashboards Across Dev, Test, and Production Environments**

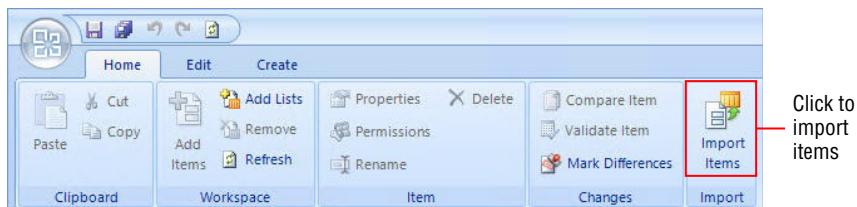
For any business intelligence solution, it's imperative to have a development, testing, and production environment in place to ensure proper development, testing, and validation procedures are followed. PerformancePoint Services BI

solutions are no different. By using a proper development, testing, and production landscape, you can ensure that only proven and vetted reports are deployed to production, giving them the best possible information to run the business.

To configure a proper development, testing, and production environment, create a separate Business Intelligence Center site collection for each environment using the steps outlined earlier in this chapter in the section “Implementing PPS Requirements for SharePoint.”

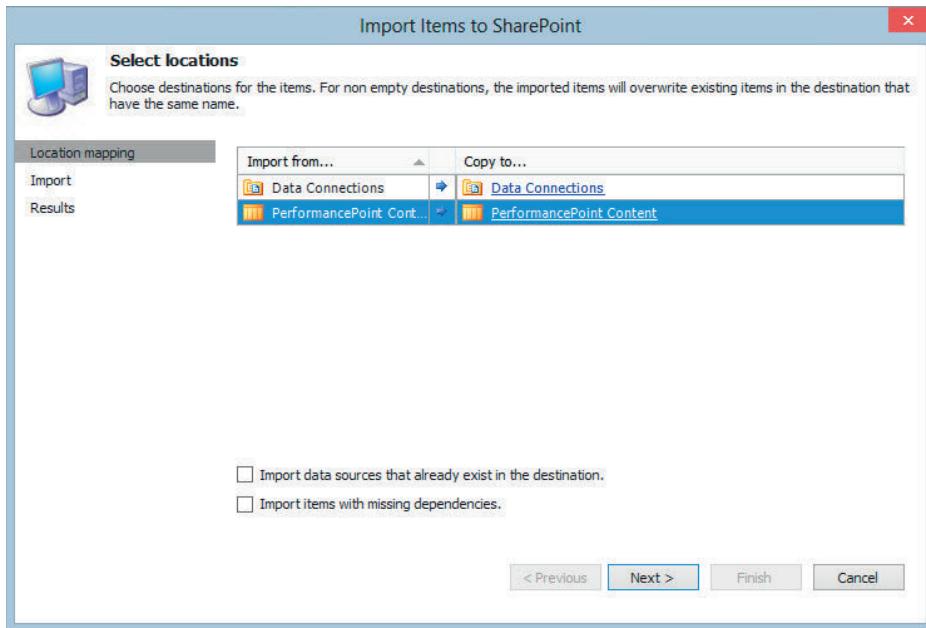
Once the site collections for the different environments are configured, it becomes easy to migrate content using a saved DDWX file that contains the objects to migrate. Follow these steps to import PerformancePoint content into a new site:

- 1. Open Internet Explorer and navigate to the Business Intelligence Center site that you want to use to import your PerformancePoint content. Click the PerformancePoint Content link on the left.**
- 2. On the PerformancePoint Content page, click the PerformancePoint ribbon at the top and click Dashboard Designer. This opens the Dashboard Designer for the current environment.**
- 3. In Dashboard Designer, click Import Items on the Home ribbon at the top. This is shown in Figure 14-31.**



**Figure 14-31:** Importing PPS content

- 4. Navigate to the DDWX file that contains the PPS content you want to import. Select the DDWX file and click Open. This opens the Import Items to SharePoint wizard.**
- 5. In the Copy to column, specify the location where you want to copy any Data Connections and PerformancePoint Content. See Figure 14-32.**
- 6. Make sure you do not check the box next to Import data sources that already exist in the destination. This is important, so that any data sources pointing to specific production, testing, or development environments are not overwritten. Click Next.**



**Figure 14-32:** Choosing destinations for PPS content

7. Click Finish after the PPS content has finished importing.
8. Deploy the content. When you're ready to deploy the content, simply right-click the dashboard you want to deploy and select Deploy. Your PPS content then deploys to the new environment.

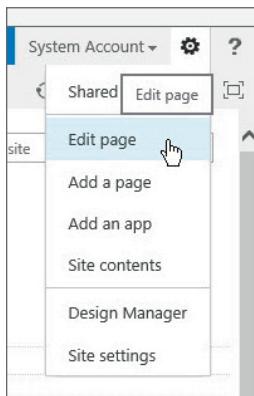
## Customizing PerformancePoint SharePoint Web Parts

When a dashboard is deployed, the dashboard report is implemented as individual SharePoint Web Parts on a Web Part page. These Web Parts are modular units that contain interactive PerformancePoint reports and Scorecards.

Because the PPS dashboard is implemented as a native SharePoint Web Part page, you can edit and customize the Web Part page using the SharePoint web-based page editor right from your Internet browser.

After you've deployed your dashboard report, you can edit the report by clicking the Settings button at the top right of the page, as shown in Figure 14-33.

Once the Web Part Page Editor has opened, you can do everything from modifying existing zones, swapping out a PPS report for a different report, creating connections between PPS reports, as well as adding new zones for additional PPS content.



**Figure 14-33:** Editing a Web Part page

You can modify an existing Web Part, click the drop-down arrow next to Web Part and select Edit Web Part (see Figure 14-34). This opens the Web Part properties window.

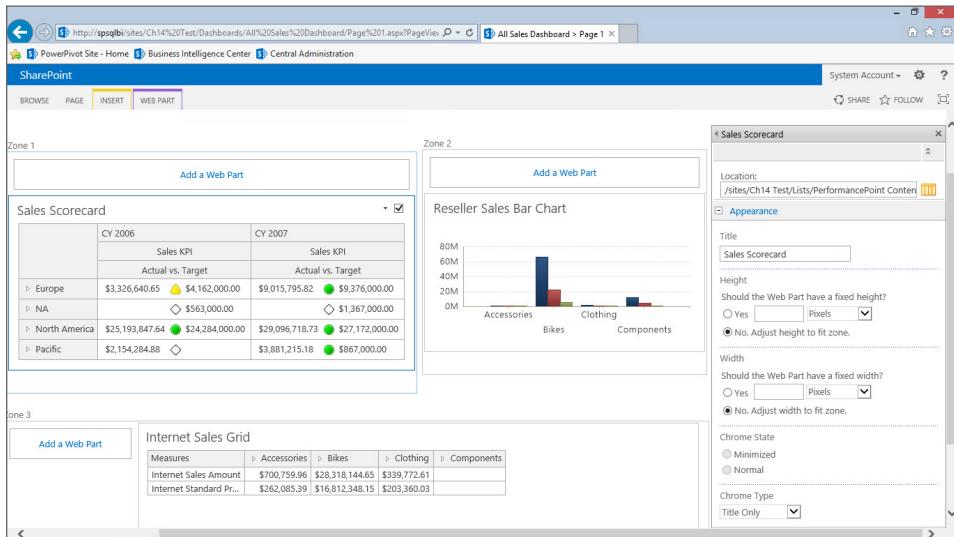
A screenshot of a SharePoint dashboard titled 'Zone 1'. It features a 'Sales Scorecard' report. On the right side of the report, there is a context menu with several options: 'Delete', 'Edit Web Part' (which is highlighted with a blue selection bar and has a hand cursor icon), 'Edit Web Part (h)', and 'Connections'. The 'Edit Web Part (h)' option is also preceded by a blue selection bar and a hand cursor icon. The main content area shows a table with data for four regions: Europe, NA, North America, and Pacific, comparing CY 2006 and CY 2007 sales figures.

**Figure 14-34:** Modifying an existing Web Part

If you want to change which PerformancePoint report is shown in the selected Web Part, click the orange Browse icon next to Location in the properties window, and you can easily select another PerformancePoint report. In the properties window you also have the ability to change the Web Part title, height and width of the Web Part within the zone, and the Chrome Type, which controls the formatting of the Web Part.

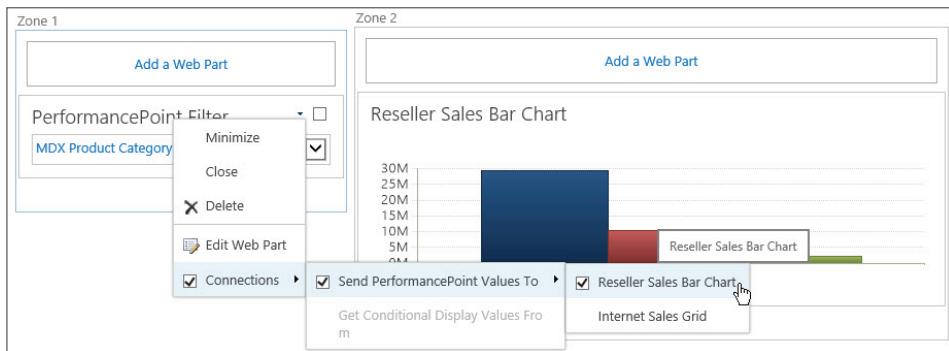
In the following example, you change a report in the dashboard to a filter you created earlier, and then create a connection between the filter and the two remaining reports.

1. Click the Settings button on the SharePoint Web Part page. Click Edit Page.
2. Click the drop-down arrow next to the Web Part you want to change. Select Delete to remove the existing Web Part so you can replace it with a PPS Filter.
3. Click Add a Web Part to add the filter. Navigate to the desired PPS Filter and click Add, as shown in Figure 14-35.



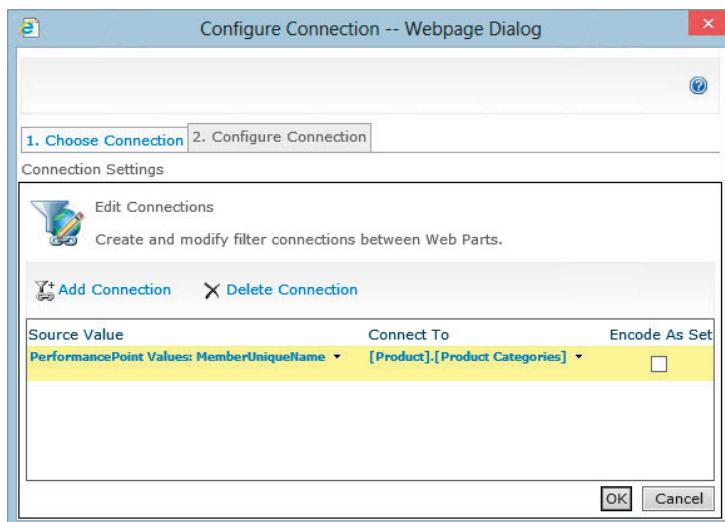
**Figure 14-35:** Adding a new Web Part

4. Open the tool pane. In the modified zone, click the Click here to open the tool pane link.
5. Select the filter you want to use. In the properties window on the right, click the Browse icon next to Location and navigate to the filter you want to use. Click Insert.
6. Also, change the Chrome Type to None. No title or border is usually recommended for a Web Part, including a PerformancePoint filter. Click OK.
7. Create the connection between the new filter and a report within the Web Part Page. You do so by clicking the drop-down arrow near the PPS filter, selecting Connections, selecting Send PerformancePoint Values To, and clicking the name of the report. In Figure 14-36, the report you'll be sending the filter values to is Reseller Sales Bar Chart.



**Figure 14-36:** Creating a connection between Web Parts

8. **Configure the connection.** After opening the Choose Connection web page dialog box, select Get Filter Values From and click Configure. Click Add Connection.
9. **For the Source Value, select MemberUniqueName, and for the Connect To value, select the corresponding hierarchy.** The hierarchy used in the PerformancePoint filter must also be used in the report to which you are creating the connection so that the values can be passed correctly. Click OK (see Figure 14-37).



**Figure 14-37:** Configuring the connection between Web Parts

10. **Create any remaining necessary connections.** When you've finished editing the Web Part Page, click Stop Editing in the top left of the Page ribbon.

Now you can dynamically filter the reports within your Web Part Page using the new filter you added via the SharePoint Web Part Page editor.

Be aware that any changes deployed from the Dashboard Designer overwrite any changes made using the SharePoint Web Page editor.

## Security and Configuration Best Practices

---

In order to ensure that PerformancePoint Services can interact with your data sources correctly and that your users will have the optimum experience from their PPS report solution, PPS must be correctly configured. This section discusses configuring the Unattended Service Account and how to configure the Performance Point Service application settings by following these suggested best practices.

### Configuring the Unattended Service Account in SharePoint

Before using PerformancePoint Services to connect to any external data sources, you must first configure the Unattended Services Account. This is a special single-shared user account that PPS uses. Whichever account you use as your Unattended Services Account, it should have access to all the various data sources required by your PerformancePoint Services solution.

To configure the Unattended Service Account, you must first configure the Secure Store Service as well. The Secure Store Service ensures that selected data stored within it is encrypted and that no passwords are stored in plain text.

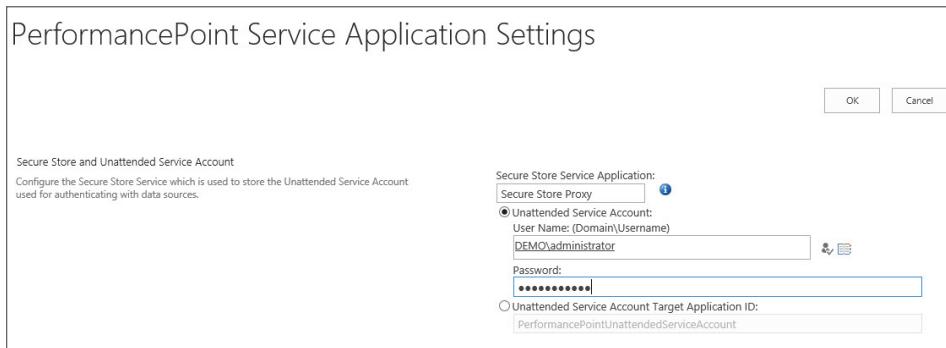
Follow these steps to configure the Secure Store Service:

1. **Open Central Administration.** Click Manage service applications under the Application Management heading.
2. **Select Secure Store Service.**
3. **Within the Edit ribbon, select Generate New Key.**
4. **Enter in a secure passphrase.** Make sure you choose a strong password because sensitive data may be involved.

With the Secure Store Service configured, you are now ready to configure the Unattended Service Account, which you do by following these steps:

1. **Open Central Administration.** Click Manage service applications under the Application Management heading.
2. **Click the PerformancePoint Service Application link.**
3. **Click PerformancePoint Service Application Settings.**

4. Beneath the Unattended Service Account radio button, enter a valid account User Name and Password, as shown in Figure 14-38. Leave the default selections for the remaining settings and click OK.



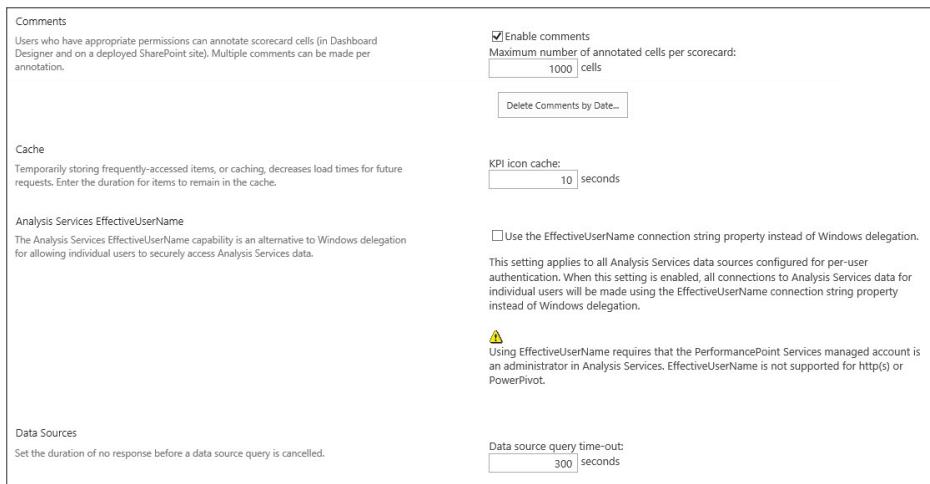
**Figure 14-38:** Configuring the Unattended Service Account for PerformancePoint Services

Once the Unattended Service Account has been configured, you are now ready to begin developing and deploying your PerformancePoint content.

## Optimizing PerformancePoint Services Application Settings

You can find the settings for managing the PerformancePoint Services Application on the Manage PerformancePoint Services page:

1. **Open Central Administration.** Click Manage service applications under the Application Management heading.
2. **Click the PerformancePoint Service Application link.**
3. **Click PerformancePoint Service Application Settings.** With the PerformancePoint Service Application Settings open as shown in Figure 14-39, you can configure the settings to manage the performance impact of various PPS features.



**Figure 14-39:** Optimizing PerformancePoint Service Application Settings

These settings offer the following options and limits to the PerformancePoint Service Application, which affect all site collections within the service:

- **Secure Store and Unattended Service Account:** As covered in the previous section, this is where you specify the Secure Store Service Application and the Unattended Service Account. The credentials for the Unattended Service Account are stored within the Secure Store Service Application so they are not stored in plain text.
- **Comments:** This setting controls limitations on the number of annotated cells per Scorecard. Large numbers of comments can negatively impact the performance of a Scorecard, even if the comments are not displayed. Make sure you configure the Delete Comments by Date setting because comments are never automatically deleted. Configuring this setting ensures that all comments older than a specified date are deleted.
- **Cache:** The KPI icon cache specifies how long a custom indicator is cached. This does not apply to the default indicators.
- **Analysis Services EffectiveUserName:** This setting applies to all SQL Server Analysis Services data sources configured for per-user authentication. When this setting is enabled, the EffectiveUserName property is added to the connection string.
- **Data Sources:** This setting specifies how long a query can run before timing out.
- **Filters:** The Filters settings control the user experience with filters within dashboards. Using these settings, you can manage the number of days a

user's filter selections are remembered, the maximum number of items to load in a filter tree, and the initial search result limit for a filter. These settings are useful for limiting the performance impact caused by having users interact with large dimensions via filters.

- **Select Measure Control:** This setting helps manage the performance implications caused by having users attempting to add many measures to their reports. If you have a cube with more than 1000 measures, you may want to raise this setting higher than the default value of 1000.
- **Show Details:** The Initial retrieval limit setting limits the number of rows initially retrieved when a user invokes the Show Details functionality. The Maximum retrieval limit setting limits the number of rows retrieved during subsequent record retrievals.
- **Decomposition Tree:** This setting manages the number of items retrieved when a user invokes the Decomposition Tree feature. If you find that your users are frequently using the Decomposition Tree and you have some dimension attributes with many members, limiting the number of items that can be retrieved by the Decomposition Tree can help manage the performance impact of this function.

## Summary

---

This chapter examined in-depth how you can leverage some of the advanced functionality of PerformancePoint Services to give your organization better insights into the business in a unique, powerful, and flexible way.

You learned the different report visualizations available in PPS; how to add advanced functionality to PPS dashboards; and deployment, security, and configuration best practices. You also learned about important considerations when deciding if PPS is the best tool to meet the requirements.

After reading through this chapter and working through the examples, you should ideally find yourself better prepared to design highly effective and useful PowerPoint Services content for your organization.

Part

IV

## Deploying and Managing the Business Intelligence Solution

### In This Part

---

**Chapter 15:** Implementing a Self-Service Delivery Framework

**Chapter 16:** Designing and Implementing a Deployment Strategy

**Chapter 17:** Managing and Maintaining the Business Intelligence Environment

**Chapter 18:** Scaling the Business Intelligence Environment



# Implementing a Self-Service Delivery Framework

Creating dashboards with scorecards, reports, graphs, interactive maps, and other elements is fun when your data is ready for analyses. Unfortunately, most of the time in a business intelligence project you deal with suboptimal datasets. Typically, the *data quality* does not meet your expectations. Especially important is to have a good data quality for the most important data in an enterprise, the *master data*.

In order to create meaningful analyses, you need good data. Raising the data quality and maintaining the master data should be a part of a successful business intelligence project. Including the data issues in a project's lifecycle means creating a viable business intelligence solution *delivery framework*, no matter if this is a self-service or an enterprise-level centralized implementation.

## Planning a Self-Service Delivery Framework

---

Maintaining data quality requires good planning for the self-service delivery framework. Included among the information you may need to identify is the following:

- The part of the data in your organization constituting the master data
- The kind of data quality issues that exist in your data

- Who the key people and roles are that manage the most important datasets
- Whether you have appropriate software in place, or whether you need to introduce some new applications specialized for resolving data quality issues and maintaining the master data

Of course, you should also consider the time component: Raising the data quality once does not help much if the quality starts deteriorating immediately after.

## **Creating a Data Governance Plan for Enterprise, Team, and Personal BI**

The *data governance* plan you create should include enterprise, team, and personal levels. Data governance refers to the activities performed to improve and maintain data quality. Data quality projects are very intensive in terms of resources needed. In order to execute a project successfully, you have to show possible benefits to key stakeholders. First, you need to understand the business needs. You can use interviews, overviews of organizational charts, analyses of existing practices in an enterprise, and other such information sources. You must prioritize the business issues and make a clear project plan. For a project to be successful, you should start either with a business area that is very painful for the stakeholders, or with a business problem that is quite simple to solve. You should always implement data quality projects step by step.

### ***Including Overviews***

Before you start any data quality project, you have to understand the sources and destinations of problematic data. Therefore, data quality activities must include overviews. You must make an extensive overview of all the schemas of the databases that pertain to the problem data. You should interview domain experts and users of data. This is especially important for gaining an understanding of the quality of the schema dimensions. In addition, after this step, you should have a clear understanding of the technology that the enterprise is using. If necessary, you might need to include appropriate technology experts in the project. During the overview of the data, you should also focus on the data lifecycle so that you can understand retention periods and similar concepts.

### ***Including Assessments***

The next step is the data quality assessment. You assess different aspects of data quality, called *data quality dimensions*. You can measure some data quality dimensions with tools such as Transact-SQL (T-SQL) queries. Measurable dimensions are called *hard dimensions*. Some of the most important hard dimensions include

completeness, accuracy, and consistency. You measure hard dimensions with procedural analysis of the data, also known as *data profiling*. Many different tools for data profiling exist, and you should exploit all the knowledge you have and all the tools available for this task.

In contrast to hard dimensions, some dimensions depend on the users' perception of the data. These are called *soft dimensions*. You cannot measure soft dimensions directly. You can measure them only indirectly through interviews with data users or through any other kind of communication with users, such as complaints or even informal meetings. Note that this communication can unfortunately include unpleasant events, such as customer complaints—the events you want to prevent. Measuring soft data quality dimensions must be a part of your data governance plan as well. Typical soft dimensions include timeliness, ease of use, trust, and presentation quality.

### ***Improving Low-Quality Data***

After finishing with the data assessment, you should plan to reassess the business impact of low-quality data. You should meet again with key stakeholders to review the priorities and elaborate on the improvement part of the project plan in detail.

An improvement plan should include two parts: *correcting* existing data and, even more important, *preventing* future errors. If you focus on correcting existing data only, you will have to repeat the correction part of the data quality activities regularly. You do need to spend some time correcting existing data, but you should not forget the prevention part. When you have both parts of the improvement plan, you can start implementing it.

Implementation of corrective measures involves automatic and manual cleansing methods. Automatic cleansing methods can include your own procedures and queries. If you have known logic for correcting the data, you should use it. For example, you can solve consistency problems by defining a single way to represent the data in all systems, and then replace inconsistent representations with the newly defined ones. For de-duplication and merging from different sources, you can use string-matching algorithms. For correcting addresses, you can use validation and cleansing tools that already exist in the market and use some registries of valid addresses. However, you should always prepare for the fact that part of the data cleansing must be done manually.

Preventing new inserts of inaccurate data involves different techniques as well. The most important one is to implement a *master data management (MDM)* solution.

Your data governance plan has to include a training plan for the people that create and use the most important data as well. Measuring data quality improvements helps you to realize if you are on the right track with your activities, and

helps you in communicating the results of the activities to the key stakeholders. Therefore, don't forget to include these measurements and communication activities in your data governance plan.

## Identifying Stakeholders, Subject Matter Experts, and Data Stewards

Dealing with data quality issues is never a simple and straightforward process. Some problems you might experience include:

- **Different metadata:** Sometimes it is even hard to define who exactly the customer is.
- **Different presentations of the same data:** This involves determining which presentation is correct.
- **Authority:** You have to find out who owns the source.

You need to solve such questions together with the appropriate people:

- **Key stakeholders:** The people who approve the budget for your activities. They can also help you with the authority issues. Most times these are the managers for whom you are preparing your dashboards and Scorecards. You should try to get full cooperation of the owners of each data source.
- **Subject matter experts:** In order to define the correct metadata and presentations, you need to interview subject matter experts, who define the business processes and drive the design and architecture of the databases and applications in an enterprise. Subject matter experts should be available throughout a data quality part of a business intelligence project.
- **Data steward:** As mentioned, data governance refers to the activities performed to improve and maintain data quality. Data stewards are the people responsible for these activities. For a data quality and/or master data management project to be successful, explicit *data stewardship* roles must be defined. Without explicit responsibility and accountability, it is impossible to maintain high data quality over time.

## Understanding Industry Compliance Considerations

In addition to the data quality or MDM solution, you should also focus on source systems. It is very unlikely that your solution will ever cover all possible master entities with all their attributes. Therefore, part of the data is still going to be maintained in the source—the operational applications. You have to enforce proper data models, constraints, and good user interfaces wherever possible.

An informal industry standard for transactional line of business applications is to use the *relational model* in the databases that support such applications. The

relational model was conceived in the 1960s by Edgar F. Codd, who worked for IBM. It is a simple, yet rigorously defined conceptualization of how users perceive and work with data. The most important definition is the Information Principle.

### INFORMATION PRINCIPLE

The *Information Principle* states that all information in a relational database is expressed in one (and only one) way, as explicit values in columns within rows of a table. In the relational model, a table is called a *relation* and a row is called a *tuple*, which consists of *attributes*.

Each relation represents some real-world entity, such as a person, place, thing, or event. An *entity* is a thing that can be distinctly identified and is of business interest. Relationships are associations between entities. A row in a relation is a *proposition*, such as an employee with an identification number equal to 17 and the full name of Davide Mauri, who lives in the city of Milan. The relation header, or schema of the relation, is the predicate for its propositions. A *predicate* is a generalized form of a proposition, such as an employee with identification EmployeeId(int) and full name EmployeeName(string), who lives in City(CitiesCollection). Note the name/domain pair of placeholders for concrete values. The domain, or the data type, is the first point where a relational database management system (RDBMS) can start enforcing data integrity.

Without a good naming convention, it is hard to reconstruct predicates and propositions from a database schema. With different source systems, you typically do not have influence on the source schema and naming conventions. However, you should be aware that bad schema and naming conventions mean more problems for your MDM project and worse data quality.

### ***What Is Normalization?***

One of the most important tools for enforcing data integrity in a relational model is *normalization*. Tables are normalized when they represent propositions about entities of one type—in other words, when they represent a single set. This means that entities do not overlap in tables and that tables are orthogonal or normal in mathematical terms. When a table meets a certain prescribed set of conditions, it is in a particular normal form. A database is normalized when all tables are normalized.

### ***What Are Generalization and Specialization?***

Another important relational design technique is *generalization* and *specialization*. You can have NULLs in your data because the value is unknown or because an

attribute is not applicable for a subset of rows. For example, you can have people and companies in the Customers table. For people, the birthdate makes sense; for companies, the number of employees might be valuable information. In order to prevent NULLs, you can introduce subtype tables, or subtypes for short.

Two entities are of distinct, or primitive, types if they have no attributes in common. Some relations can have both common and distinct attributes. If they have a common identifier, you can talk about a special *supertype-subtype* relationship. Supertypes and subtypes are helpful for representing different levels of generalization or specialization. In a business problem description, the verb *is* (or explicitly *is a kind of*) leads to a supertype-subtype relationship. Specialization leads to additional decomposition of tables.

### ***What Are Constraints?***

Besides data types and normalization, you can use many additional tools provided by modern RDBMSs like SQL Server. You have *declarative constraints*, an example of which is the Check constraint, which further narrows down possible values of an attribute. You can define whether all values have to be known, or you can prohibit NULLs. (NULL is a standard placeholder for unknown or not applicable values.) You can implement integrity rules programmatically with *triggers* and *stored procedures*. You can implement them programmatically in the middle tier or client tier of an application as well. You can even implement constraints as strings in a table or a file and then build the code that evaluates the strings, thus enforcing the constraints dynamically in your application. The important thing for a MDM project is to understand the source data. Realizing whether the design of operational databases (which are the sources where the data is introduced in an enterprise for the first time) is appropriate helps a lot when evaluating the cost of possible approaches to an MDM solution.

### ***A Checklist for Compliance***

In short, a database that supports a transactional line of business application should use a properly normalized, specialized, and constrained relational model. Things you should remember to make your database compliant include:

- **For analytical applications, the *dimensional model* has become an informal standard.** You can learn more about the dimensional model techniques from the author of the model, Ralph Kimball, and his associates at the Kimball Group site at <http://www.kimballgroup.com/data-warehouse-business-intelligence-resources/kimball-techniques/dimensional-modeling-techniques/>.

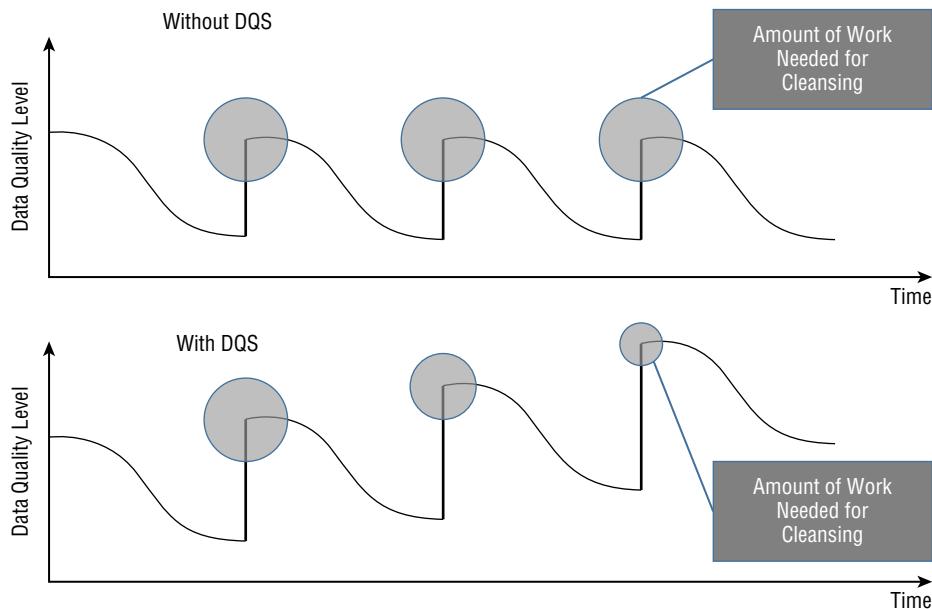
- You should check the schemas of the databases in an enterprise through the aspects known as *schema dimensions*. These dimensions are schema completeness, schema correctness, documentation, compliance with theoretical models, and minimization.
- Besides the schema, you should also check whether the data values comply with industry standards, when such standards exist. Most of the time it is recommended to use such standards, especially when you need to merge data from multiple sources. By standardizing data values like ZIP codes, product SKUs, and more, merging from multiple sources is mitigated a lot.

## Managing Data Quality and Master Data

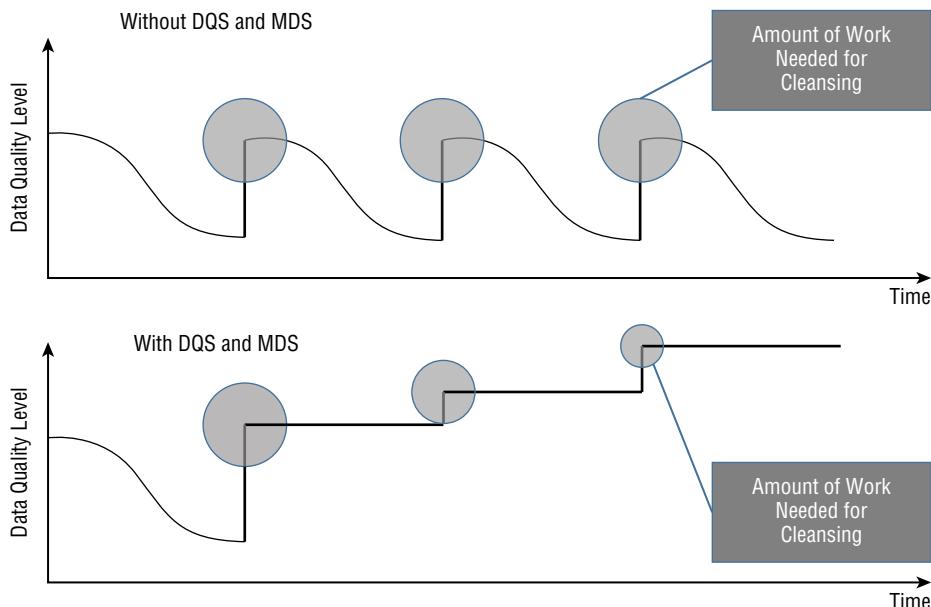
Many companies or organizations do regular data cleansing. This is the most important data quality activity. When you cleanse the data, the data quality goes up to some higher level. The data quality level is determined by the amount of work invested in the cleansing. As time passes, the data quality deteriorates, and you need to repeat the cleansing process. If you spend an equal amount of effort as you did with the previous cleansing, you can expect the same level of data quality as you had after the previous cleansing. And then the data quality deteriorates over time again, and the cleansing process starts over and over.

The idea of specialized data quality applications, such as *Data Quality Services (DQS)*, is to mitigate the cleansing process. Although the amount of time you need to spend on cleansing decreases, you will achieve higher and higher levels of data quality. While cleansing, you learn what types of errors to expect, discover error patterns, find domains of correct values, and so on. You don't throw away this knowledge. You store it and use it to find and correct the same issues automatically during your next cleansing process. Figure 15-1 shows the data quality level over time, with occasional data cleansing, and the time, effort, and resources needed for each of these occasional cleansings without and with DQS.

The idea of master data management applications, what the *Master Data Services (MDS)* is, is to prevent data quality from deteriorating. Once you reach a particular quality level, the MDS application—together with the defined policies, people, and master data management processes—allows you to maintain this level permanently. Figure 15-2 shows the data quality level over time, with occasional data cleansing, and the time, effort, and resources needed for each of these occasional cleansings without DQS and MDS and with DQS and MDS.



**Figure 15-1:** The data quality and the amount of work needed for occasional cleansing without and with DQS



**Figure 15-2:** The data quality and the amount of work needed for occasional cleansing without and with DQS and MDS

Only master data should be maintained by MDS. You need to identify the master data. You should interview subject matter experts to find out which datasets are the most important for an enterprise. A good criteria for defining the master data is reuse.

**DEFINITION** *Master data is frequently reused in many lines of business applications, and represented as dimensions in a multidimensional model in analytical applications. Typical master dataset examples are customers, products, contracts, and similar.*

## Identifying Target Audience and Roles

Key data quality and master data management roles, as well as people needed for planning, namely stakeholders, subject matter experts, and data stewards, were already introduced in this chapter. Additionally, you need to focus on end users; by interviewing end users, you measure the soft data quality dimensions. You can get a lot of insight on the state of the company by comparing hard and soft dimensions:

- If your evaluation of the hard dimensions is bad, but you get good evaluations for the soft dimensions, then the company does not realize that it has problems with data. In such a case, additional assessment of the potential damage caused by the bad data can help key stakeholders understand the data quality problems.
- If both soft and hard dimensions get bad evaluations, then the company is ready for a data quality and/or MDM project.
- If hard dimensions get good evaluations and soft dimensions get bad evaluations, this means that for some reason domain experts and users do not trust the data. Usually this situation arises because of a previous system, a previous version of the system, or a lack of education.
- If both hard and soft dimensions get good evaluations, the company does not need a special data quality project. However, the company could still decide to initiate an MDM project in order to minimize expenses with master data maintenance.

Of course, end users are those who insert and update the master data. Data stewards and IT professionals are typical users of Data Quality Services. However, if you plan to implement Master Data Services in your ecosystem, then you also need to identify the following roles and people:

- Ordinary end users who should use a simple MDS interface, namely the Master Data Manager web application
- Advanced end users who should maintain batches of master data using Excel with an MDS add-in

- Developers who should change existing applications to import master data from MDS, which becomes the authoritative source
- Business intelligence developers, who should change extract - transform - load (ETL) applications

## Developing a Training Plan

An important part of a self-service delivery framework is continuous learning. You should develop an initial training plan and an ongoing training plan. The training plan should include:

- An exact explanation for everyone on the project of why data quality and master data management is important and how it contributes to the overall company's success. With the introduction of solutions for these tasks, end users, data stewards, and many times even IT staff and SMEs get additional work, at least in the initial stage. They should all understand why they need to perform this work.
- In-depth training on data quality issues and tools like DQS and MDS should be provided for data stewards.
- Appropriate training for end users should be considered if you plan to implement MDS.
- IT professionals are many times in need of data profiling and automated data cleansing training. Of course, in such a case, they should be trained for their duties as well.
- Stakeholders might need some training to understand how to evaluate potential improvements in data quality.

## Inventorying Tools and Skillset

---

In Microsoft SQL Server suite, Data Quality Services and Master Data Services are the applications intended for data cleansing and master data management. In Office, Excel becomes an MDS and implicitly also a DQS client tool with the MDS add-in. This section introduces all these tools.

### Understanding Data Quality Services

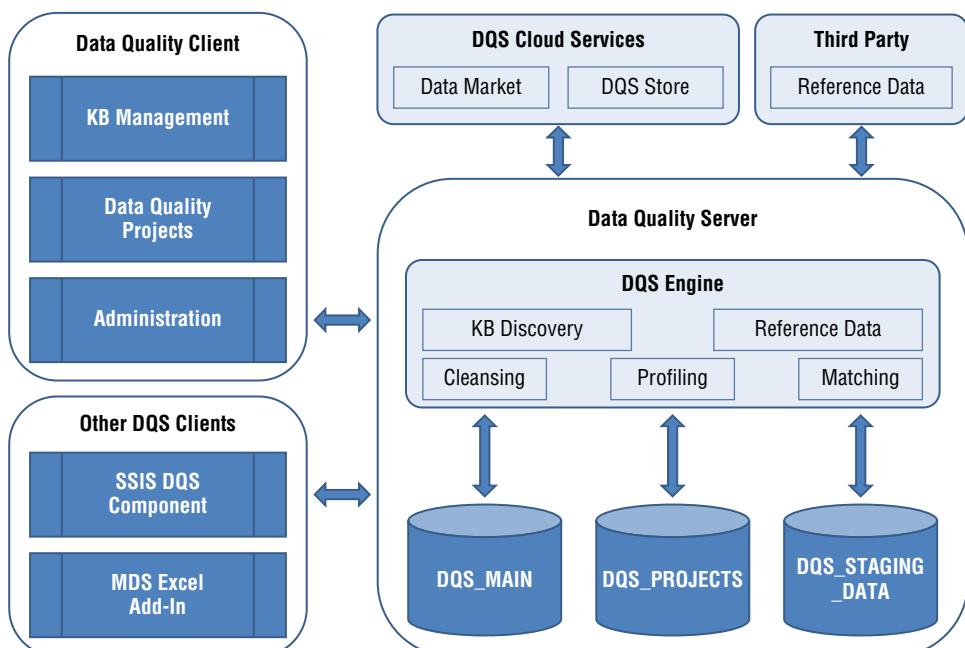
SQL Server Data Quality Services (DQS) is a knowledge-driven data quality solution. This means that it requires you to maintain one or more *knowledge bases (KBs)*. In a KB, you maintain all knowledge related to a specific portion of data—for example, customer data. In DQS projects, you perform cleansing,

profiling, and matching activities. You can also use an intermediate staging database to which you copy your source data and export DQS project results. DQS includes server and client components. Before you can use DQS, you must start by installing the DQS components.

In a knowledge base, you store all the knowledge related to a specific type of data source. Knowledge is contained in *data domains*. Domain knowledge includes:

- A semantic representation of a type of data
- A list of trusted values, invalid values, and erroneous data
- Synonym associations
- Term relationships
- Validation and business rules
- Matching policies

Figure 15-3 shows the DQS architecture.



**Figure 15-3:** The DQS Architecture

The Data Quality Server component includes three databases:

- **DQS\_MAIN:** Includes DQS stored procedures. The DQS stored procedures make up the actual DQS engine. In addition, DQS\_MAIN includes published knowledge bases. A published KB is a KB that has been prepared to be used in cleansing projects.

- **DQS\_PROJECTS:** Includes data for knowledge base management and data needed during cleansing and matching projects.
- **DQS\_STAGING\_DATA:** Provides an intermediate storage area where you can copy source data for cleansing and where you can export cleansing results.

You can prepare your own knowledge bases locally, including reference data. However, you can also use:

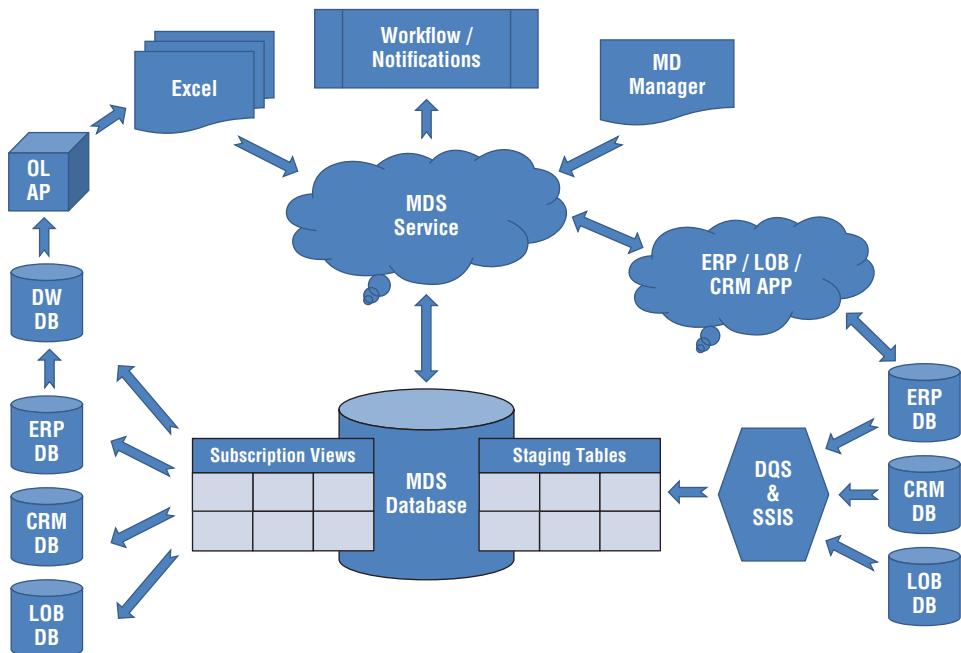
- Reference data from the cloud.
- Windows Azure Marketplace DataMarket to connect to reference data providers.
- A direct connection to a third-party reference data provider through a predefined interface.

With the Data Quality Client application, you can manage knowledge bases; execute cleansing, profiling, and matching projects; and administer DQS. SQL Server includes two new tools to assist with these tasks:

- **The SSIS DQS Cleansing transformation:** Performs cleansing inside a data flow of your SSIS package. This allows you to perform batch cleansing without the need for interactivity required by the Data Quality Client. For details about this transformation, please refer to MSDN at <https://msdn.microsoft.com/en-us/library/ee677619.aspx>.
- **The free Master Data Services (MDS) Add-in for Microsoft Excel:** With this, you can perform master data matching in an Excel worksheet. The DQS components must be installed together with MDS in order to enable DQS/MDS integration. Working with this add-in is covered more in depth in the “Managing Data Quality and Master Data in Excel” section of this chapter.

## Understanding Master Data Services

There are four main parts of the Master Data Services application. In this application, the master data is stored along with MDS system objects. MDS system objects include system tables and many programmatic objects such as system-stored procedures and functions. The MDS service performs the business logic and data access for the MDS solution. Master Data Manager is a web application for MDS users and administrators. In addition, advanced users can use the Master Data Services Add-in for Microsoft Excel. Figure 15-4 shows the MDS architecture at a glance.



**Figure 15-4:** The MDS architecture

You can insert master data into your MDS database manually through Master Data Manager or through Excel with the help of the MDS Add-in for Excel. In addition, you can import master data from existing databases through staging tables. The staging tables are a part of the MDS database. During the import process, you can cleanse your data with the help of Data Quality Services and integrate it from multiple sources with the help of SQL Server Integration Services. You can also export MDS data to other transactional databases and data warehouses. You can do the exporting through subscription views you create in the MDS database.

You can also integrate your applications with MDS. You perform the integration through the MDS web service. In addition, you can create master data workflows and notifications. MDS uses SQL Server Database Mail for notifications. For workflows, you can either create a custom solution through the web service or use Microsoft SharePoint workflows. You define workflows and notifications through business rules.

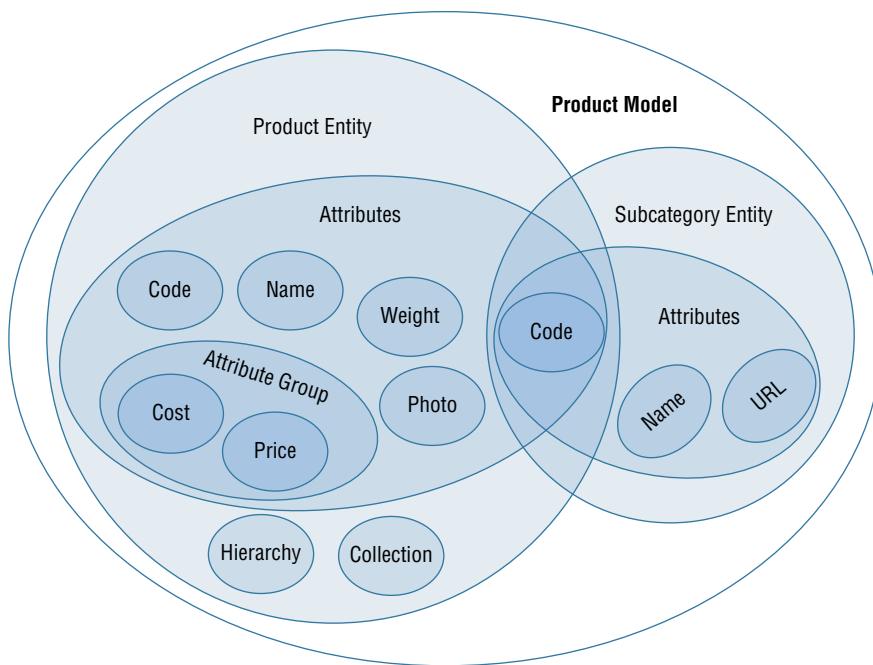
### ***Understanding Master Data Reusability***

The MDS architecture enables you to achieve one of the most important master data management goals—master data reusability—in many ways.

*MDS models* organize master data into logical groups for a specific business area. As mentioned earlier, master data is typically organized into four types: subjects, things, concepts, and places. A model contains the following objects:

- Entities
- Attributes and attribute groups
- Explicit and derived hierarchies
- Collections

MDS fully supports versioning out of the box. You can have multiple versions of a model at the same time. An MDS administrator can even revert transactions. Figure 15-5 introduces the MDS model, the objects in a model, and relationships between the model objects.



**Figure 15-5:** The MDS model and objects

### ***Understanding Entities***

A model contains entities. In Figure 15-5, the Product model contains two entities: Product and Subcategory. Entities contain attributes. For example, the Product entity contains the attributes Code, Name, Weight, and more. Some attributes are domain-based. This means that their value comes from a domain of possible values defined by another entity. In a classical relational model, this second

entity would be referred to as a *lookup table*. For example, one Code attribute in Figure 15-5 connects the Product and Subcategory entities. This code is the key for the subcategories.

### ***Understanding Attributes, Hierarchies, and Collections***

You might wonder why the name “Code” is used twice in Figure 15-5—once to identify products and once to identify subcategories. This is because in MDS, every entity must have two attributes with defined names: Code and Name.

Attributes can be further organized into attribute groups. Also, entities can have hierarchies. Some hierarchies are natural and stem from domain-based attributes. For example, a product belongs to a subcategory, which further belongs to a category. In addition to the natural hierarchies, you can also define explicit hierarchies in which you organize data according to your business needs. For example, you can organize countries, regions, and cities for your sales representatives in a way that optimizes travel expenses instead of using the natural hierarchy of country/region/city.

Furthermore, you can define arbitrary collections of entities. Use collections when you do not need a complete hierarchy and you want to view different groupings of members for reporting or analysis, or when you need to create a custom taxonomy. A collection is not a hierarchy; it is a flat list of members. However, a collection can contain other collections. This way, you can create custom taxonomies.

### ***Understanding Business Rules***

MDS business rules ensure data integrity. You can use business rules to find erroneous data, to automatically generate or update data, to send email messages, or to start a workflow. You express business rules by using If . . . Then statements: If an attribute value meets a specific condition, then MDS takes an action. You can specify a condition on a specific attribute value or whenever an attribute value changes. In order to use business rules, you have to publish them after you create them. You can apply business rules to a complete set of data or to a subset of data. You can also apply them to a specific version of data.

MDS can also send notifications for a simple workflow, or start an advanced SharePoint Server workflow. Business rules and workflows enforce the data quality of the master data over time.

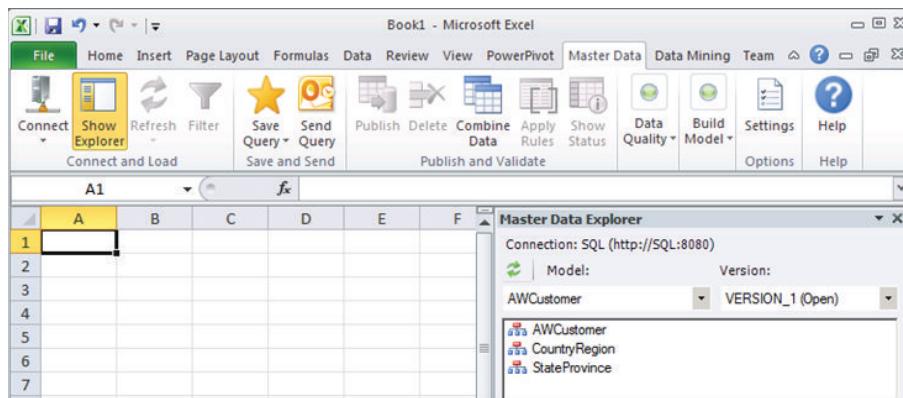
## **Managing Data Quality and Master Data in Excel**

The Master Data Services Add-in for Excel puts a lot of power into the hands of advanced users. Instead of updating MDS data row by row and entity by

entity, users familiar with Excel 2010 or 2013 can use it with all its capabilities to edit batches of data. When the edit is finished, they can publish the batch to the MDS database. MDS security is maintained inside Excel as well; users can load and publish only data for which they have been granted permissions. You can download and install the MDS add-in from the Master Data Manager home page.

Follow these steps to take advantage of this capability:

- 1. In Excel, establish a connection to your MDS service.** The connection string is the URL to your MDS instance. For example, if the instance is on your local computer and it listens on port 8080, you would use the URL <http://localhost:8080>. If you have the MDS Add-in for Excel installed, a new Master Data tab appears in Excel.
- 2. On this tab, click the Connect button in the Connect and Load group in the upper left corner.** There you can select a saved connection or create a new one through the Manage Connections dialog box.
- 3. Click the Show Explorer button to display the Model Data Explorer pane on the right side of the worksheet.** After you connect, you can select the model and version you want to edit. The list box below the model contains a list of entities. If you double-click an entity in this list, you automatically select all its members for loading into Excel. Figure 15-6 shows a worksheet in Excel with the MDS Add-in installed and the Master Data Explorer open.



**Figure 15-6:** The MDS Add-in for Excel

You probably don't want to load an entity with millions of members into a worksheet. Instead of double-clicking an entity in the Model Data Explorer, just select it there and click the Filter button in the Connect and Load group of the Master Data tab. Then you can select only the attributes you need and filter rows on values of attributes. While you are filtering your entity data, you can also reorder columns for display in Excel. After you have finished editing the

data, you can publish it to your MDS model. You do this by clicking the Publish button in the Publish and Validate group.

Each published change is a transaction. You can add annotations to each transaction. You can add an annotation to each row that has changed or to a batch of rows you are publishing. Before publishing, you can even combine data from two worksheets into one and compare it. Then you can further edit and correct the combined data before publishing. You can use the Combine Data button in the Publish and Validate group to do so.

One of the most powerful options in the MDS Add-in for Excel is the ability to de-duplicate data by using Data Quality Services. You can use this option if your Data Quality Services instance is installed on the same SQL Server instance as Master Data Services. You use the Match Data button in the Data Quality group for this task.

When you publish data, the published data is automatically validated against business rules. You can see the status of the validation when you click the Show Status button in the Publish and Validate group. In the same group, you can use the Apply Rules button to validate in advance, before publishing.

With the MDS Add-in for Excel, you can even edit the model itself. You can create an entity and change an attribute property. This is possible with the Create Entities and Attribute Properties buttons in the Build Model group of the Master Data tab. Of course, this option should be reserved for the most advanced users and data stewards only.

## Business Intelligence Features Across the Microsoft Data Platform Versions and Editions

Microsoft SQL Server 2014 includes some business intelligence features even in a free edition; for details, read the “Features Supported by the Editions of SQL Server 2014” article at <https://msdn.microsoft.com/en-us/library/cc645993.aspx>.

As a starting point for entering SQL Server business intelligence, you might consider using the Express with Advanced Services free edition of SQL Server 2014. It already includes SQL Server Reporting Services.

Analysis Services and Integration Services come with the Standard edition or higher. However, there are some limitations with the Standard edition.

For Data Quality Services and Master Data Services, you need at least the Business Intelligence edition.

Of course, the Enterprise edition offers all the features. You can try out the features by downloading and installing the Evaluation edition from the Microsoft TechNet Evaluation Center at <http://www.microsoft.com/en-us/evalcenter/evaluate-sql-server-2014>. And, if you are a developer, you might decide to use the Developer edition, which also gives you all the Enterprise features.

Finally, besides SQL Server 2014, you can also use SQL Server version 2012 for testing. SQL Server 2012 already brings the vast majority of the business intelligence features also available in the 2014 version.

## Defining Success Criteria

---

When implementing a self-service delivery framework, you should know the criteria for a successful project. Of course, when your users regularly use your business intelligence solution for improving their decisions, you can relax and say that your solution is a successful one. In addition, a better outcome for the company that uses this solution is also a good sign. However, defining how much improved data quality contributed to this success might be a more complex task. Improving the data quality requires huge investments in terms of time, money, and resources, but the results are somehow hidden in a broader business intelligence project.

Therefore, after you have implemented the data quality corrective and preventive solutions, you should measure how they perform. Even better, you should prepare the improvement infrastructure in advance, before starting to implement your solutions. By measuring the improvements, you can easily show the value of the solutions to key stakeholders. In addition, this approach allows you some measure of control over your work, in case your improvements fail and lead to even worse data quality problems. You can measure soft dimensions by conducting ongoing interviews with end users and domain experts. You can store the results of the interviews in a special data quality data warehouse to track the data quality over time. For hard dimensions, you can measure these automatically on a predefined schedule, and again store the results in the data quality data warehouse. Figure 15-7 shows a potential schema for a data quality data warehouse for measuring completeness and accuracy.

The schema in Figure 15-7 includes two central fact tables, one for tables and one for columns, with measures for the number of unknown values and for the number of errors in both of them. The surrounding tables, the dimensions, enable you to aggregate these numbers over column names, table names, table schemas, applications and databases, and of course, over time with regular calendar hierarchy year-quarter-month-day. Finally, you can aggregate the numbers of unknown values and errors over employees responsible for the specific tables and columns, or said differently, over data stewards. In this table, you have two different hierarchies—one regular (department-employee) and one parent-child (manager-employee) hierarchy. The foreign key on the Employees table that points to the Employees table itself denotes the parent-child hierarchy.

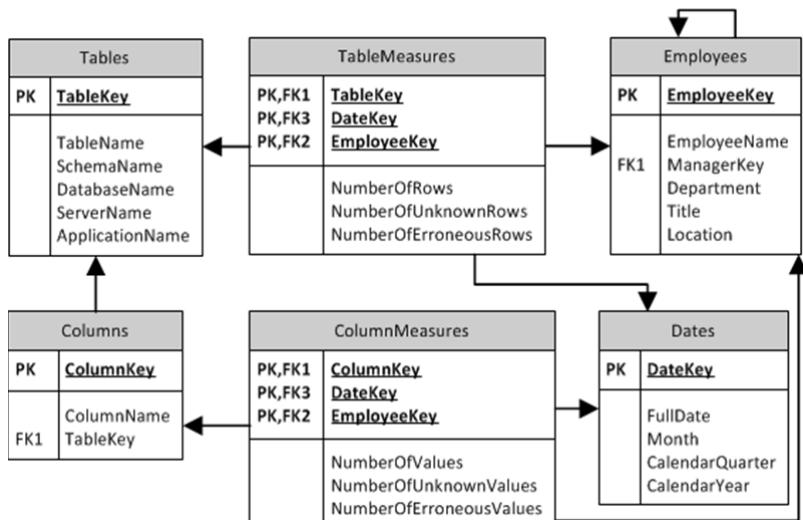


Figure 15-7: A potential schema for a data quality data warehouse

Don't forget, it is very important to communicate actions and results throughout the project. The more communication you have, the better. Domain experts can always help you with their knowledge. IT professionals and end users put much more effort into helping a project achieve success if they are involved, if they feel that the project is their project. Key stakeholders should always know how the project progresses and must be involved in all decision milestones.

## Summary

Implementing a self-service delivery framework requires a lot of effort dedicated to the data quality problems. You should create a detailed plan for the data governance and define explicit data stewards responsible and accountable for the master data. With data quality activities, you occasionally raise the quality of your data, but the quality might deteriorate over time. With master data management, you maintain the data quality over time. In the SQL Server suite, Data Quality Services and Master Data Services help you with data cleansing and master data management. In order to define a successful outcome of a project, you should measure the results and communicate them to all people and roles involved in the solution you are implementing.



# Designing and Implementing a Deployment Plan

*Application lifecycle management* (ALM) includes all phases of developing software, such as gathering requirements, writing code, testing software, managing deployments, and more. By following ALM best practices, business intelligence teams can increase their development productivity, reduce delivery times, and minimize maintenance disruptions. You can easily design and implement a deployment plan, an ALM best practices, using the Microsoft suite of business intelligence tools.

This chapter walks you through all the information you need to create a deployment plan for your business intelligence application. You will learn the definition of a deployment plan, why you must have one for your business intelligence solution, and some of the stumbling blocks you may face while implementing the deployment plan. You also learn how to deploy some of the business intelligence products, and finally, the steps needed to implement the deployment plan.

## What Is a Deployment Plan?

---

After you have gathered requirements from the users, designed your architecture, and developed and tested your code artifacts, you need to put your code somewhere for consumption. Moving your code and any associated assemblies,

artifacts, and configurations for someone to use is known as *code deployment*. Code deployment is an often overlooked, but important, part of the application lifecycle, and, as such, should be managed appropriately.

### CODE DEPLOYMENT

A *code deployment* is the act of moving code to a different environment.

Throughout the life of an application, you can deploy code multiple times, particularly to different server environments. Some organizations just have one environment, whereas others may have five or more environments, and anywhere in between. It is the authors' opinion that at least three environments are necessary to satisfy an appropriate division of responsibilities for each environment. Table 16-1 contains a list of each environment type, its purpose, and if it is required. The rest of this chapter focuses on having three server environments, but you can easily scale the number up or down to match your number of environments.

**Table 16-1:** Environment Server Options

| ENVIRONMENT          | DESCRIPTION                                                                                                                                             | REQUIRED? |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Development          | The development environment provides a place for developers to upload their code to ensure it works with all other systems and dependencies.            | Yes       |
| Test                 | The test environment provides a place for developers to run positive and negative tests on their code.                                                  | No        |
| User Acceptance Test | The user acceptance test (UAT) environment provides a place for end users to test the application.                                                      | Yes       |
| Staging              | The staging environment provides a place for production control and testers to ensure the production release will succeed before pushing to production. | No        |
| Production           | The production environment contains the final set of code where end users can use the application.                                                      | Yes       |

Moving code through each environment should be a repeatable process, where each step is documented and defined. The people in charge of each step can repeat the same actions each time. Ideally, they have as little manual intervention to the process as possible, which reduces the risk of human error. The documentation for this process is called a *deployment plan*.

**DEPLOYMENT PLAN**

A *deployment plan* is the document that contains the steps your organization uses to move code through its server environments. The deployment plan should be repeatable, non-intrusive, and deterministic.

You should document the deployment plan for use by anybody in the organization; whether it be the production control team, the DBA (database administrator), or the development staff. Especially in the case of staff turnover, documenting the plan and approach ensures that new staff can quickly and smoothly pick up the reins and continue deployments. Additionally, you should decide and document who performs each step of the deployment and if any step depends on another step and/or person. If so, the deployment plan should include a communication aspect to say who informs whom of the “baton handoff” to ensure the deployment completes.

Your organization will see many benefits if you use deployment plans for your releases. Not only will your deployments go more smoothly, but the plans can help answer questions in case they come up about the process. Of course, if your organization does not buy into the process, using a deployment plan may not be the best option for you. Keep in mind that creating the deployment plan and following it through can take time. But if you can make the time to create and follow the plan, you will reap the rewards.

## How Do You Deploy Business Intelligence Code?

There are many available deployment approaches, not even specific to business intelligence applications. Depending on the number of your organization’s servers, the availability of your staff, and the amount of time you want to initially set up and continuously run deployments, you may choose a different approach. This section discusses some of the approaches that you can take to deploy business intelligence code.

At a high level, you can deploy your code manually or continuously. A manual deployment uses a pull model, where you decide when to deploy your code, gather the necessary resources, and perform the deployment. A continuous deployment uses the push model where a separate application, such as a Team Foundation Server (TFS) server, gathers the necessary resources and deploys the code on a regular basis.

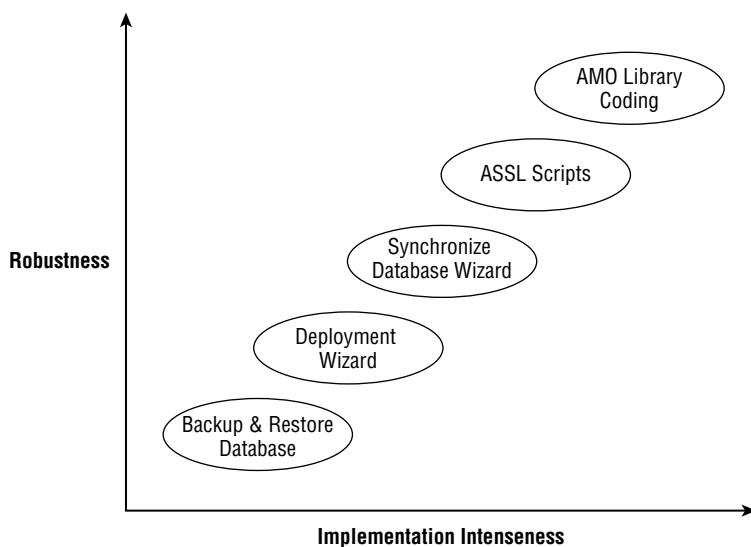
At a lower level, you can deploy your code using a variety of methods: through scripts, through user interfaces, or manually. Scripts tend to fall under the continuous deployment methodology, whereas user interfaces are more manual. You can absolutely use a combination of these methods if that makes sense for

your organization; you want to make the process as easy and painless as possible. The next sections discuss two of the business intelligence tools and how to perform deployments using those tools.

## Using Analysis Services (Multidimensional or Tabular)

You have several options to deploy the code associated with your Analysis Services databases. For additional information on the Analysis Services models and the type of code that needs to be deployed, see Chapters 7 and 8. Luckily, the deployment methods are similar whether you are using a multidimensional or tabular model.

The deployment options vary from easy deployments and less robustness to harder deployments and more robustness, as shown in Figure 16-1. Additionally, you should consider the types of code and layout that you have in place when picking an option. Based on those factors, each organization could pick a different option along the spectrum. This section explains each of the deployment options and provides examples of how to perform the deployment.



**Figure 16-1:** Analysis Services deployment methods

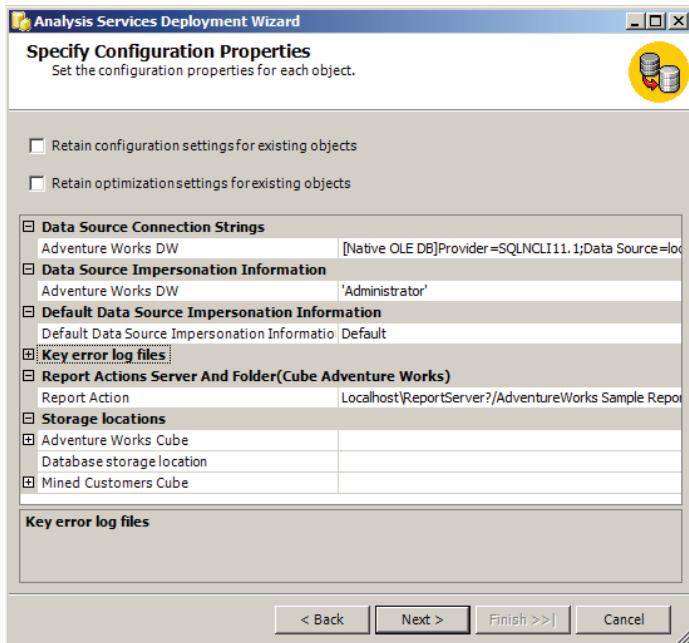
### *Performing a Manual Backup*

The simplest way to deploy an Analysis Services database is to perform a manual backup of an existing database and restore it to the new environment. You can

back up the database in SQL Server Management Studio (SSMS) by right-clicking on the database and selecting the Back Up option. Then you can restore the same file on the new server by similarly using the Restore option either with or without the security information. This option moves the data as well as the code to the new environment, so it does not work if your database has enabled any write-back functionality or you use automated processes to push data into your database. Finally, after you've done the restore, don't forget to change the data sources to point to the appropriate databases.

### **Using the Deployment Wizard**

You can also easily use the Deployment Wizard to deploy an Analysis Services database. You can find the Deployment Wizard under the Start menu, Microsoft SQL Server 2014 folder, and Analysis Services folder. This method requires you to have an Analysis Services project file, so it is a great candidate for continuous deployments. You have additional flexibility of your deployment with this option, allowing you to retain existing partitions, to replace roles and members if desired, and to specify the new data source during the deployment. You can see some of the properties you can change using the Deployment Wizard in Figure 16-2.



**Figure 16-2:** Analysis Services Deployment Wizard

### ***Using the Synchronize Database Wizard***

A more complex method is to use the Synchronize Database Wizard to compare two databases and copy the differences to a new environment. You can access the wizard in SSMS by right-clicking on the Databases folder and selecting the Synchronize option. A wizard opens where you can enter the source database for synchronization; the destination database is the server where you opened the wizard. The wizard allows you to choose your security information and either run the synchronize process now or script it for later.

### ***Using ASSL***

The next-most-complex method is to use Analysis Services Scripting Language (ASSL), part of Expression Markup Language for Analysis (XMLA), scripts to publish the objects to a new environment. This method only pushes the objects, but can also be used to kick off processing to populate the data if needed. Three sample ASSL scripts follow, one to back up the original database, one to restore the database, and one to synchronize the databases. These scripts are the programmatic method of the previous methods. By using a programmatic method, you can change the script before running it, and you have less chance for error during deployment.

```
<Backup xmlns="http://schemas.microsoft.com/analysisservices/2003/
engine">
 <Object>
 <DatabaseID>AdventureWorksDWMultidimensional</DatabaseID>
 </Object>
 <File>AdventureWorksDWMultidimensional.abf</File>
</Backup>

<Restore xmlns="http://schemas.microsoft.com/analysisservices/2003/
engine">
 <File>C:\Program Files\Microsoft SQL Server\MSAS12.MSSQLSERVER\OLAP\
 Backup\AdventureWorksDWMultidimensional.abf</File>
 <DatabaseName>AdventureWorksDWMultidimensional</DatabaseName>
</Restore>

<Synchronize xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
 <Source>
 <ConnectionString>Provider=MSOLAP.6;Data Source=DEVSERVER;
 Integrated Security=SSPI;Initial Catalog=AdventureWorks
 </ConnectionString>
 <Object>
 <DatabaseID> AdventureWorksDWMultidimensional </DatabaseID>
 </Object>
```

```

</Source>
<SynchronizeSecurity>CopyAll</SynchronizeSecurity>
<ApplyCompression>true</ApplyCompression>
</Synchronize>

```

### ***Writing Code with the AMO Library***

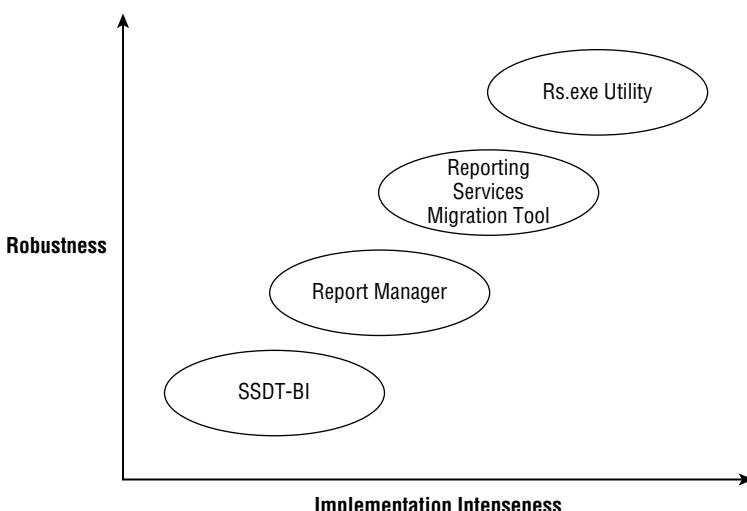
Finally, you can write code with the Analysis Management Objects (AMO) library to deploy an Analysis Services database. This typically involves writing a program that someone can run on demand after entering some parameters. If you have many changes to the Analysis Services deployment that can be automated, this may be the way to go. Some examples of this include adding new partitions, or dynamically changing data sources based on disaster recovery or failover scenarios.

For more information on the AMO object model and how it can be used to deploy objects, see: [https://technet.microsoft.com/en-us/library/bb522603\(v=sql.110\).aspx](https://technet.microsoft.com/en-us/library/bb522603(v=sql.110).aspx).

## **Using Reporting Services**

You must also deploy Reporting Services reports and data sources as part of the deployment plan. For an overview of Reporting Services and the benefits that it provides, see Chapter 11. Keep in mind that you can install Reporting Services inside of SharePoint or on its own, which will affect how you deploy code.

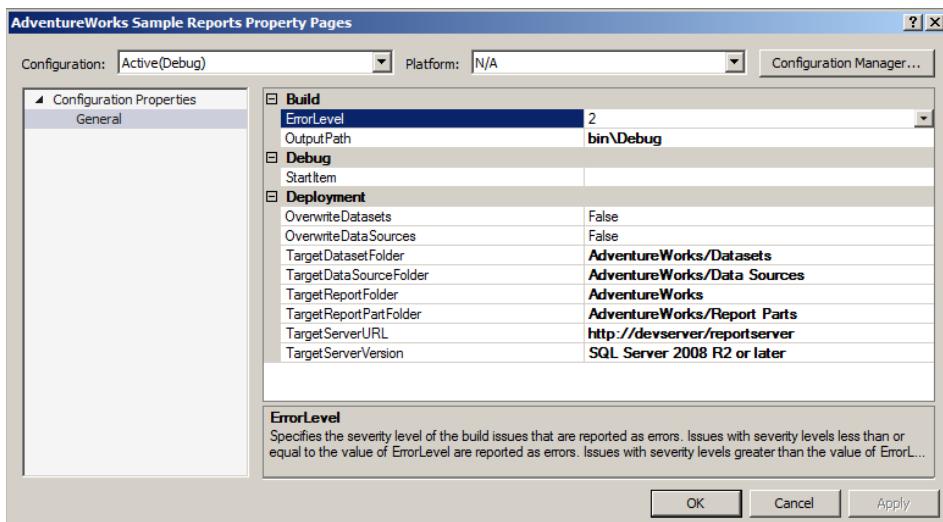
Reporting Services comes with several deployment methods, and similar to Analysis Services, these methods range in ease of implementation and riskiness. These methods are shown in Figure 16-3.



**Figure 16-3:** Reporting Services deployment methods

### **Using SSDT-BI**

One of the easiest methods available to deploy Reporting Services reports is to use the development environment, either SQL Server Data Tools - Business Intelligence (SSDT-BI) or Report Builder. Within the report project, you can configure the report server deployment properties, as shown in Figure 16-4. By switching the Configuration drop-down list at the top of the window, you can set different properties for each environment. Once the properties are set, you can simply right-click the project name or an individual report, select the Deploy option, and you are done!



**Figure 16-4:** Reporting Services Property Page

### **Using Report Manager or SharePoint Interface**

The next-simplest method is to use the Report Manager or SharePoint interface. This method requires you to go to the site and select the reports that you want to deploy. You must know the location of the reports on your local machine and select the correct ones to be deployed. You deploy each report individually, so this can be time consuming if you have many reports.

### **Using Reporting Services Migration Tools**

Your environment may also allow you to use the Reporting Services Migration Tool. Microsoft provides this add-on at the following location: <http://www.microsoft.com/en-us/download/details.aspx?id=29560>. You can run the application through a command line or the provided user interface. Unfortunately, the version as of SQL Server 2014 only supports a migration from a native

mode server to a SharePoint integrated mode server, so it may not work for all organizations.

### ***Using Rs.exe Utility***

The final method to deploy your Reporting Services code is to use the rs.exe utility. Provided with the Reporting Services install, this utility allows you to input a script that lists out the source server, destination server, and the items and properties for deployment. Microsoft even provides a sample script, which you can use and modify, found here: <https://msdn.microsoft.com/en-us/library/dn531017.aspx>.

Don't forget to set any additional properties, such as subscriptions or caching. You can set these manually or through any of the scripting methods described previously.

## **How Do You Implement the Deployment Plan?**

---

Now that you understand the purpose and the benefits of the deployment plan and the different deployment options, you can put it all together! Implementing the deployment plan takes several steps: planning, creating the scripts, documenting the steps, testing the plan, and finally, training your staff to carry out the plan. This section takes you through each of these steps.

### **Planning the Deployment**

To start creating the deployment plan, you must pick the desired type of deployment. Whether continuous or manual, scripted or GUI-based, you want to pick what will work best for your organization.

Consider the following tenets while planning your deployment:

1. Identify separate instructions for deployment of server features versus code.
2. Create the same plan no matter whether this is an initial deployment or an incremental deployment.
3. Do not modify any code. Instead, use external configurations to change environment-specific variables.
4. Remember how many environments and servers per environment must be managed during the deployment.
5. Keep your deployments as simple as possible, which may even mean splitting one deployment into multiple deployments.

Keeping these tenets in mind ensures you have a robust and scalable plan.

## Designing Scripts

The next step in creating your deployment plan is to design the scripts needed to deploy the code. Based on the previous descriptions, the scripts can be simple or more complex. There are several items to keep in mind as you create the scripts:

- Create a rollback script in case you need to revert the deployment.** This rollback should put the environment back to the same state prior to the deployment for both data and code. If you have multiple deployment scripts, a rollback could be performed for an individual script or multiple scripts.
- Add functionality to the script to “clean itself up” as it goes.** This includes removing temporary tables or objects that were created during the deployment.
- Check for existence of any new objects or tables to ensure the script does not fail.** For example, if you are creating a new partition in Analysis Services, only create it if it does not already exist.
- Document, document, document.** Remember that these scripts are not only for you, they are also for the person that takes over your job when you get promoted.

After you have created your scripts, you can put them together in a document with steps.

## Documenting Steps

The deployment scripts are only as useful as the order in which you run them. A deployment plan also contains the steps necessary to perform the deployment. Including the steps provides multiple benefits: The scripts will be performed in the same order each time, different people can perform each step if needed, and everyone will know the expected outcome. Figure 16-5 shows part of a deployment plan, which highlights an example format for your documentation of steps.

Deployment Plan				
Description: Add Customer Subject Area to Data Warehouse			Date: January 20, 2015	
Step	Assigned To	Task	Expected Result	Failure Steps
1	Jen Smith, Architect	Email users that the deployment is starting and to stop using the reports.	N/A	N/A
2	Robert Jones, DBA	Add new users to the database by running script, 20150120-A-AddUsers.sql, in Management Studio.	The script prints out: 22 users were added. Script complete.	If the script fails, a login was probably mistyped. Fix login and rerun. Otherwise, manually add the user. If still not added, halt deployment.
3	Robert Jones, DBA	Add new tables to the database by running script, 20150120-B-AddTables.sql, in Management Studio.	The script prints out: DimCustomer was created. DimCustCategory was created. Script complete.	If the script fails, check that you are running under your administrator account. If still not added, halt deployment.
4	Samantha Doe, Production Control	Move reports from previous environment (development to test to production) using the rs.exe script, 20150120-C-MoveCustomerReports.rss.	The executable completes without error.	If the utility fails, check that the report names are spelled properly in the script file. If so, try manually uploading reports through Report Manager. If still fails, halt deployment.

**Figure 16-5:** Deployment Plan format

Your deployment plan could include, but is not limited to, the following types of steps:

1. Communication to business stakeholders that the deployment has started and completed.
2. Installation of any dependencies, either third-party components or internal assemblies.
3. Script information, including who should run it, what the script name is, where to run the script, and its purpose.
4. Validation that the deployment went smoothly, such as executing the report or seeing data in a table.

Include as much information as possible and be as descriptive as possible when filling out the steps. Your future-self will thank you.

## Testing the Plan

Testing your deployment plan is just as important as testing your code. A failed deployment can result in lack of confidence, extended downtime, and even loss of data. A great recommendation is to use each environment as a test of your deployment plan and scripts. In other words, don't wait until after the code is in your test environment to create your deployment scripts and steps. Create the plan before you deploy any code, and follow the plan step by step as you move the code through each environment, adapting it and refining it at each deployment to ensure solidity.

Additionally, you want to test each of the characteristics of the plan in a safe environment. Allot some testing time for the deployment plan in your project plan to test these characteristics.

For example, test the rollback code in the test environment. Ensure that no errors exist, that all code was successfully removed or restored to its previous version, and that no data was changed in the process. Then run the same deployment script to ensure that the deployment happens as expected.

Another test is to make sure that the deployment plan will run again even if it stops due to an error. Run the test all the way through, and then run it again. Even though the first run didn't error, by running it a second time, you ensure that every step can be rerun without failure.

Run more tests as needed; you know your plan the best. The important thing is to try out multiple scenarios and correct any errors that you find. After all of your testing, you can feel confident that the deployment into production will go quickly and smoothly!

## Training Your Staff

Finally, you must train the people who will be performing the steps in the deployment plan. These people could include developers, database administrators, or production control staff. Although it is often the architect or lead developer who creates the development plan, having these other groups of people involved and bought into the process is just as important.

Training can be as simple or as in-depth of a process as your organization needs. You can create formal training classes where you walk through each step in the plan and have each person perform his or her task, or you can set up a show-and-tell of the deployment plan and open the floor for questions. The goal here is to make sure all staff members know their role in the deployment plan and that there are no questions during the actual deployment. Once you have completed your training, you are ready to perform your deployment!

## Summary

---

It is important to have a plan in place to deploy your business intelligence code. Now that you understand the definition of a deployment plan and its usefulness to your organization, you are ready to take the next step and create your own plan. Make the deployment as repeatable as possible, and you will be happy with the result. The next chapter discusses how to manage and maintain your business intelligence environment.

# Managing and Maintaining the Business Intelligence Environment

One of the key success factors of a business intelligence project is the end-user experience. This experience—unfortunately for those individuals involved in the project—is solely based on how the solution performs: simply put, how fast the solution provides answers, renders reports, delivers data, and so on. If any aspect of this is not optimal, the outcome could be detrimental to the entire project. For example, if the CEO opens a dashboard and it takes more than a couple of seconds, how likely will he or she revisit that dashboard? Although this is an overly simplistic example, in actuality something that simple can easily mar the perception of months or even years of work. As a result, any solution should include a performance monitoring strategy that keeps a proactive eye on all aspects of the deployment.

Remember that this book's topics focus on the aspects of business intelligence that assume an Extract, Transform, and Load (ETL) process and data warehouse are already implemented. Therefore, this chapter does not focus on monitoring performance as it relates to those topics, but instead on tools and features that have already been discussed.

## Using SQL Server Reporting Services

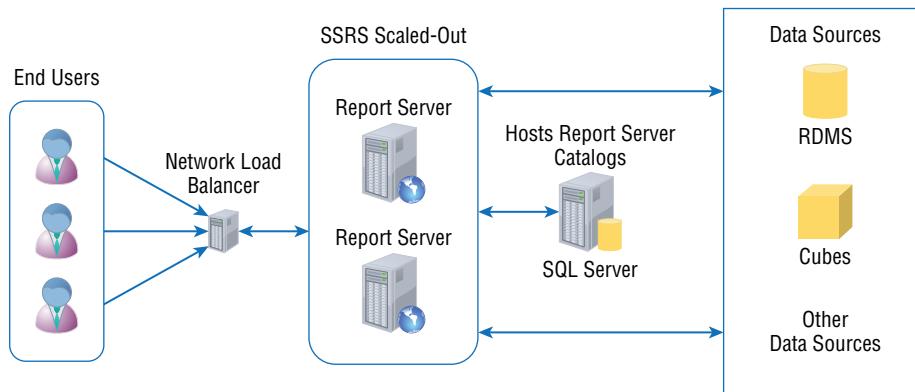
---

The methods, techniques, and tools used to monitor Reporting Services in native or SharePoint integrated mode are mostly the same. However, SharePoint does include diagnostic logging events that can be used in addition to what SSRS

itself provides. A SharePoint administrator may be required to configure the diagnostic logging events. Regardless, alone or together, it is possible to mitigate or identify SSRS performance issues.

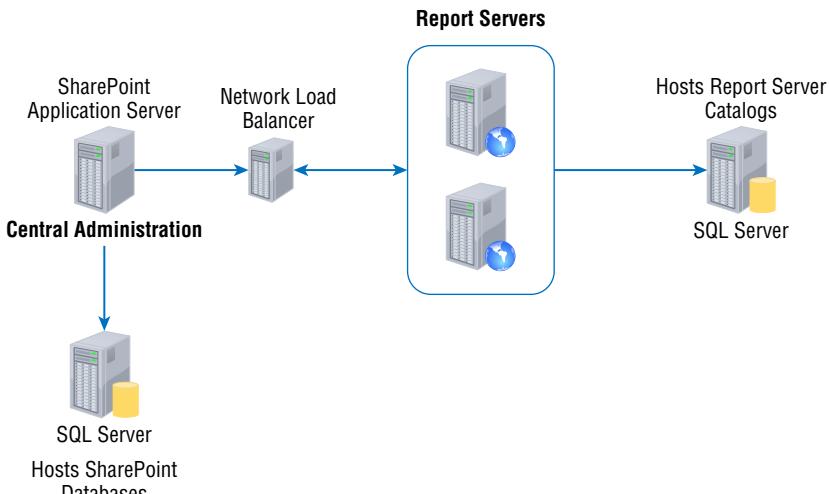
With any technology, the best approach is to take a proactive approach in regards to performance and scale. Instead of waiting for problems to occur, plan accordingly during the design and implementation process. Selecting the correct hardware, software, and topology is critical. Because SQL Server is natively built for 64-bit hardware, your deployment should not only include 64-bit hardware but also 64-bit software. The first reason is because the databases that every SSRS deployment depends on are hosted on SQL Server. As a result, the same approach used for optimizing any SQL Server database should be considered here. In most cases, leveraging 64 bit is always the better choice.

The second reason is because of a change that was made to SSRS memory dependency. When SQL Server 2008 was released, Microsoft decided to remove the heavy memory dependency from SSRS. In other words, unlike other versions, it and all of its predecessors are not memory bound. This means that if SSRS is under memory pressure, it can leverage the file system as a backup to ensure that reports still render. Now, is this optimal? Absolutely not. If SSRS has to use the file system during report processing, end users will definitely experience degraded performance. This is especially important to consider when you have a large number of concurrent users hitting the report server, or if you have several large reports running at the same time. The reports will render in either case, but they will render slowly. Scaling SSRS is one approach that could solve this problem. Figure 17-1 displays a scaled-out native SSRS deployment.



**Figure 17-1:** Scaled-out SSRS deployment with a Network Load Balancer (NLB)

You can perform a similar deployment with SSRS in SharePoint integrated mode, shown in Figure 17-2.



**Figure 17-2:** Scaled-out SharePoint SSRS integrated deployment

With either deployment, multiple Report Servers act as a single unit responding to user requests. The NLB in both topologies helps to control traffic and ensures that requests are equally distributed across the Report Servers. You must use separate load-balancing hardware or software because SSRS does not natively support load balancing.

## Configuring Memory

After all the hardware and software is properly deployed, the next step is to properly configure memory thresholds. By default, SSRS is configured to consume all the available memory on a server. This is specified when the service is started. The first step in configuring the memory is to locate the RSReportServer.config file. The location is based on whether or not you are running SSRS native or integrated. Table 17-1 provides details for each.

**Table 17-1:** SSRS Config File Locations

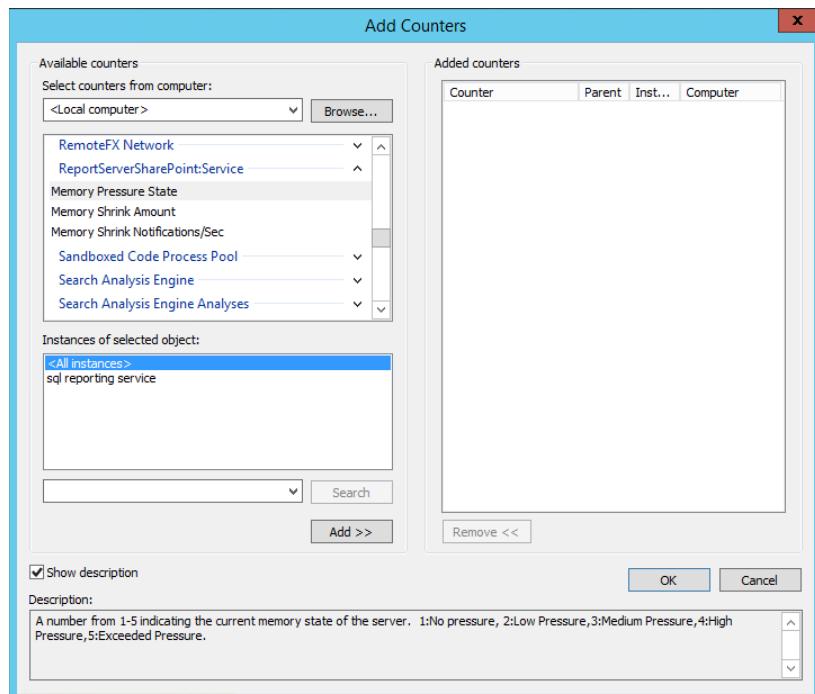
SSRS MODE	CONFIGURATION FILE LOCATION
Native	C:\Program Files\Microsoft SQL Server\MSRS12.MSSQLSERVER\Reporting Services\ReportServer
Integrated	C:\Program Files\Common Files\microsoft shared\Web Server Extensions\15\WebServices\Reporting

When SSRS is under memory pressure, it can dynamically adjust memory allocations based on machine resource constraints. As memory changes—specifically, increases—SSRS reports memory pressure as described in Table 17-2:

**Table 17-2:** Memory Pressure Definitions

MEMORY PRESSURE	SYSTEM RESPONSE
Low	All requests continue as normal.
Medium	All requests continue as normal. New requests may or may not be accepted.
High	SSRS requests begin to leverage the file system instead of solely utilizing memory. New requests are denied. Memory allocations are reduced.

You can actually monitor memory pressure using Performance Monitor, which is a built-in Windows Server tool often used for monitoring server performance. An entire book could be written on using Performance Monitor with SSRS. However, for the sake of brevity, refer to the link <https://technet.microsoft.com/en-us/library/cc749115.aspx> for details on configuring collection sets to monitor performance. Figure 17-3 shows the Add Counters window used by Performance Monitor.

**Figure 17-3:** Adding counters to Performance Monitor

This particular image displays adding a counter from an SSRS in SharePoint integrated mode. The object used is ReportServerSharePoint:Server and the

counter is Memory Pressure State. When you click Add, then OK, this object appears on a graph depicting the value visually. If the value changes, the graph would also change. One thing to mention before moving on is that, if this was a native install, then the Performance object would be ReportServer:Service and the counter would be the same. Regardless of the mode, the values are reported the same: 1 (No Pressure), 2 (Low Pressure), 3 (Medium Pressure), 4 (High Pressure), and 5 (Exceeded Pressure).

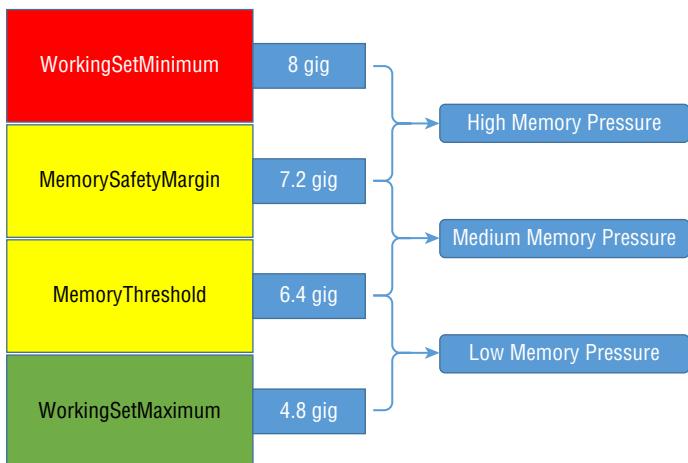
Although the Memory Pressure counter is not configurable, by using the rsReportServer configuration file, you can make changes to define when the responses are sent. In most cases, if the servers are sized and scaled properly, you have no need to change the settings. However, if periods of slowness occur, which may equate to memory pressure, then adjustments may be required. There are four configuration settings: WorkingSetMaximum, WorkingSetMinimum, MemorySafetyMargin, and MemoryThreshold. Table 17-3 provides a description of each.

**Table 17-3:** SSRS Configuration Values

CONFIGURATION SETTING	DESCRIPTION
WorkingSetMaximum	Maximum amount of memory SSRS will consume. By default, equal to amount of memory available on system. Also, unavailable in rsReportServer.config unless specified. Determined at SSRS start time. Expressed as kilobytes. Does not accept new requests if SSRS reaches this value.
WorkingSetMinimum	Minimum amount of memory SSRS will consume. By default equal to 60 percent of WorkingSetMaximum. Also, unavailable in rsReportServer.config unless specified. Determined at SSRS start time. Expressed as kilobytes.
MemorySafetyMargin	Percentage of WorkingSetMaximum, which determines range between medium and high memory pressure. Processing slows if SSRS memory usage reaches this value. Default value is 90.
MemoryThreshold	Percentage of WorkingSetMaximum, which determines range between low and medium memory pressure. Default value is 80.

To illustrate a sample configuration, assume a server has 8 gigabytes of RAM. Using the default settings, shown in the following block of code, memory pressure range distribution would resemble Figure 17-4.

```
<WorkingSetMinimum>4800000</WorkingSetMinimum>
<MemorySafetyMargin>80</MemorySafetyMargin>
<MemoryThreshold>90</MemoryThreshold>
<WorkingSetMaximum>8000000</WorkingSetMaximum>
```



**Figure 17-4:** A sample of what the settings would look like in `rsReportServer.config`

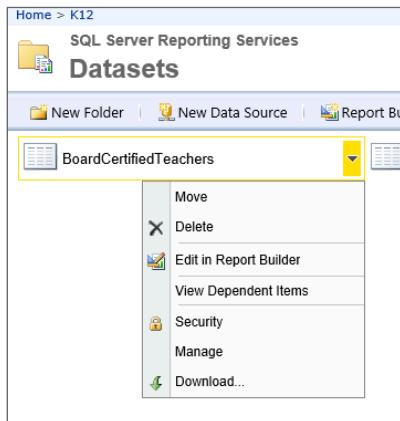
Remember, only adjust these when there is suspected memory pressure. Otherwise the default setting should be sufficient for most environments.

## Caching Data and Pre-Rendering Reports

Another option to consider are several features built into SSRS, both native and integrated modes. There are three great options: Schedule Delivery, Cache Executions, and Snapshots. These methods can reduce load on the Report Server by minimizing memory requirements. For example, caching the data up front minimizes live query executions. Or if you use a snapshot, you can pre-render a large report during nonworking hours, and, as a result, minimize contention between live users rendering reports and the rendering of larger reports.

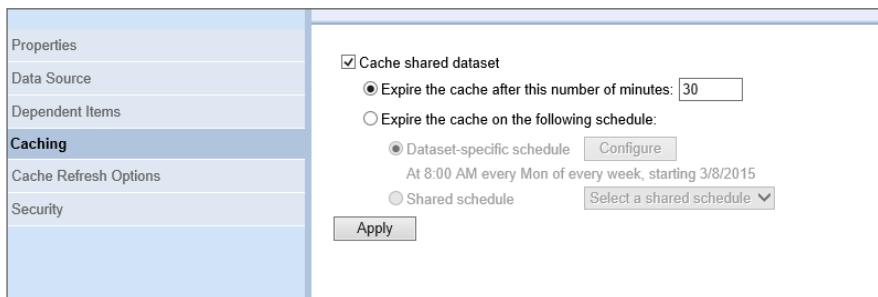
The key thing to remember is that by performing the work up front, you can use either option to assist in minimizing memory pressure. When end users access the large or resource-intensive reports, instead of hogging server resources, they can access the report or report data easily using little or no resources. Configuring these capabilities is similar between both modes. The only difference is how you access them. In native mode, the method of access is via the Report Manager, whereas in integrated mode it's done via an SSRS document library. Follow these steps:

- 1. Identify your item.** Once in the Report Manager, you identify a report or shared dataset and hover over it. A drop-down arrow appears.
- 2. Access the Dataset Manage page.** If you click the drop-down arrow, a menu opens, which you can see in Figure 17-5. Select Manage from the list of available choices and the Dataset Manage page opens. A list of menu choices are available on the left of this page.



**Figure 17-5:** SSRS Shared Dataset option

3. **Set your caching options.** Select Caching from the list of available choices as shown in Figure 17-6. Select the Cache shared dataset check box. From there, the person configuring the caching can provide the number of minutes to make the cache available or can use a schedule. Either method will accomplish the goal.



**Figure 17-6:** SSRS Shared Dataset Caching page

The first person that executes the report has to wait while the query completes, but all subsequent users can access the cached dataset. This is until the cache expires. The full details about caching, snapshots, and schedules are beyond the scope of this book. However, they are options that should be considered when performance issues arise, but ultimately the best approach in this case would be to start with tuning the source query for the report.

## Using ExecutionLog Views

The ExecutionLog views included in the Report Server database catalogs provide information that can also assist with performance monitoring and tuning. Although there are three ExecutionLog views, you always use the latest copy in

numerical order. In the case of SQL Server 2014, ExecutionLog3 is the preferred view. So far, most of this chapter's focus has been on memory, but there are definitely other causes to poor-performing reports, one being a slow query or stored procedure. Executing the following query can quickly help to identify reports that are slow due to slow queries:

```
SELECT *
FROM dbo.ExecutionLog3
ORDER BY TimeDataRetrieval DESC
```

The ORDER BY DESC in the query quickly brings to the top the reports that spend large amounts of time retrieving data. One important thing to remember is that both SSRS and Power View report executions are contained within this table. Therefore, this is a quick and easy ad-hoc approach. Alternatively, you can create and deploy an SSRS report to provide data in a more formatted view to assist in diagnosing and solving performance issues.

Another column, AdditionalInfo, also contains a wealth of information. One particular element, EstimatedMemoryUsageSK, is just what the name implies: It provides an estimate of the amount of memory that each part of a report utilizes during rendering. The following code snippet displays a sample from a report execution.

```
<AdditionalInfo>
<ProcessingEngine>2</ProcessingEngine>
<ScalabilityTime>
 <Pagination>0</Pagination>
 <Processing>0</Processing>
</ScalabilityTime>
<EstimatedMemoryUsageKB>
 <Pagination>4</Pagination>
 <Processing>86</Processing>
</EstimatedMemoryUsageKB>
<DataExtension>
 <SQL>2</SQL>
</DataExtension>
<Connections />
</AdditionalInfo>
```

If the numbers are high, this could be an indicator that a corresponding report is causing memory pressure. You can use many other columns in the table and elements within the XML to diagnose performance issues. You should definitely spend time reviewing the results of these views often. The results are a good place to start the diagnosis and can provide direction, preventing you from randomly checking other areas.

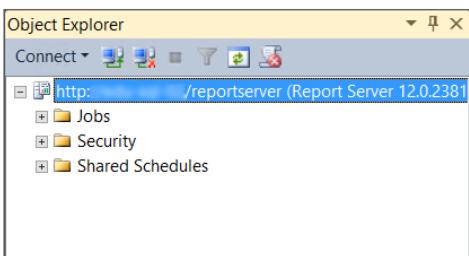
In addition to what is logged by default, both native and integrated modes of SSRS have the ability to capture more data by enabling verbose logging. An explanation of how to enable verbose logging and performance-tuning techniques is discussed in the “Using SharePoint to Improve Performance” section. However, to configure verbose logging in native mode, follow these steps:

1. **Connect to the server.** Open SSMS and connect to a Report Server (see Figure 17-7).



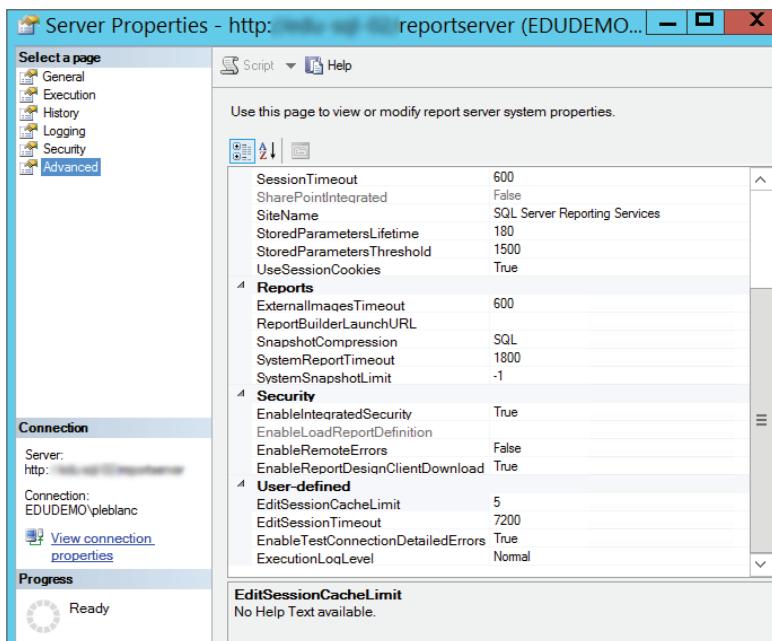
**Figure 17-7:** Connect to Server dialog box

2. **Set connection options.** Select Reporting Services from the Server type drop-down list. Then enter the report server URL, `http://<server name>/reportserver`, and click Connect. Once a connection is established, the Object Explorer opens, which is shown in Figure 17-8.



**Figure 17-8:** SSMS Object Explorer connection to an SSRS Server

3. **Change the appropriate properties.** Right-click the server name and select Properties from the context menu. In the left navigation, select Advanced from the list of available choices. Scroll down to the bottom of the list of properties and change the ExecutionLogLevel from normal to verbose, as shown in Figure 17-9.



**Figure 17-9:** Server Properties window

**WARNING** Once enabled, the XML displayed in the Additional Info column contains more robust data as it relates to a report execution. You should use this setting with caution because it can require additional resources. Therefore, when all diagnosis is complete, make sure that you reset the value back to normal.

## Working with SQL Server Analysis Services

Before this discussion begins, understand that complete books and white papers have been written on diagnosing and tuning SSAS performance problems. As a result, I'll provide only a brief overview or synopsis of each; this section primarily outlines and discusses the key starting points.

### Using Multidimensional Models

In Chapter 8, the section “How Do You Enhance the Model?” provides three of the key methods for tuning a multidimensional model. In addition to these techniques, warming the cache is another viable option. You do this by preloading data into the SSAS caches, storage engine, and query processor. To accomplish

this, simply execute several MDX queries using the CREATE CUBE or WITH CACHE statement. For example, you might follow these steps:

1. **Clear the cache.** You can do so using the following statement:

```
<ClearCache xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
 <Object>
 <DatabaseID>Adventure Works DW</DatabaseID>
 </Object>
</ClearCache>
```

**WARNING** It's highly recommended that you change the name of the database in the DatabaseID element to match the database where the cache is being cleared. This is required to ensure that the query is not already cached and is strictly for testing purposes. Please do not do this on product systems.

2. **Add events and filters.** Launch the profiler connecting to that server and add the following events and filters on the DatabaseName:

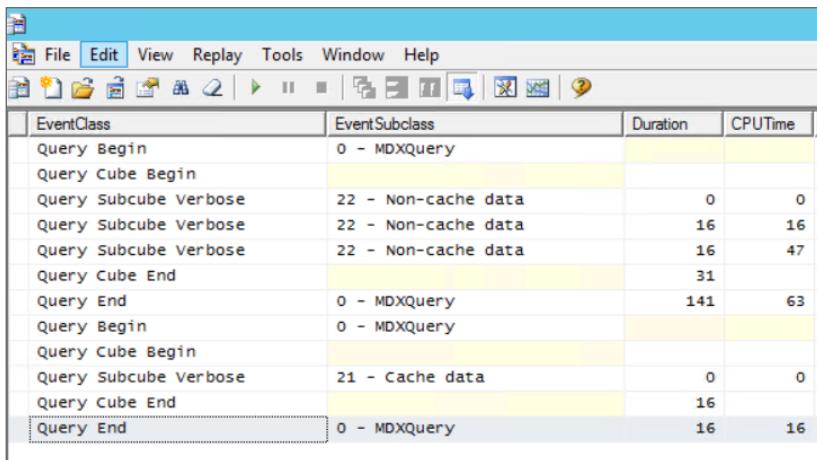
- Query Begin
- Query Cube Begin
- Query Subcube verbose
- Query Cube End
- Query End

3. **Ensure the profiler is running.** Execute the following query twice, ensuring that the profiler is running:

```
SELECT
 ON COLUMNS,
 NON EMPTY { ([Geography].[Country].[Country].ALLMEMBERS) }
 ON ROWS
 FROM [Adventure Works]
```

4. **You can view a sample trace in Figure 17-10.**

5. **Review each section of the trace.** Make note of the Duration and CPUTime columns in the first execution, located in the top highlighted section of the trace. Then review the same two columns in the lower highlighted section. What should be obvious is that a minimal amount of time is spent accessing the data, and also that the CPUTime has decreased significantly. This is all because the query has been cached.



**Figure 17-10:** Profile trace showing non-cached data and cached data

One more thing is leveraging the SQL Server Profiler. Looking at the example in Figure 17-10, an administrator can quickly detect which queries are leveraging the cache and which are not. Moreover, you can use other categories and events to diagnose other issues, such as poorly performing queries and disk subsystems. You can couple this trace with the Performance Monitor to provide even more detailed results.

## Using Tabular Models

By default, tabular models are meant to perform efficiently because the data is scanned in memory. Of course that is not the case. If the server is not properly sized, performance problems will likely arise. This is also the same reason these types of models perform well; if sufficient memory is allocated, people or applications accessing the model may experience out-of-memory exceptions. To mitigate this, connect to a tabular model using SSMS. Right-click the server and select Properties from the context menu. In the left navigation, select General. Scroll down the list of choices until you can see Memory\VertiPaqPagingPolicy. Change the value from 0 to 1 or 2.

Changing it to 1 causes the model to use the pagefile.sys and 2 uses memory-mapped files. If the value is zero (0) neither will occur and operations will be cancelled if sufficient memory is not available.

Another task that could be done is to ensure that only the necessary data has been imported into the model. Importing too much unnecessary data into the model could cause overuse or misuse of memory. As a result, data that needs to be in memory may be cached out to disk—that is, if you configured the server in such a way. If you did, performance may be degraded.

	Log \ FlightRecorder\Enabled Log \ FlightRecorder\FileSizeMB Log \ FlightRecorder\LogDurationSec Log \ FlightRecorder\SnapshotDefinitionFile Log \ FlightRecorder\SnapshotFrequencySec Log \ FlightRecorder\TraceDefinitionFile Log \ MessageLogs Log \ QueryLog\CreateQueryLogTable Log \ QueryLog\QueryLogConnectionString Log \ QueryLog\QueryLogSampling Log \ QueryLog\QueryLogTableName Log \ Trace\TraceReportFQDN LogDir MaxIdleSessionTimeout Memory\HardMemoryLimit Memory\HeapTypeForObjects Memory\LowMemoryLimit Memory\MemoryHeapType Memory\TotalMemoryLimit Memory\VertiPaqMemoryLimit <b>Memory\VertiPaqPagingPolicy</b> MinidleSessionTimeout Network\Listener\IPv4Support Network\Listener\IPv6Support Network\Listener\MaxAllowedRequestSize Network\ListenOnlyOnLocalConnections
<b>Connection</b>	Server: Connection\Tabular Connection: EDUDEMO

**Figure 17-11:** SSAS Tabular Model General Properties window

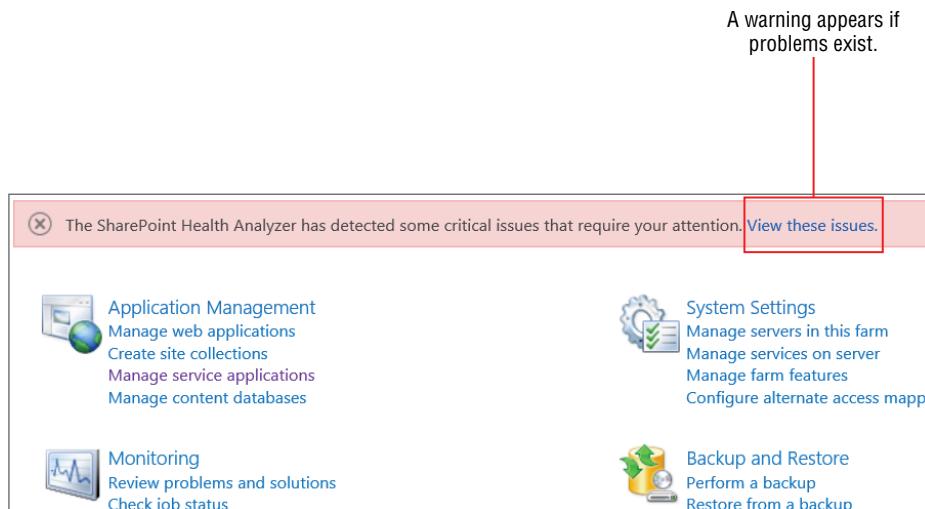
In addition, to minimize the amount of data, avoid string data types. Why? Because tabular models are based on columnar storage with dictionaries. These dictionaries hold distinct values. Therefore, the smaller the number of distinct values, the larger amount of data that can fit in memory. This is because the storage amount is based on the number of distinct values. For example, including a string in the model that contains hundreds or even thousands of distinct values could bloat the available memory with dictionaries that may be unnecessary. Therefore, developers should take a careful approach when selecting data that will occupy space in the model.

## Using SharePoint to Improve Performance

This section briefly discusses how you can use SharePoint to improve performance and also how to use certain aspects of SharePoint to monitor performance. To start, be sure to follow all best practices as they relate to running SQL Server that hosts a SharePoint environment. Following some of the SQL Server 101 best practices, such as proper disk-subsystem design and proper database placement, is critical. Create multiple TempDB files across multiple disks and also increase the initial size and AutoGrowth settings.

In addition to SQL Server optimizations, properly sizing and ensuring that the system will scale is critical to SharePoint performance. Instead of consolidating

application and web servers on a single machine, scale them out into dedicated servers. This type of deployment is similar to that shown in Figure 17-2, which illustrates an SSRS SharePoint integrated scaled-out deployment. Not only should SSRS be scaled, but any other Service Applications as well that may have large amounts of activity, such as concurrent users. Finally, SharePoint also includes a Health Analyzer, and each time the Central Administration tool is loaded, a warning appears if any problems exist, which is shown in Figure 17-12.

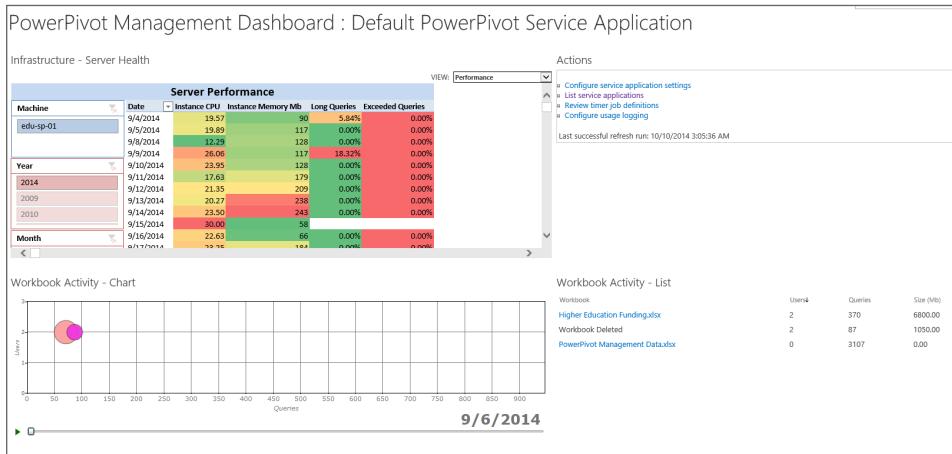


**Figure 17-12:** SharePoint Central Administration tool showing Health issues

Clicking the View these issues link opens a page providing a detailed listing and description of all possible issues.

Not only does SharePoint provide tools that monitor itself, but there are also tools and features that assist in monitoring specific integrations such as SSRS and Power Pivot. For example:

1. Open the Central Administration tool and click Application Management in the left navigation menu.
2. On the Application Management page under the Service Applications section, click Manage Service Applications.
3. Locate the Power Pivot Service application and click the link on the first occurrence; the PowerPivot Management Dashboard opens, shown in Figure 17-13.



**Figure 17-13:** PowerPivot Management Dashboard

This page provides two reports that list and visualize workbook activity. This activity assists in identifying workbook size and number of users and queries. From this an administrator can decide if a workbook is under- or over-utilized and can also see the size of the workbook. You can make suggestions as to whether you actually need the workbook and if you need to optimize it because it is too large.

One final thing: Many capabilities are available to monitor SSRS Integrated into SharePoint. You can configure verbose logging similar to SSRS in native mode. To do so:

1. **Return to the Service Applications page in the Central Administration tool.**
2. **Click the first occurrence of the SSRS service application.**
3. **On the Manage Reporting Services Application page, click System Settings.** Scroll all the way to the bottom of the page. Change the ExecutionLogLevel from Normal to Verbose and click OK. This enhances and increases the logging capabilities in the ExecutionLog views.

SharePoint also includes diagnostic categories that you can use for monitoring other business intelligence capabilities. To do so:

1. **Open Central Administration and click Monitoring.** In the Reporting section, click Configure diagnostic logging.
2. **Review the list of choices.** Notice that this list contains PerformancePoint, Power Pivot, and SQL Server Reporting Services categories.
3. **Expand the SSRS Category and review the list.** Figure 17-14 displays an abbreviated list of these categories.

<input checked="" type="checkbox"/> SharePoint Translation Services	Information	Monitorable
<input checked="" type="checkbox"/> SQL Server Reporting Services	Information	Medium
<input type="checkbox"/> Power View	Information	Medium
<input type="checkbox"/> Report Server Alerting Runtime	Information	Medium
<input type="checkbox"/> Report Server App Domain Manager	Information	Medium
<input type="checkbox"/> Report Server Buffered Response	Information	Medium
<input type="checkbox"/> Report Server Cache	Information	Medium
<input type="checkbox"/> Report Server Catalog	Information	Medium
<input type="checkbox"/> Report Server Chunk	Information	Medium
<input type="checkbox"/> Report Server Cleanup	Information	Medium
<input checked="" type="checkbox"/> Report Server Configuration Manager	Information	Medium
<input type="checkbox"/> Report Server Crypto	Information	Medium
<input type="checkbox"/> Report Server Data Extension	Information	Medium
<input type="checkbox"/> Report Server DB Polling	Information	Medium
<input type="checkbox"/> Report Server Default	Information	Medium
<input type="checkbox"/> Report Server Email Extension	Information	Medium
<input type="checkbox"/> Report Server Excel Renderer	Information	Medium

**Figure 17-14:** SharePoint SSRS diagnostic categories

4. **Click a few of these choices.** This logs them to the SharePoint ULS log, which is located here: C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\15\LOGS\.
5. **After a short time, log check the most recent file in this directory.** Information about SSRS should be logged in the document.

## Summary

---

This chapter explained techniques that assist in tuning various aspects of the business intelligence deployment. In the end, each of these methods should provide an efficiently performing environment.

The next chapter looks at the ways to scale business intelligence environments, the benefits of scaling, which tools to use, and how these techniques can be combined.

# Scaling the Business Intelligence Environment

The business intelligence environment contains many different products and servers. When performance issues arise and you have exhausted all other tuning options, it is time to look at scaling your solution. Scaling comes in many forms, but is typically hardware related. Scaling each of the business intelligence tools helps alleviate the performance issues and makes your users happy again.

This chapter explains the scaling options for your business intelligence environment. First, you learn what scaling means, when you should scale, and what benefits you gain from scaling. Then, you learn about the available scaling options and the different scaling architectures for several of the enterprise business intelligence tools. Putting all the pieces together lets you scale the entire business intelligence environment to provide the best solution for your users.

## Why Would You Scale the Business Intelligence Environment?

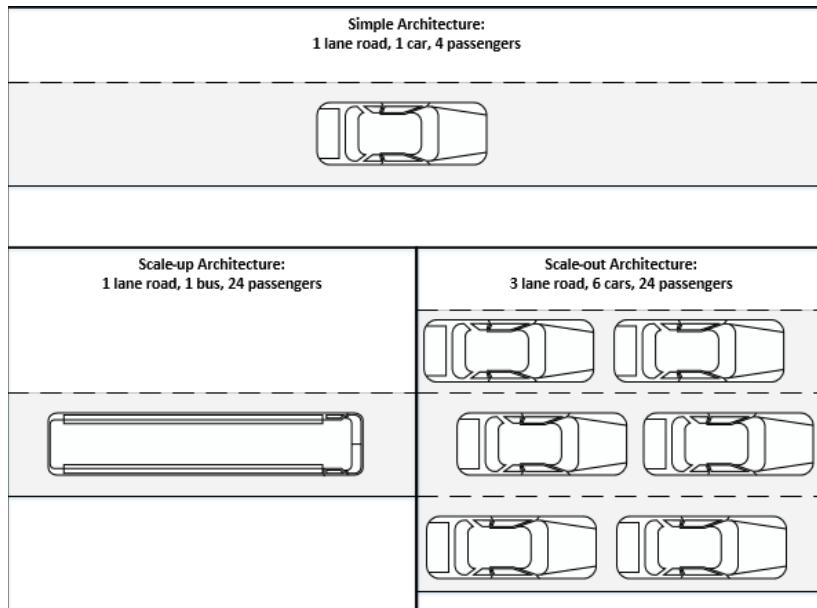
---

As your reporting and analysis needs increase, your business intelligence solution must grow also. This growth is called *scale* and can come in two fashions: scale up or scale out. A scale-up approach means adding more memory, hardware, or processing power to allow the existing services to run more efficiently. A scale-out approach includes adding new servers or services to extend the number of nodes working to provide the service.

**SCALING**

*Scaling* is the process of increasing software or hardware to handle growth of inputs, processing, or outputs.

To explain the concept of scale up versus scale out, consider driving down a one-lane road in a regular, four-passenger car. This is the simple architecture. You could take that simple architecture and beef it up by swapping the car for a bus. The bus would still use the one-lane road, but it can contain more passengers. This gives a scale-up architecture. On the other hand, you could increase the number of lanes and vehicles, which would give a scale-out architecture. The visual of these two approaches is shown in Figure 18-1.



**Figure 18-1:** Scaling analogy

Many business intelligence solutions provide great value without ever having to scale their tools. If you project that the number of users and/or requests will increase over time, or if you see poor performance due to a high number of users and/or requests, you need to take action. Try performance tuning the individual tools first by decreasing query time and optimizing the solutions. If you still have performance issues, you can turn to scaling.

In general, try to scale up your solution before scaling out. Because the tool or service typically handles scaling up, you will get more bang for your buck. Additionally, you do not need to set up an entirely new server (and license!) as you would for scaling out.

By scaling your environment, you gain many benefits. Processing, querying, and response times typically decrease. You can also increase the number of concurrent users and handle more requests. You can even reduce memory errors and timeouts, resulting in much happier users. You often gain higher availability by scaling out your server as well.

As with any added complexity to an architecture, you must support your change through documentation, maintenance, and disaster recovery scenarios. Definitely weigh this con against the pros of scaling out before making this leap. If you do take the leap, however, you can make your solution much faster and more reliable.

## How Do You Scale Each Tool?

---

Each business intelligence tool involves a different process to scale. The process can involve hardware, software, and different configurations. The following sections discuss four of the enterprise business intelligence tools that you have learned about: Analysis Services, Reporting Services, Power Pivot, and Power View. You will learn about how to scale up and scale out each tool.

### Using Analysis Services (Multidimensional or Tabular)

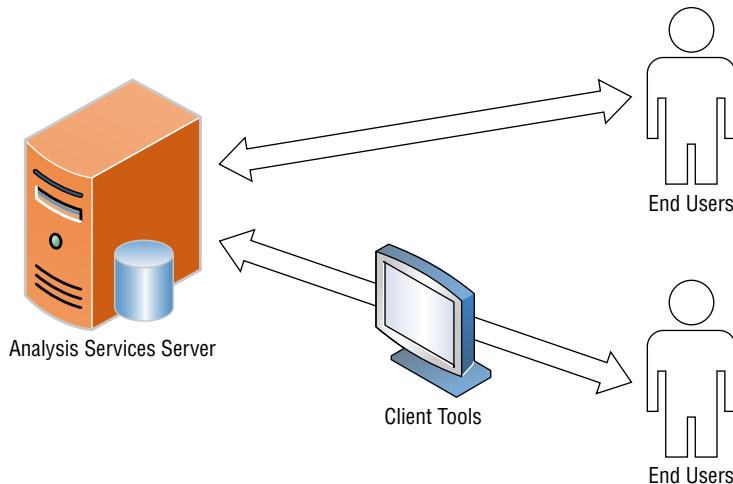
As discussed in Chapters 7 and 8, Analysis Services is a semantic modeling tool that provides a business focus and fast querying. As a part of SQL Server, you can install Analysis Services separately from any other services. Users access Analysis Services either directly, through an interface such as Excel, SQL Server Management Studio, or a third-party tool, or through another application, such as Reporting Services or Power View.

Analysis Services in multidimensional mode uses quite a bit of memory to satisfy queries for the end users. It performs its calculations in memory as part of accessing the formula engine. Analysis Services in tabular mode uses quite a bit of memory because it pulls the entire database into memory to provide the fast querying. Of course, loading all that information into memory is CPU-intensive. So overall, Analysis Services uses a lot of memory and processing power.

After the Analysis Services database has been processed and loaded with data, the querying architecture is rather simple. The diagram in Figure 18-2 shows this simple architecture, which contains users who either use client tools or directly access the Analysis Services database.

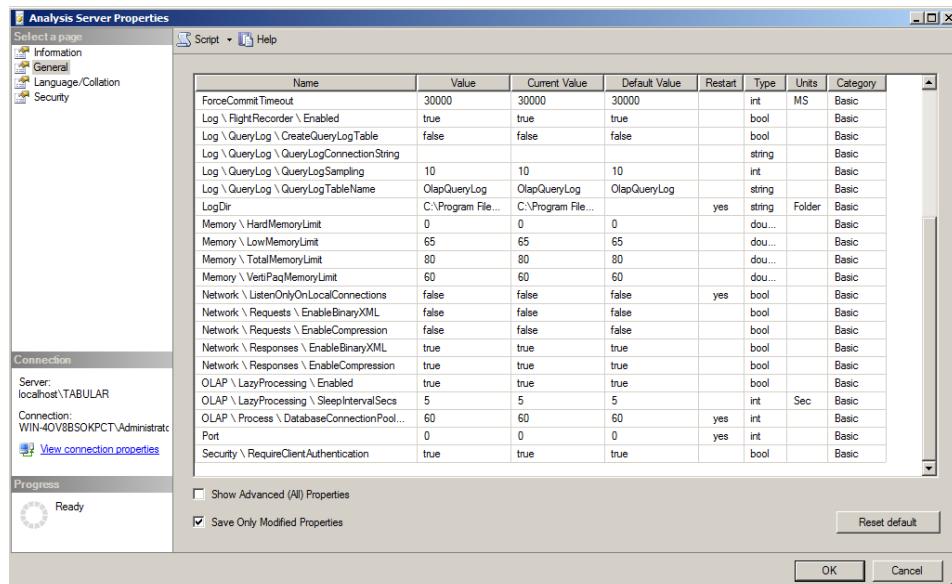
#### *Scaling Up*

When discussing scale up for Analysis Services, you typically want to add more memory. This allows both multidimensional and tabular models to return queries more quickly. Start by adding memory to the server, and then telling Analysis Services to use the additional memory.



**Figure 18-2:** Simple Analysis Services architecture

Once you add memory to the server, you need to verify and potentially adjust several properties, which you can access through SQL Server Management Studio (SSMS). After connecting to the Analysis Services server, right-click the name of the server and select the Properties menu option. On the General menu, you will see the basic memory properties shown in Figure 18-3.



**Figure 18-3:** Analysis Server Properties screen

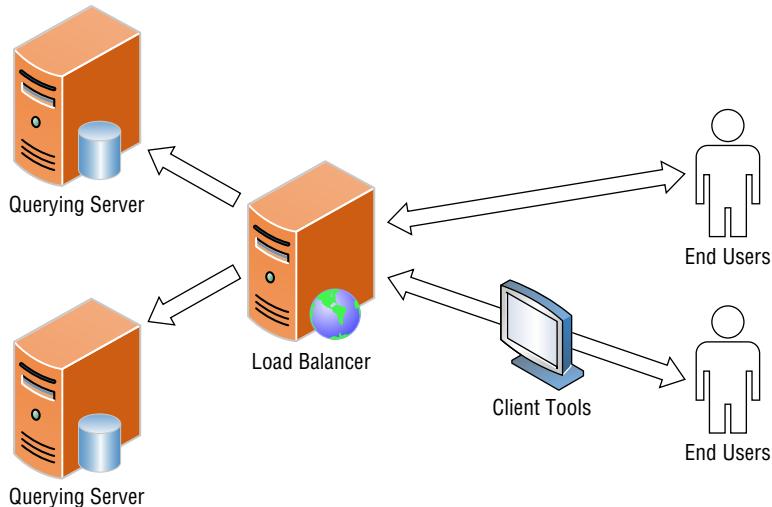
The following list describes each of these properties:

- **LowMemoryLimit:** The minimum amount of memory that Analysis Services reserves for its processes. This is a percentage of the total amount of memory on the server. Once the memory usage on the machines reaches this value, Analysis Services releases memory until the value is reached. By default, this value is 65.
- **TotalMemoryLimit:** The threshold amount of memory that Analysis Services uses before more aggressively releasing memory. This is a percentage of the total amount of memory on the server. By default, this value is 80.
- **HardMemoryLimit:** The maximum amount of memory that Analysis Services uses before rejecting requests until the memory is released. This is a percentage of the total amount of memory on the server. By default, this value is 0, which means between the value of *TotalMemoryLimit* and the total amount of memory on the server.
- **VertiPaqMemoryLimit:** (only for tabular instances) The threshold amount of memory that Analysis Services uses before paging to disk. This is a percentage of the total amount of memory on the server. Once the memory usage drops below this value, the Analysis Services server uses only memory. By default, this value is 60.

If any of these values have been changed and you have increased the amount of memory, you may want to increase or decrease this value based on how much memory is needed by any other applications on the server. Adjust as necessary, restart the service, and your server will start using the additional memory!

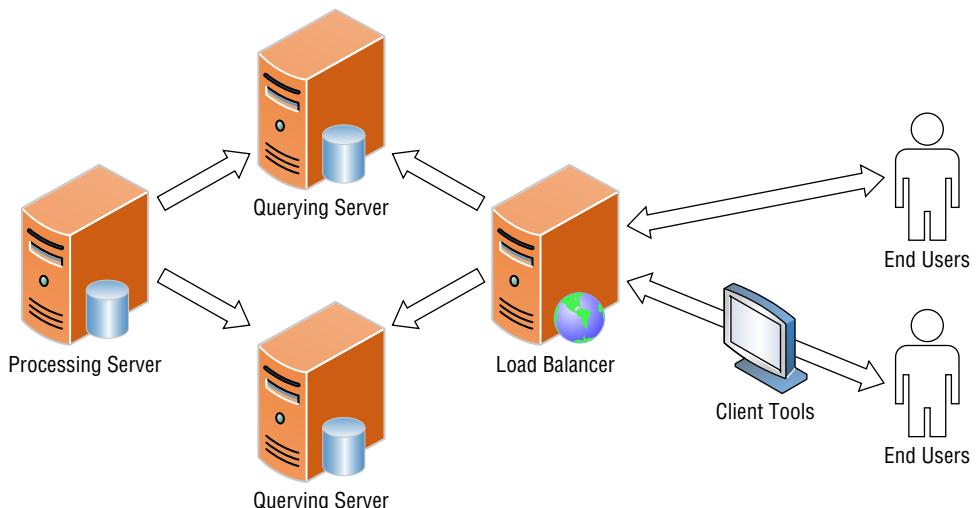
### ***Scaling Out***

If a scale up does not provide the performance enhancement you want, you can move to scaling out your Analysis Services servers. You can start by adding additional servers to respond to the query requests. The nice part of the scale-out process for querying is that a load balancer can distribute the query requests to each of the servers, as shown in Figure 18-4. You install and set up each querying server separately, and each server processes its databases. By scaling out the querying servers, you can handle more concurrent user requests.



**Figure 18-4:** Scaled-out Analysis Services architecture

However, the previous architecture diagram does not prevent contention between processing demand and query requests because each query server still has to process its own database. To prevent this, you can add a third server to handle only the processing. Once the processing server has completed processing the database, you can synchronize the database from the processing server to the querying servers. The synchronization takes less time than a full process, reducing the chance of contention. This architecture is shown in Figure 18-5.



**Figure 18-5:** Scaled-out Analysis Services architecture with processing server

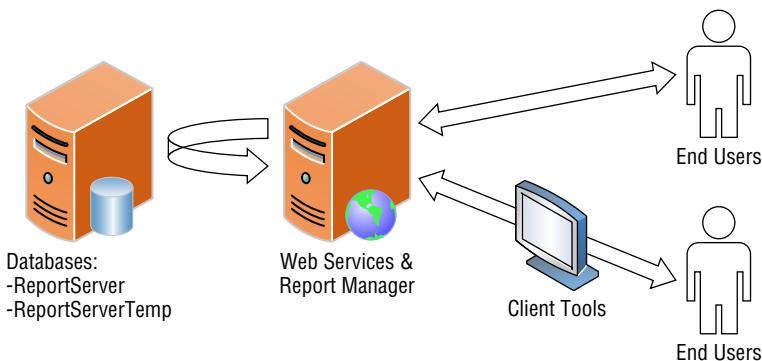
For more information on scaling out Analysis Services and synchronization, see the Microsoft white paper “Scale-Out Querying for Analysis Services with Read-Only Databases”: [https://technet.microsoft.com/en-us/library/ff795582\(v=sql.100\).aspx](https://technet.microsoft.com/en-us/library/ff795582(v=sql.100).aspx).

## Reporting Services

As discussed in Chapter 11, the Reporting Services server has three components:

- The Reporting Services engines
- The databases to store the information
- A web service layer to access the report information

Depending on which mode you used to install Reporting Services, you can also access the reports through Report Manager or a SharePoint site. Figure 18-6 shows the components in a simple server architecture of a native mode installation.



**Figure 18-6:** Simple Reporting Services architecture

### Scaling Up

For a scale-up approach for Reporting Services, your most likely option is to increase the amount of memory available for Reporting Services to use. Reporting Services uses memory for Report Manager and the windows and web services in the engine. Once you have added the additional memory to the server, you can tell Reporting Services to use the additional memory within the `rsreportserver.config` file if the values have been previously set. By default, the configuration file includes two properties where you can change the values:

- **MemoryThreshold:** The value that Reporting Services uses as a percentage of the `WorkingSetMaximum` value to determine at what memory value it will slow down in taking requests until the memory has decreased. By default, this value is 90.

- **MemorySafetyMargin:** The value that Reporting Services uses as a percentage of the WorkingSetMaximum value to determine the memory threshold value between operations' resources and system resources. By default, this value is 80.

If someone has manually added properties to the configuration file, you need to increase two more properties:

- **WorkingSetMaximum:** The largest amount of memory that Reporting Services uses. By default, the value is set to the amount of memory on the server.
- **WorkingSetMinimum:** The smallest amount of memory that Reporting Services uses. By default this value uses 60 percent of the WorkingSetMaximum value.

A restart of the Reporting Services service enables it to use the new values and use the extra memory.

### *Scaling Out*

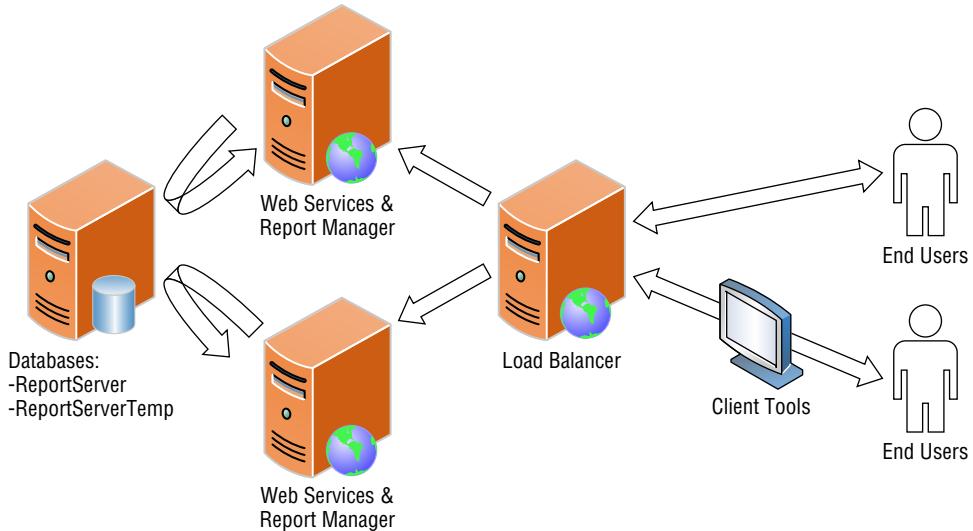
The component that provides the most benefit from a scale-out approach is the report server that contains the web services and report manager. The report manager and web services retrieve information from the ReportServer database, which stores the report item metadata, including folder structure, report metadata, and security information. Scaling out the application server allows more people to access the reports and reduces the amount of time the server takes to return the report to the end user.

To scale out the web application, you have three tasks to accomplish:

- Add additional servers to the web application layer.
- Provide a way for end users and client tools to access the application.
- Point everything on the web server side to use the same back-end databases.

After completing these tasks, the server architecture will look similar to Figure 18-7. Notice the additional web server compared to the simple server architecture.

The scale-out process takes place in the Reporting Services Configuration Manager, which you can find in a location similar to: Program Files  $\Leftrightarrow$  Microsoft SQL Server 2014  $\Leftrightarrow$  Configuration Tools  $\Leftrightarrow$  SQL Server 2014 Reporting Services Configuration Manager. After setting up the first server, you can use the Configuration Manager to connect to the second server. Complete the existing menu options, and select the existing report server database that you created with the first server. Navigate to the Scale-out Deployment menu and notice both servers in the list, with one having a status of "Waiting to Join." Click the Add Server button, and you have completed the scale-out process.



**Figure 18-7:** Scaled-out Reporting Services architecture

If you want to set up the server to use a network load-balanced cluster, you have additional steps to take. Follow the direction provided by Microsoft Books Online, found here: <https://msdn.microsoft.com/en-us/library/cc281307.aspx>. Additionally, if you have a SharePoint integrated report server, follow the directions discussed next under the Power Pivot and Power View section.

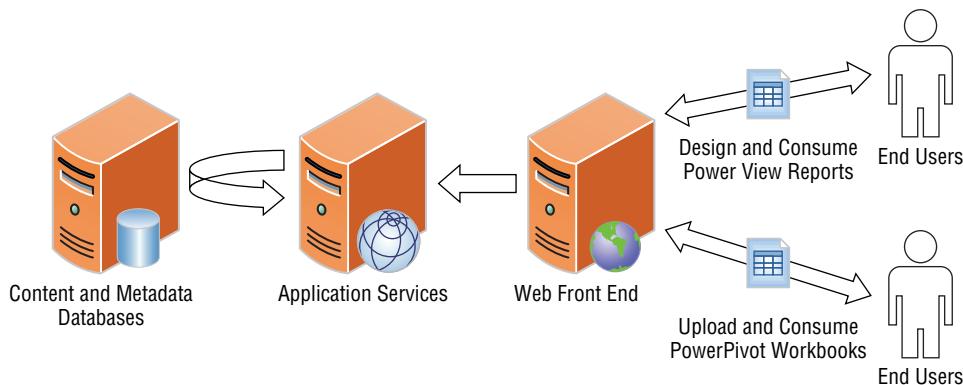
## Using Power Pivot and Power View

Chapter 6 introduced Power Pivot, the mash-up Excel-based tool that allows you to create semantic models for slicing, dicing, and pivoting. You can also upload your workbooks to a SharePoint Server Power Pivot Gallery for easy distribution and consumption of the data. Because scaling a business intelligence tool is for enterprise solutions, your Power Pivot scaling mostly focuses on the SharePoint Server side of things.

Chapter 12 introduced Power View, the interactive ad hoc visualization tool for analysts. Because Power View can use data from either a Power Pivot workbook or a tabular model, you have a similar option to deploy the report to a SharePoint Server. Figure 18-8 shows a simple SharePoint server architecture.

As shown in Figure 18-8, you have three layers in SharePoint:

- A web front-end server which handles client requests
- An application services server which handles the processing work
- A database server, which contains the content and metadata that SharePoint needs to run



**Figure 18-8:** Simple SharePoint architecture

You can simplify this architecture even further by combining these three servers into one server, but because the separation shown here is often the first scale-out recommendation for SharePoint, this is the simple architecture described here.

### Scaling Up

Keep in mind that Power Pivot and Power View have two different versions: the Excel version that sits on your desktop, and the service that lives inside SharePoint. You can scale up both methods.

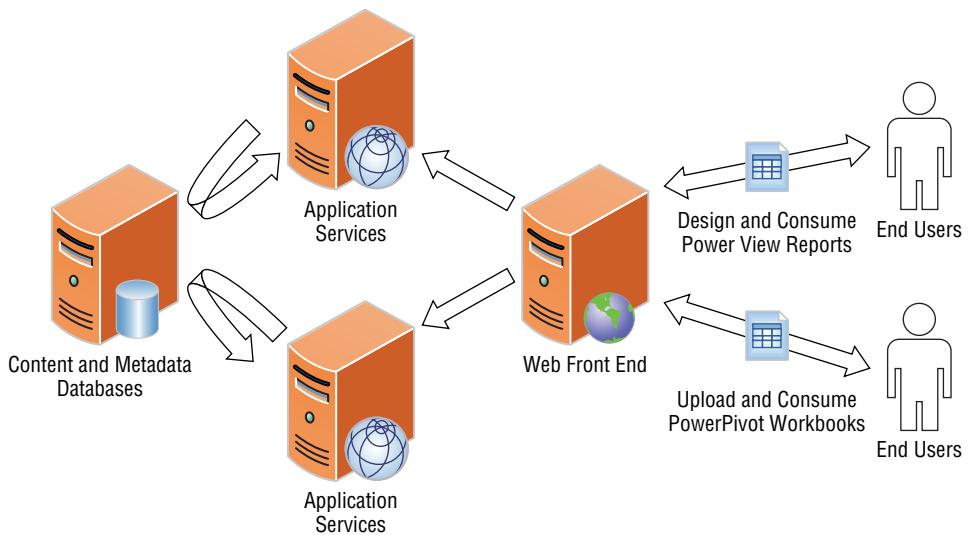
For the Excel versions, scaling up involves adding additional memory to your local machine. Because Power Pivot loads its entire model into memory to provide fast querying capability, memory is essential to the execution of the model. In addition, Power View uses that model from memory in its execution. If you use a 64-bit version of Excel, you have no limitations on your local machine; however, if you want to upload your workbook to SharePoint, the workbook's size must be less than 2 gigabytes. If you exceed that workbook size, your next best scale-up option is to put your Power Pivot model and data into an Analysis Services tabular model and connect to that on the server.

Within SharePoint, Power Pivot uses its own service and Excel Services; and Power View runs under the Reporting Services service. Additionally, there are front-end services for Excel, Power Pivot, and Reporting Services. You can tweak the configuration and memory values in SharePoint's Central Administration. For example, you can modify the amount of disk space that Power Pivot uses to cache the databases. For Power Pivot, go to the SQL Server Analysis Services service under Application Management → Manage Services on Server. The property you want to modify, *Total disk space*, is the maximum amount of size in gigabytes that Power Pivot can use. If you decrease or increase the amount of memory on the server, modify this property accordingly.

### Scaling Out

The nice thing about scaling out Power Pivot or Power View is that SharePoint manages the whole process. There are two main places where you can add servers to scale out these servers: either at the application services layer or at the web front-end layer. For each of these, you can add additional servers to the SharePoint farm to handle all services, or you can add servers solely for one service.

If your performance concern is related to the service taking a long time to respond, you want to scale out your application service servers. Add the application server by running the Power Pivot for SharePoint 2013 Configuration tool. Use the existing Analysis Services database that you previously created during the first server's installation, and use the same service account to run the service. Once you correctly add both servers to the farm, the web front-end server distributes requests to the application service servers. This architecture is shown in Figure 18-9.



**Figure 18-9:** Scaled-out Application SharePoint architecture

Alternatively, if you see performance issues in the designer or user interface, you want to look at scaling out the web front-end servers. In the case of Power Pivot and Power View, you can use the SQL Server installation file and install the Reporting Services Add-in for SharePoint Products and SQL Server Power Pivot for SharePoint features. Because users use the web front-end to access the application, you want to put a load balancer in front of the servers, so that users can access one server and the server will automatically redirect the request to the next web front end. This scale-out architecture is shown in Figure 18-10.

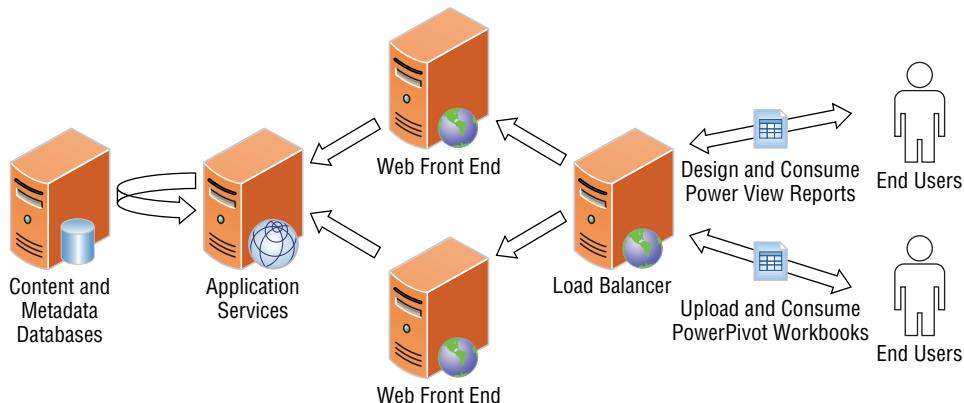


Figure 18-10: Scaled-out web front-end SharePoint architecture

## Summary

This chapter discussed the scaling options available to many of the business intelligence tools that you will use in your environment. Start by scaling up your services with memory or processing power, and then move on to scaling out your solution with additional server nodes. Although this chapter showed each option individually, you can easily implement multiple scale-up and scale-out servers within the solution.

Throughout this book, you have learned all about the Microsoft business intelligence stack. You know the different tools available and when you should use each one. You know how to make your solution flexible, scalable, and user friendly. The only thing left is for you to start implementing your own solution. Good luck!

# Index

## NUMBERS

3-D visualization, Power Map and, 15

## A

ad hoc reporting, 8, 10–11

aggregation, multidimensional model, 166–169

ALM (application lifecycle management), 351

environment server options, 352

AMO (Analysis Management Objects) library, 357

Analysis Services

deployment plan

AMO library, 357

ASSL, 356–357

Deployment Wizard, 355

manual backup, 354–355

Synchronize Database

Wizard, 356

scaling, 381–385

analytical models, 28–29

analytical reports *versus* operational, 228

animation, Power Map, 288–290

architecture

analytical model, 28–29

data architecture

approval, 46–47

big data model, 40

business goals and, 43–44

business requirements and, 43–44

cloud systems, 41–42

complexity and, 43

corporate analysis, 41

data repositories and, 43

data vault, 39

EDW (enterprise data warehouse), 37

finalizing, 46–47

fragility and, 42–43

hub and spoke data mart, 39–40

late-binding models, 40

maintenance and, 42–43

ODS (operational data store), 38

on-premise systems, 41–42

organizational structure and, 42

research selection factors, 42–44

selecting, 34–47

self-service analysis, 41

size and, 42

stakeholders, interviews, 44–46

system configuration and, 42–43

volume and, 43

data governance, 26–28  
 data sources  
   external, 24  
   internal, 23–24  
 data warehouses, 24–26  
 goals, 23  
 ASSL (Analysis Services Scripting Language), 356–357  
 audience for business intelligence solutions, 21–23  
 availability considerations, 31–32

**B**

bar charts, 256  
 big data models, 40  
 Bing maps, 275  
 BISM (Business Intelligence Semantic Model), 73, 74–75  
 data access layer, 75–77  
   MOLAP, 78  
   Power Pivot, 77  
   tabular model, 78–79  
 Data Model layer, 82  
 departmental, 84–85  
 model comparisons, 83–84  
 model selection, 84–87  
 organizational, 87  
 query languages  
   DAX, 79–81  
   Direct Query, 81–82  
   MDX, 81  
   ROLAP, 81–82  
 sources, 76  
 team, 86  
 business intelligence solutions  
   audience, 21–23  
   delivery solution, 29–30

**C**

caching data, 368–369  
 CDC (Change Data Capture), 5  
 charts  
   Power View  
     bar charts, 256

column charts, 256–257  
 line charts, 258  
 multiples, 261  
 pie charts, 258–259  
 scatter charts, 259–260  
 trellis, 277  
 code deployment, 352  
 column charts, 256–257  
 consuming reports, 217  
 continuous variables, 180–181  
 monotonic variables, 181

**D**

Dashboard Designer, 18–19  
 dashboards (PPS)  
   deployment best practices, 319–321  
   drilldown, 305–306  
   drillthrough, 305–306  
   dynamic measures, 306–309  
   Scorecard comments, 309–311  
   STPS (Simple Time Period Specification), 301–304  
 data access layer (BISM), 75–77  
   MOLAP, 78  
   Power Pivot, 77  
   tabular model, 78–79  
 data access types, 100–101  
 data architecture  
   approval, 46–47  
   big data model, 40  
   business goals and, 43–44  
   business requirements and, 43–44  
   cloud systems, 41–42  
   complexity and, 43  
   corporate analysis, 41  
   data repositories and, 43  
   data vault, 39  
   EDW (enterprise data warehouse), 37  
   finalizing, 46–47  
   fragility and, 42–43  
   hub and spoke data mart, 39–40  
   late-binding models, 40  
   maintenance and, 42–43  
   ODS (operational data store), 38

- on-premise systems, 41–42  
 organizational structure and, 42  
 research selection factors, 42–44  
 selecting  
     benefits, 35–36  
     challenges, 34–35  
     options, 36–47  
 self-service analysis, 41  
 size and, 42  
 stakeholders, interviews, 44–46  
 system configuration and, 42–43  
 volume and, 43  
 data caching, 368–369  
 data distribution, 181–182  
 data governance, 26–28  
 data mining, 173  
     Association Rules, 185, 186  
         measures, 187  
         parameters, 187–188  
         testing, 188–189  
     best practices, 185–201  
     business value, 174–179  
     Clustering, 185, 189–190  
         EM (Expectation-Maximization), 190  
         example, 191–192  
         hard clustering, 190  
         parameters, 190–191  
         soft clustering, 190  
     continuous learning cycle, 205–206  
     continuous values, 180–181  
         monotonic variables, 181  
     cube source, 202  
     cycle, 176–179  
     data distribution, 181–182  
     Decision Trees, 185  
         over-fitting, 194  
         recursive partitioning, 193  
     derived variables, 183  
     directed (supervised) approach, 174  
     discrete values, 180  
     DMX, 201, 204  
     DW processes, 206–207  
     ETL processes, 206–207  
     Excel and, 207–208
- Linear Regression, 186, 197–199  
 Logistic Regression, 186  
 model, 184–185  
     deployment, 202–203  
     maintenance, 205–206  
 Naïve Bayes, 185, 192–193  
 neural networks, 186, 194–195  
     activation functions, 196  
     backpropagation, 196–197  
     example, 197  
 Regression Trees, 185, 197–199  
 relational source, 202  
 reporting services and, 207  
 Sequence Clustering, 186, 199  
 structure, 184  
 test data sets, 182–183  
 Time Series, 186, 200–201  
 training, 182–183  
 undirected (unsupervised)  
     approach, 174  
 use cases, 175–176  
 variables, 174  
 Data Model layer (BISM), 82  
 data sources  
     external, 24  
     internal, 23–24  
 Power View  
     Excel, 247–248  
     SharePoint, 249–251  
     shared, 234–235  
 data transformation  
     aggregating data  
         Group By option, 66–67  
         Query Editor, Applied  
             Steps, 69–70  
     Transform ribbon, 67–69  
     combining from multiple  
         sources, 62–64  
     splitting data, 64–65  
 data types, Power Pivot, 99–103  
 data vault, 39  
 data warehouse, 179  
     building, 4–5  
     Power Pivot, 24–25

- database engine  
 data warehouse, 4–5  
 RDBMS, 5
- databases, importing data from, 57–59
- datasets, shared, 234–235
- date and time data types, 101–103
- DAX (Data Analytic Expression),  
 14, 79–81
- Decision Trees  
 over-fitting, 194  
 recursive partitioning, 193
- delivery framework (self-service),  
 331–332  
 assessments, 332–333  
 constraints, 336  
 data governance, 332  
 data quality, 337–339  
 data stewards, 334  
 DQS, 340–342  
 entities, 335  
 Excel and, 345–347  
 generalization, 335–336  
 industry compliance, 334–337  
 low-quality data, 333–334  
 master data, 337–339  
 MDS, 342–345  
 normalization, 335  
 overviews, 332  
 predicates, 335  
 propositions, 335  
 relational database model, 334–335  
 roles, 339–340  
 skillset, 340–342  
 specialization, 335–336  
 stakeholders, 334  
 subject matter experts, 334  
 success criteria, 348–349  
 target audience, 339–340  
 tools, 340–342  
 training plan, 340
- delivery solution, 29–30
- departmental business  
 intelligence, 84–86
- deployment plan, 351–353
- Analysis Services  
 AMO library, 357  
 ASSL, 356–357  
 Deployment Wizard, 355  
 manual backup, 354–355  
 Synchronize Database Wizard, 356
- implementation, 359  
 documentation, 360–361  
 scripts, 360  
 testing, 361  
 training, 362
- Reporting Services, 357–358  
 Migration Tools, 358–359  
 Report Manager, 358  
 Rs.exe utility, 359  
 SharePoint interface, 358  
 SSDT-BI, 358
- Deployment Wizard, 355
- derived variables, 183
- development tools  
 Dashboard Designer, 18–19  
 Report Builder, 19  
 SSDT (SQL Server Data Tools), 16–17  
 SSMS (SQL Server Management  
 Studio), 17–18
- dimensional model, 155
- Direct Query, 81–82
- DMX (Data Mining Extensions),  
 201, 204
- DQS (Data Quality Services), 27–28

**E**

- EDW (enterprise data warehouse), 37
- ESRI shapefiles, 275
- evaluation matrices, visualization  
 and, 221–223
- Excel  
 business intelligence and, 14–16  
 data mining, 207–208  
 Power Map, 15–16  
 Power Pivot, 14  
 Power Query, 14  
 Power View, 14–15
- external data sources, 23–24

**F**

files, importing data from, 59–60  
 filters  
   Power View, 264–266  
     cross-filters, 267–268  
     report URLs, 270  
     slicers, 266–267  
     tiles, 268–270  
   PPS, 313–317

**G**

geographic data, 273

**H**

hardware requirements, Power Pivot, 90  
 HOLAP (Hybrid Online Analytical Processing), 29  
 hub and spoke data marts, 39–40

**I**

importing data, 56  
   from database, 57–59  
 industry compliance  
   delivery framework, 334  
     constraints, 336  
     entities, 335  
     generalization, 335–336  
     normalization, 335  
     predicates, 335  
     propositions, 335  
     relational database model, 334–335  
     specialization, 335–336  
 internal data sources, 23–24

**K**

KDD (knowledge discovery in databases), 174  
 KPIs (key performance indicators), 89  
   PPS, 313–317

**L**

late-binding models, 40  
 line charts, 258  
 LOB (line of business), 179

**M**

M programming language, 277–278  
   Power Query and, 70–72  
 MDS (Master Data Services), 27–28  
   self-service delivery framework,  
     337–339, 342  
     architecture, 343  
     attributes, 345  
     business rules, 345  
     collections, 345  
     entities, 344–345  
     Excel and, 345–347  
     hierarchies, 345  
     reusability, 343–344  
 MDX (Multidimensional Expressions), 81  
 mining data, 173  
   Association Rules, 185, 186  
     measures, 187  
     parameters, 187–188  
     testing, 188–189  
   best practices, 185–201  
   business value, 174–179  
   Clustering, 185, 189–190  
     EM (Expectation-Maximization), 190  
     example, 191–192  
     hard clustering, 190  
     parameters, 190–191  
     soft clustering, 190  
   continuous learning cycle, 205–206  
   continuous values, 180–181  
     monotonic variables, 181  
   cube source, 202  
   cycle, 176–179  
   data distribution, 181–182  
   Decision Trees, 185  
     over-fitting, 194  
     recursive partitioning, 193  
   derived variables, 183  
   directed (supervised) approach, 174  
   discrete values, 180  
   DMX, 201, 204  
   DW processes, 206–207  
   ETL processes, 206–207

- Excel and, 207–208  
 Linear Regression, 186, 197–199  
 Logistic Regression, 186  
 model, 184–185  
     deployment, 202–203  
     maintenance, 205–206  
 Naïve Bayes, 185, 192–193  
 neural networks, 186, 194–195  
     activation functions, 196  
     backpropagation, 196–197  
     example, 197  
 Regression Trees, 185, 197–199  
 relational source, 202  
 reporting services and, 207  
 Sequence Clustering, 186, 199  
 structure, 184  
 test data sets, 182–183  
 Time Series, 186, 200–201  
 training, 182–183  
 undirected (unsupervised)  
     approach, 174  
 use cases, 175–176  
 variables, 174  
 models  
     data mining, 184–185  
     dimensional model, 155  
     multidimensional, 151–153  
         aggregations, 166–169  
         Business Intelligence Wizard, calculations and, 164–166  
         Cube Creation Wizard, 156–159  
         Data Source View, 153–156  
         data sources, 153–156  
         dimensions, 160–162  
         hierarchies, 162–164  
         Measures, 159–160  
         partitions, 166–169  
         processing  
             optimization, 170–171  
             processing modes, 170  
             storage modes, 169–170  
         querying, 171–172  
         tabular model comparison, 130  
     multiples, 261
- O**  
 ODS (operational data store), 38  
 Office 365, Power Map and, 291–292  
 operational reports, 227  
     *versus analytical*, 228  
 organizational business intelligence, 87

**P**

partitions  
 multidimensional model, 166–169  
 recursive partitioning, 193  
 performance, 30–31  
 Performance Point, 12–14  
 pie charts, 258–259  
 Power Map, 15–16, 273, 279  
   advantages, 274  
   aggregation visualization, 284–285  
   animations, 288–290  
   Bing maps, 275  
   date fields, 283–284  
   ESRI shapefiles, 275  
   features, 274  
   geography fields, 282–283  
   geospatial data, 284–290  
   layers, 281  
   map gallery, 275  
   Office 365 and, 291–292  
   requirements, 279–280  
   scenes, 280–281  
   spatial data, 275  
   temporal data, 284–290  
   time fields, 283–284  
   tours, 280  
     creating, 285–288  
     sharing, 291  
   visualization and, 219–221  
     3-D visualization, 15  
     deploying, 290–292  
     sharing, 290–292

Power Pivot, 14, 89  
 BISM data access layer, 77  
 data warehouses and, 24–25  
 DAX and, 14  
 enabling, 90–92  
 hardware requirements, 90  
 KPIs, 89  
 model design  
   columns, 103  
   data types, 99–103  
   DAX calculated measures, 103  
   importing, 92–99

reporting and  
 aggregation rules, 106–107  
 calculated columns, 114–118  
 column formatting, 106  
 column hiding/renaming, 104–106  
 data sorting, 121–122  
 DAX measures, 114–118  
 hierarchies, 118–119  
 KPIs, 119–120  
 relationships, 107–114  
 role-playing dimensions, 122–124  
 scaling, 387–390  
 software requirements, 90  
 Power Query, 14  
 aggregating data, 66–70  
 combining data, multiple sources, 62–64  
 downloading, 52–54  
 importing data, 56  
   from database, 57–59  
   from file, 61  
   from web, 59–60  
 installation, 55  
 license agreement, 53–54  
 M language, 70–72  
 Online Search window, 60  
 splitting data, 64–65  
 Workbork Queries window, 58  
 Power Query Formula Language, 277–278  
 Power View, 14–15, 245, 279  
 charts  
   bar charts, 256  
   column charts, 256–257  
   line charts, 258  
   multiples, 261  
   pie charts, 258–259  
   scatter charts, 259–260  
 data sources  
   Excel, 247–248  
   SharePoint, 249–251  
 filters, 264–266  
   cross-filters, 267–268  
 report URLs, 270

slicers, 266–267  
tiles, 268–270  
reports, 246  
creating, 251–252  
Excel and, 263–264  
exporting, 271–272  
tables, 253–254  
views, multiple, 252–253  
visualizations, 253  
scaling, 387–390  
system requirements, 246–247  
trellis chart, 277  
visualizations, 218–219  
cards, 261–262  
converting, 254–255  
maps, 262–263  
matrices, 255–256

PPS (PerformancePoint Services),  
293, 294  
Analytic Charts, 295  
Analytic Grids, 296  
Application settings, 326–328  
Content Libraries, 317–319  
dashboards  
deployment best practices, 319–321  
drilldown, 305–306  
drillthrough, 305–306  
dynamic measures, 306–309  
Scorecard comments, 309–311  
STPS (Simple Time Period  
Specification), 301–304

Data Connections, 317–319  
deployment, 317–325  
Excel Services Workbooks, 296–297  
filters, 313–317  
KPIs, 313–317  
reports  
Decomposition Tree, 298  
KPI Details, 297  
Reporting Services, 297  
Scorecard, 298  
Strategy Map, 297  
Web Page, 297  
SharePoint and, 300–301

SSRS, 311–313  
Unattended Services Account,  
325–326  
Web Parts, 321–325  
predictive analytics, 174  
publishing reports, 217

## R

RDBMS (Relational Database  
Management System), 5  
recursive partitioning, 193  
Report Builder, 19, 229–230  
Reporting Services. *See SSRS (SQL  
Server Reporting Services)*  
reports  
data mining intergation, 207  
operational, 227  
Power Map and, 274–280  
Power Pivot  
aggregation rules, 106–107  
calculated columns, 114–118  
column formatting, 106  
column hiding/renaming, 104–106  
data sorting, 121–122  
DAX measures, 114–118  
hierarchies, 118–119  
KPIs, 119–120  
relationships, 107–114  
role-playing dimensions, 122–124  
Power View  
creating, 251–252  
Excel and, 263–264  
exporting, 271–272  
tables, 253–254  
views, multiple, 252–253  
visualizations, 253

PPS (PerformancePoint Services),  
293–294  
Analytic Charts, 295  
Analytic Grids, 296  
Application settings, 326–328  
Content Libraries, 317–319  
dashboards, 305–311, 319–321  
Data Connections, 317–319

- Decomposition Tree, 298  
 deployment, 317–325  
 Excel Services Workbooks, 296–297  
 filters, 313–317  
 KPI Details, 297  
 KPIs, 313–317  
 Reporting Services, 297  
 Scorecard, 298  
 SharePoint and, 300–301  
 SSRS, 311–313  
 Strategy Map, 297  
 Unattended Services Account, 325–326  
 Web Page, 297  
 Web Parts, 321–325  
 pre-rendering, 368–369  
 source control, 231–233  
 version control, 231–233  
**ROLAP (Real-time Online Analytical Process), 29, 81–82**
- S**  
 scaling, 379–381  
 Analysis Services, 381–385  
 Power Pivot, 387–390  
 Power View, 387–390  
 Reporting Services, 385–387  
 scatter charts, 259–260  
 self-service delivery framework, 331–332  
 assessments, 332–333  
 data governance, 332  
 data stewards, 334  
 DQS (Data Quality Services), 337–342  
 Excel and, 345–347  
 Excel and, 345–347  
 industry compliance, 334  
 constraints, 336  
 entities, 335  
 generalization, 335–336  
 normalization, 335  
 predicates, 335  
 propositions, 335  
 relational database model, 334–335  
 specialization, 335–336  
 low-quality data, 333–334  
 MDS (Master Data Services), 337–339, 342  
 architecture, 343  
 attributes, 345  
 business rules, 345  
 collections, 345  
 entities, 344–345  
 Excel and, 345–347  
 hierarchies, 345  
 reusability, 343–344  
 overviews, 332  
 roles, 339–340  
 stakeholders, 334  
 subject matter experts, 334  
 success criteria, 348–349  
 target audience, 339–340  
 training plan, 340  
 semantic model (SSAS), 6  
 shared data sources, 234–235  
 shared datasets, 234–235  
 SharePoint, 11–12, 375–378  
 PPS and, 300–301  
 software requirements, Power Pivot, 90  
 source control, reporting and, 231–233  
 SQL Server, database engine  
 data warehouse, 4–5  
 RDBMS, 5  
 SSAS (SQL Server Analysis Services), 6–7  
 multidimensional models, 372–374  
 semantic model, 6  
 tabular models, 374–375  
 SSDT (SQL Server Data Tools), 16–17, 229–230  
 SSMS (SQL Server Management Studio), 17–18  
 SSRS (SQL Server Reporting Services), 215–216, 227, 228–229, 278–279  
 ad hoc reporting, 8, 10–11  
 charts, 230  
 client component, 229  
 consuming reports, 217

data caching, 368–369  
 datasets, shared, 234–235  
 deployment plan, 357–358  
   Migration Tools, 358–359  
   Report Manager, 358  
   Rs.exe utility, 359  
   SharePoint interface, 358  
   SSDT-BI, 358  
 development tools, 216  
 diagnostic logging events, 363–365  
 drilldown/drillthrough, 241–243  
 engine, 229  
 ExecutionLog views, 369–372  
 filters, 240–241  
 memory, configuration, 365–368  
 Native mode, 7  
 Operational Reports, 7, 8–10  
 parameters, 240–241  
 performance, 237–239  
 PPS and, 311–313  
 publishing reports, 217  
 Report Builder, 229–230  
 reports, pre-rendering, 368–369  
 requirements, 217  
 RSReportServer, 365  
 scaling, 385–387  
 server component, 229  
 SharePoint Integrated mode, 7  
 sources, shared, 234–235  
 SSDT, 229–230  
 tablix, 230  
 templates, 236–237  
 textboxes, 230  
 visualizations, creating, 239–240  
 stakeholders, interviews, 44–46  
 Synchronize Database Wizard, 355

**T**

tabular model, 127–130  
 BISM data access layer, 78–79  
 calculated columns, 135–136  
 data import, 131–134  
 development, 130–131  
 hierarchies, 137–140

measures, 136–137  
 multidimensional model  
   comparison, 130  
 partitions, 141–144  
 perspectives, 140–141  
 processing optimization, 144  
   processing modes, 145  
   storage modes, 145–146  
 query optimization, 147–149  
 relationships, 134–135  
 team business intelligence, 86  
 templates, Reporting Services,  
   236–237  
 TFS (Team Foundation Server), 231  
 trellis charts, 277  
 TSQL (Transact-SQL), 17

**U**

UDM (Unified Dimensional  
 Modeling), 73  
 URLs (Uniform Resource Locators),  
 reports, 270

**V**

variables  
   continuous, 180–181  
   monotonic, 181  
   derived, 181  
 version control, reporting and,  
   231–233  
 visualization  
   Power Map, 15  
   aggregation, 284–285  
   deploying, 290–292  
   sharing, 290–292  
 Power View  
   cards, 261–262  
   converting, 254–255  
   maps, 262–263  
   matrices, 255–256  
 Reporting Services, 239–240  
 visualization tools, 211–212  
   business capabilities, 214  
   evaluation matrices, 221–223

information gathering, 215  
Power Map, 219–221  
Power View, 218–219  
Reporting Services, 215–216  
    consuming reports, 217  
    development tools, 216  
    publishing reports, 217  
    requirements, 217  
selection, 213–214

technical capabilities, 214  
users, 212–213

**W-Z**

web, importing data from, 59–60  
wizards  
    Cube Creation Wizard, 156–159  
    Deployment Wizard, 355  
    Synchronize Database Wizard, 355

# **WILEY END USER LICENSE AGREEMENT**

Go to [www.wiley.com/go/eula](http://www.wiley.com/go/eula) to access Wiley's ebook EULA.