

A vibrant field of orange lilies and green foliage. The flowers are in various stages of bloom, with some fully open and others as buds. The background is a dense field of similar flowers, creating a sense of depth. The overall scene is bright and natural.

**Gerald V. Post**

# **Data Mining Applications/2e**

**ANALYZING BUSINESS DATA**

# Data Mining Applications

Analyzing Business Data

**Version 2.0.0**

Gerald V. Post

*University of the Pacific*

Data Mining Applications/2e  
Analyzing Business Data

Copyright © 2013 by Gerald V. Post

All rights reserved. No part of this publication may be reproduced or distributed in any form or stored in any database or retrieval system without the prior written consent of Gerald V. Post.

Students:

Your honesty is critical to your reputation. No company wants to hire a thief—particularly for jobs as critical as application development and database administration. If someone is willing to steal something as inexpensive as an e-book, how can that person be trusted with billions of dollars in corporate accounts?

You are not allowed to “share” this book in any form with anyone else. You cannot give or sell any information from this publication in any form to anyone else.

A note on the cover photo:

Even in a wildflower preserve you can find patterns and clusters.

Or you can think of data mining as a way to search through acres of hills, flowers, sand, and ocean to find interesting facts.

Or can data mining software find the physical site of the image?

To purchase this book or other books: <http://JerryPost.com/Books>



For my students.





## Business Applications of Data Mining

---

Most managers commonly report that they are buried in data. Database management systems, enterprise resource planning systems, and Web sites now collect and store transactions from all aspects of business and organizations. Managers have access to terabytes (or more) of historical data. But how is this data supposed to be used? Database management systems can create standard reports and charts, and query languages (SQL) can be used to answer specific questions. But it is not enough.

Managers need statistics to analyze data—but few managers are statisticians, and configuring and interpreting traditional statistical tools can require extensive experience. Data mining (DM) tools were created partly from statistics and partly from computer science to analyze large data sets. Some tools can run relatively unsupervised—simply select the data and run the routines to obtain results and recommendations. Other tools require more supervision, from selecting variables to tweaking models to interpreting results and reconfiguring the analysis.

If you want a job as a manager, you need to know how to use information technology. But it is not simply a matter of knowing how to use a word processor or spreadsheet. You need to use the technology to collaborate with other workers, to analyze data, and to find ways to improve your organization.

Over the past decade, the field of data mining, or business intelligence or analytics, has received increased attention by academics, managers, and tool vendors. Notably, the tools have improved dramatically in terms of ease of use and interpretation. Many books and Web sites focus on data mining; but most of them emphasize either the statistics or the computer science issues. Few of them deal with the managerial issues of how to configure the tools, apply them to business problems, and interpret the results to make decisions. This focus is the main purpose of this book.

This book primarily focuses on using SQL Server Analysis Services—the data mining component of SQL Server. This tool is useful because it is widely available to universities through the MSDN Academic Alliance. Trial versions are also available to anyone via the Microsoft Web site. It is also relatively easy to use—although a few tips and tricks are necessary to fully understand some of the tools. Other tools, including the open-source Weka tool are used to demonstrate a few techniques. Many other tools exist and could be used to analyze the data provided with this book.

## Learning Assessment

---

After finishing the book, students should be able to understand the primary data mining tools, know what types of problems they are used to evaluate, and understand how to interpret the output and apply the results of the tools to help make business decisions.

Anyone who teaches this material knows that this learning objective is difficult to meet—because of the huge number of possible issues and the flexibility required in analyzing problems. To meet the objective, smaller, and more concrete goals are presented in each chapter. Each of these is spelled out as a series of questions at the start of the chapter. By the end of each chapter, students should be able to provide intelligent answers to the various questions.

## About the Book

### Organization

The book begins with an introduction to database queries because queries are often used to configure data for specific tools and because queries are useful for finding answers to specific questions. Students do not need a database course to be able to use this book. But, this book only lightly covers queries and just touches on database design issues. Chapter 3 explains the purpose and configuration of OLAP data cubes and the basic cube browsers. It emphasizes the importance of identifying facts and dimensions in any problem and then shows how the cube browser computes subtotals and filters data. It also explains how to configure hierarchies of various data dimensions. Chapter 4 reviews basic concepts in probability and statistics. The objective is to explain all of the concepts needed in the rest of the book. It would be best if students have already had a separate course in probability and statistics. .

The second section explores the fundamental data mining tools used in business analysis. Note that the chapters are orga-

nized by business goals as opposed to by individual tools. Chapter 5 examines basic clustering tools used to define groups—it is particularly useful in marketing to identify groups of customers that have certain attributes in common. Chapter 6 covers association or market basket analysis. It explains how probability computations are used to identify which items are associated or commonly purchased together. Chapter 7 is a long chapter because it explores how dimensions are related to fact dimensions using basic tools such as regression, logistic regression, and neural networks for nonlinear analysis. Chapter 8 covers the basics of time series analysis.

Clearly, all tools covered in this section could be explained in greater detail. In fact, individual books have been written about each type of analysis. But, the point of this book is not to make managers experts in every single tool. Instead, the goal is to make it possible for managers to apply the right technique needed for a specific decision and to correctly interpret the results and make better decisions.

Part Three is new in the second edition. Chapter 9 covers geographic information. This section is more about visualization and display of data than about searching. Many items in business are related to location and it is often helpful to examine decisions in that context. The chapter also introduces sequence analysis. Much of the hard-core work in sequences has been done in biology—specifically DNA analysis. But some of the tools are useful for certain business problems such as tracing Web browsing patterns. Chapter 10 introduces the basic foundations of multidimensional expressions (MDX). This language was initially defined by

#### Chapter 1: Introduction

#### Part One: Fundamental Tools

- Chapter 2 Finding and Storing Data
- Chapter 3 OLAP Cubes
- Chapter 4 Probability and Statistics Summary

#### Part Two: Business Analysis

- Chapter 5 Cluster Analysis
- Chapter 6 Association and Market Baskets
- Chapter 7 Evaluation of Dimensions
- Chapter 8 Time Series Analysis

#### Part Three: Specialized Tools

- Chapter 9 Geographic and Sequence Analysis
- Chapter 10 Multidimensional Expressions (MDX)

Microsoft but has been adopted by many vendors. It is useful for creating complex computations that are difficult to evaluate with other methods..

## Features That Focus on Solving Problems

Why do students need a textbook? Most definitions exist on the Web and many Web sites and blogs contain information on data mining, individual tools, and even how to solve specific problems. And yes, if you are serious about data mining, you should use all of these resources to learn as much as possible about the details. But students still need a textbook. This textbook provides a structure to the information. It teaches you how the many pieces fit together to solve business problems. It provides the context and ideas on how to analyze the millions of tidbits of data floating on the Web. Use the textbook to provide the foundation knowledge that describes data mining, how to configure data, and how to interpret results. As technology evolves and new problems are encountered, use the Internet to share ideas and solutions with others.

Each chapter contains several unique features to assist in understanding the material and in applying it to analyze data and make business decisions. The most important parts of each chapter are the examples. The chapters explain the basic problem, how to configure the data, how to run the analysis, and how to interpret the results. The writing explains the process, but it generally avoids a step-by-step presentation. The problem with step-by-step presentations is that readers focus too heavily on tracking steps and losing attention on the overall process. Also, as tools change, the steps quickly become obsolete, although the overall process remains the same.

Each chapter contains various features to explain the purpose of the chapter and help students understand and review the main concepts. The basic features are:

- **What you will learn in this chapter.** A series of questions highlight the important issues.
- **Chapter summary.** A brief synopsis of the chapter highlights—useful when reviewing for exams.
- **Key Words.** A list of words introduced in the chapter. A full glossary is provided at the end of the text.
- **Review Questions.** Designed as a study guide of the main topics in the chapter.
- **Exercises.** Problems that apply the knowledge learned in the chapter. Each set of exercises begins with a list of exercises that mirror the actions taken in that chapter (Book Exercises). Additional exercises using the main cases are next, followed by a set of exercises designed to be solved by a team of students. The team exercises involve work that can be partitioned among team members and combined to achieve a final answer.
- **Additional Reading.** References for more detailed investigation of the topics.

## This Book Is Different from Other Texts

First, this book is a business text designed for an upper division or MBA course. It is not a statistics book and it is not a computer science book. Those types of books are useful and some excellent versions exist, but they are less useful to business students.



Second, this book relies on commonly available tools along with several realistic data sets to illustrate how to analyze business data. Students can use tools that already exist, apply them to the data sets provided, and immediately begin analyzing the results. Many other books require students to obtain tools with limited licenses and restricted capabilities, or to write their own programs. The goal is to get students analyzing business data as quickly and easily as possible, using commonly available tools.

## Instructor Resources

Instructors have access to the following resources for course presentation and management. All the instructor supplements were created by the author, except the test bank:

- Instructor's Manual includes answers to all end-of-chapter review questions, exercises, and teaching notes for the industry-specific cases. Teaching tips and ties to the PowerPoint slides are included for each chapter.
- A test bank contains true/false, multiple choice, and short answer questions.
- Lecture notes are available as slide shows in Microsoft PowerPoint format. The slides contain all of the figures along with additional notes. The slides are organized into lectures and can be rearranged to suit individual preferences.
- Several databases and exercises are available online. The instructor can add new data, modify the exercises, or use them to expand on the discussion in the text. Most of the primary databases were created with custom data generators. These data generators are available to instructors (typically within Microsoft Access databases) so that new data with new patterns can be generated.
- The book's Web site at JerryPost.com provides resources for instructors and students using the text.

## Installing the Sample Databases

---

The book uses several databases to demonstrate how to set up various tools and how to interpret results. These databases are used throughout the book and as exercises in each chapter. The individual databases are described in Chapter 1. Some of the databases were chosen to emphasize specific data mining topics. The Rolling Thunder Bicycle company database is used more than the others because it contains a wider variety of data. The data will be easier to understand if you learn a little about bicycles. Some background information is on the Web site: <http://www.JerryPost.com>.

In every case, you should approach the situation as if you were a manager or consultant. If necessary, visit a similar organization, talk to a manager or employee in the field, read annual reports for a firm in the industry, or find additional background information on the Web. It is easier to understand and apply the results when you understand the goals and decisions facing the organization.

## Software

The majority of the book relies on Microsoft SQL Server and its associated SQL Server Analysis Services (SSAS). When SSAS tools depart from traditional applications, the book also uses traditional techniques to show the difference. The tools chosen, including SQL Server, were specifically selected because they are readily available to students and they are relatively easy to use.

### *Microsoft SQL Server*

SQL Server can be obtained for educational use through the MSDN Academic Alliance, now known as DreamSpark. This organization does require a small fee (\$500 first year, \$320 a year for renewal), but it provides a license for all of Microsoft's development tools, including the right to provide copies to students. Many schools are already members, but additional details can be found at: <https://www.dreamspark.com>.

If you are working alone or unable to join MSDN-AA, a trial version of SQL Server can be downloaded free from Microsoft. It is valid for 180 days. Note that the SQL Server Express version will not work. You need the Standard, Enterprise, or Developer version. The Standard version does not support all of the Analysis Services options, but only a couple of them are used in this book. The examples in this book were built with the SQL Server 2012 Developer's Edition, which is equivalent to the Enterprise edition.

Installing SQL Server on a single computer is relatively straightforward. Just be sure to install the Analysis Services. Also, it is useful to specify Mixed Mode security—which sets SQL Server to accept logins using both Windows accounts and internal SQL Server accounts. When installing Reporting Services and Analysis Services, be sure to specify at least one administrator account that you control. The best approach is to install everything onto a single computer and give yourself administrator rights to each tool. This approach reduces the learning hurdles by minimizing interference from the security systems.

In a class environment, it is possible to install a single copy of the SQL Server database engine, Reporting Services, and Analysis Services on a central server. Individual student computers can then run just the Visual Studio and SQL Server Management Studio components and attach directly to the central server. However, it is necessary to configure security on the server to give relatively open access to each student. Do not use a production server! For the purposes of learning the tools, it is generally simpler to install all of the components on each student's computer and giving the student full control over that instance.

The first time you create an Analysis Services project usually requires some additional configuration steps. Some of the steps require experimentation. The most difficult one is the connection from the Analysis Services system to the database (impersonation). The tool provides four choices. You might have to test all four choices before finding one that works for your installation. The examples in the book suggest one choice that often works, but it does not work in all cases.

### *Weka*

Weka is an interesting open source tool available from the University of Waikato in New Zealand at <http://www.cs.waikato.ac.nz/ml/weka/>. It is available for several operating systems and uses the GNU general public license. It supports several standard data mining algorithms for classification, clustering, association, and attribute selection. It also has some useful graphics tools for visualization. It can read some specialized data files, but is easiest to use with flat CSV files. Installation is straightforward—simply download and run the installation package from the Web site.

### *gretl*

Data mining is heavily based on statistics and econometrics has developed several tools to analyze data. The details of many of the tools and the problems they

solve are beyond the scope of this book. However, many of the tools are relatively automated and can be used to handle specialized problems. High-end (read “expensive”) statistical packages have been used for these tasks. However, Allin Cottrell and Riccardo Lucchetti have created the open source tool gretl that is both free and easy to use. It is available at <http://gretl.soruceforge.net>. Tasks involving regression and time series analysis are particularly useful in gretl. Again, the tool is easiest to use with flat CSV files. The process of extracting the data from SQL Server and creating a CSV file is explained in this book when the tool is used.

## R

R or R System is not used in this book. However, it is a powerful open source statistics package that many other people are using for data mining. R is available free from <http://www.r-project.org>. The GUI interface Rattle is also useful and it is available from <http://rattle.togaware.com>. R is often run on the Linux platform, but versions exist for Windows. The interesting aspect of R is that it is designed for people to create new tools and algorithms. By itself, it is largely a platform. Developers create new tools that can be added to the system to analyze data. This approach makes it easy for others to contribute to the development of new tools, so R can grow. But, it makes it harder for new students to select the proper tool.

## Database Installation

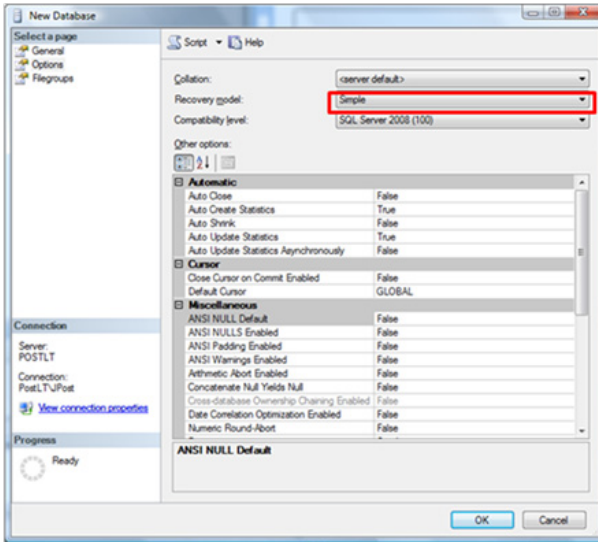
The data for the databases and set up scripts are available for download from the book’s Web site. Each database is packed into a compressed ZIP file. Download each file and extract its contents into a separate folder. The contents consist of several CSV files and a setup SQL script. The script creates the individual tables in SQL Server and bulk loads the data from the CSV files. However, you have to make a couple of changes to the script to tell it where you stored the files. The process is described here for a single database. Repeat it for each database you wish to use.

Start SQL Server Management Studio. Log in with an account that has at least Create Database permissions—generally you want to use a DBA account, or the Windows account that was used to install SQL Server. Create a new database—right-click the Databases entry and choose New Database. Enter a name that represents the database, such as RT for Rolling Thunder. If you are building the database on a central server, you might want to change the file locations to improve performance, but it is not critical.

A more important change is to set the Recovery Mode. The default mode in SQL Server is designed for transaction databases where all data changes are logged to protect the data in the event of failures. This approach creates log files that can be several times larger than the database table files. As with most data warehouses, you can always reload the data, so recovery is unimportant. As shown in the figure, click the Options page and change the Recovery mode to Simple (instead of Full). This mode does not write any data to the log file and saves a considerable amount of space—particularly for the Bakery database because it has several million rows of data.

After the database has been created, you can load the data. Create a new query. The easiest method is to right-click the new database and choose New Query. Load the SQL script from the download into the query window. One quick method is to use the Windows Explorer to navigate to the downloaded files and drag the SQL file into the query window. You need to change two things in the script that tailor





it to your situation: <DB>, the name of the database, and <PATH>, the location of the CSV files.

Use Ctrl-H (or Edit/Find and Replace/Quick Replace) to start the replacement tool. Enter <DB> as the search text—including the brackets. Enter the name of your database (such as RT or Bakery). Click the Replace All button. Repeat the process and search for <PATH> with the brackets, and enter the full local path name of the folder that holds the CSV

files (such as C:\Downloads\Bakery\). You can usually copy the path name from Windows Explorer, but you need to add the trailing back slash (\). Click the button to replace all instances. Note that the CSV files should be on the same computer as the SQL Server program. If you are using a central server, you should copy the CSV folder to that physical server. Otherwise, you need to specify a network path and assign appropriate security permissions. When <DB> and <PATH> have been replaced, execute the script. After a few minutes, you will see a list of the tables and the number of rows. If you get error messages, read them to check for obvious problems. Then reload the script, and run the two replacement commands more carefully. The scripts have been tested on many different machines. Problems that arise are usually due to errors in the replacement path or to local security issues.

# Brief Contents

---

1 Introduction

## **Part One: Fundamental Tools**

2 Database Management Systems

3 OLAP Cubes

4 Probability and Statistics Summary

## **PART TWO: BUSINESS ANALYSIS**

5 Cluster Analysis

6 Association and Market Baskets

7 Evaluation of Dimensions

8 Time Series Analysis

## **Part Three: Specialized Tools**

9 Sequence and Geographic Analysis

10 Multidimensional Expressions (MDX)

Business Applications of Data Mining, v  
Learning Assessment, v

About the Book, vi

*Organization, vi*

*Features That Focus on Solving Problems, vii*

*This Book Is Different from Other Texts, vii*

*Instructor Resources, viii*

Installing the Sample Databases, viii

*Software, viii*

*Database Installation, x*

## **Introduction, 1**

Introduction, 3

Business Situation, 4

*Finance, Risk, and Costs, 4*

*Marketing, 6*

*Production and Supply Chain Management, 11*

*Human Resources Management, 12*

Perspectives, 14

*Probability and Statistics, 15*

*Machine Learning, 16*

*Computer Science: Challenges of Large Data Sets, 16*

*Management Applications, 16*

Database, 17

*Traditional Transactions Processing, 18*

*Data Warehouse and Analytical Processing, 19*

*Data Sources, 19*

*Data Extraction, Transformation, and Loading, 19*

Software Tools, 20

*Data Mining Techniques, 20*

*Data Mining Tools, 21*

*Statistical Tools, 22*

*Production Systems and Scaling, 22*

Potential Dangers, 24

*Human Errors, 24*

*Insufficient Data, 25*

*Bad Data, 25*

*Over Fitting, 26*

*Random Chance, 27*

*Estimation Instability, 28*

*Model Instability, 29*

Introduction to Cases, 30

*Rolling Thunder Bicycle Company, 30*

*Diner, 31*

*Corner Med, 31*

*Basketball, 32*

*Bakery, 32*

*Cars, 32*

Summary, 33

Key Words, 34

Review Questions, 34

Exercises, 34

Additional Reading, 37

## **Fundamental Tools, 39**

### **Database Management Systems, 40**

Introduction, 42

Relational Databases, 43

*Tables, 43*

*Data Types, 45*

Four Questions to Retrieve Data, 47

*What Output Do You Want to See?, 48*

*What Do You Already Know?, 48*

*What Tables Are Involved?, 48*

*How Are the Tables Joined?, 48*

Query Basics, 49

*Single Tables, 50*

*Introduction to SQL, 52*

*Sorting the Output, 52*

*Criteria, 53*

*Useful WHERE Clauses, 56*

Computations, 57

*Basic Arithmetic Operators, 57*

*Aggregation, 58*

*Functions, 60*

Subtotals and GROUP BY, 61

*Conditions on Totals (HAVING), 63*

*WHERE versus HAVING, 64*

*The Best and the Worst, 65*

Multiple Tables, 66

*Joining Tables, 66*

*Identifying Columns in Different Tables, 67*

*Joining Many Tables, 68*

*Views: Saved Queries, 69*

*LEFT JOIN, 71*

*UNION, 71*



- Data Manipulation, 72
  - UPDATE*, 73
  - INSERT*, 73
  - DELETE*, 74
- SQL Server Reports, 75
  - Administration Configuration*, 75
  - Creating a Report*, 76
- Database Design Concepts, 82
  - Notation*, 84
  - First Normal Form*, 86
  - Second Normal Form*, 86
  - Third Normal Form*, 87
- Summary, 88
- Key Words, 89
- Review Questions, 90
- Exercises, 90
- Additional Reading, 93
- OLAP Cubes, 94**
- Introduction, 96
- Challenges with the Relational Model, 98
  - Indexes*, 99
  - Data Warehouse*, 99
  - Extraction, Transformation, and Loading*, 100
  - MOLAP, ROLAP, and HOLAP*, 101
- OLAP Design, 102
  - Facts and Dimensions*, 104
  - Star Design*, 105
  - Snowflake Design*, 107
  - Hierarchies*, 107
- Creating a Cube with Microsoft Analysis Services, 108
  - Data Sources*, 109
  - Data Source Views*, 111
  - Cubes*, 116
- Dimensions, 120
  - Hierarchies*, 123
  - Time Dimensions*, 124
  - Custom Geographic Hierarchy*, 128
  - Attribute Relationships*, 130
- Fine Tuning the Cube, 133
  - Calculations and Queries*, 134
  - Perspectives*, 138
  - Internationalization and Translations*, 140
  - Performance: Partitions and Aggregations*, 142
- Excel PivotTables, 144
- Actions, 146
- Key Performance Indicators, 149
  - Definition*, 149
  - Creating KPIs*, 150
  - Browsing a KPI*, 153
- Summary, 154
- Key Words, 155
- Review Questions, 155
- Exercises, 156
- Additional Reading, 158
- Probability and Statistics Summary, 159**
- Introduction, 161
- Probability Basics, 161
  - Discrete and Continuous Data*, 163
  - Counting and Combinations*, 163
  - Probability Rules*, 166
- Interdependencies: Joint Probabilities, 169
  - Contingency Tables*, 170
  - Tree Diagrams*, 171
  - Bayes Theorem*, 172
- Probability Distributions, 176
  - Discrete Data*, 176
  - Continuous Data*, 181
  - Joint and Conditional Probabilities*, 182
  - Expected Value (Mean) and Variance*, 183
  - Important Continuous Distributions*, 187
- Statistics, 195
  - Samples*, 195
  - Common Statistics*, 196
  - Confidence Intervals*, 198
  - Hypothesis Testing*, 200
  - Chi-Square Hypothesis Tests*, 203
  - Information Measure*, 205
- Summary, 206
- Key Words, 208
- Review Questions, 208
- Exercises, 209
- Additional Reading, 214
- Business Analysis, 215**
- Cluster Analysis, 216**
- Introduction, 218
- Business Situation, 220

- Model, 221
  - Distance or Dissimilarities*, 222
  - Combinatorial Searches with K-Means*, 224
  - Statistical Mixture Model with EM*, 227
  - Hierarchical Clusters*, 229
  - Other Statistical Methods*, 233
- Data, 236
  - Attributes and Observations*, 236
  - Continuous and Discrete Data*, 237
  - Missing Data*, 238
- Clustering on Products: Cars, 238
  - Goals*, 238
  - Data*, 239
  - Microsoft Clustering*, 241
  - Results from Microsoft Clustering*, 243
  - Prediction*, 246
  - Larger Model and Parameter Changes*, 248
- Traditional EM Clustering, 251
  - Goals and Data*, 252
  - Results*, 254
  - K-Means Clusters*, 255
- Comparison, 256
- Customer Clustering with Categorical Data, 258
  - Data*, 258
  - Microsoft Clustering Results*, 259
  - Weka Clustering Results*, 260
- Summary, 262
- Key Words, 263
- Review Questions, 263
- Exercises, 264
- Additional Reading, 267
- Association and Market Baskets, 268**
  - Introduction, 270
  - Business Situation, 271
    - The Bakery*, 272
    - Product and Dimension Levels*, 273
  - Model, 274
    - Goal*, 275
    - Assigning Values to Rules*, 276
  - Problems with Dimensions, 282
    - The A Priori Algorithm*, 283
    - Issues in Setting Minimum Support and Confidence*, 284
  - Potential Problems, 285
    - Simpson's Paradox*, 285
    - Skewed Support Data*, 286
    - Continuous Data*, 288
    - Quantity*, 289
  - Data, 290
    - Database Structure*, 291
    - Market Basket Structure*, 291
  - Traditional Tools for Association Rules, 292
    - Goals*, 293
    - Data*, 293
    - Results*, 293
  - Microsoft Association Rules, 294
    - Goals*, 295
    - Data*, 295
    - Results*, 296
    - Comparing Results*, 298
  - Summary, 300
  - Key Words, 301
  - Review Questions, 301
  - Exercises, 302
  - Additional Reading, 305
- Evaluation of Dimensions, 306**
  - Introduction, 308
  - Business Situation, 309
  - Model, 309
  - Data, 311
    - Attributes and Observations*, 312
    - Continuous and Discrete Data*, 312
    - Missing Data*, 313
  - Linear Regression, 313
    - Goals*, 314
    - Data*, 316
    - Tools*, 318
    - Results*, 322
    - Attribute Evaluation*, 327
    - Prediction*, 328
  - Logistic Regression, 330
    - Goals*, 330
    - Data*, 332
    - Tools*, 333
    - Results*, 334
    - Attribute Evaluation*, 338
    - Prediction*, 339
  - Naïve Bayes, 340
    - Goals*, 341
    - Data*, 345

- Tools, 345*
- Results, 346*
- Attribute Evaluation, 346*
- Prediction, 348*
- Decision Trees, 349
  - Goals, 350*
  - Data, 352*
  - Tools, 353*
  - Results, 353*
  - Attribute Evaluation, 355*
  - Prediction, 355*
- Neural Network, 358
  - Goals, 359*
  - Data, 361*
  - Tools, 361*
  - Results, 361*
  - Attribute Evaluation, 362*
  - Prediction, 363*
- Model Comparisons, 365
  - Prediction, 366*
  - Attribute Evaluation, 367*
  - Nonlinear Complications, 368*
- Summary, 369
- Key Words, 370
- Review Questions, 370
- Exercises, 371
- Additional Reading, 373
- Time Series Analysis, 375**
  - Introduction, 377
  - Business Situation, 378
  - Model, 379
    - Time Series Components, 379*
    - Auto Regression, 381*
    - Moving Average, 384*
    - Trends, 387*
    - ARIMA, 389*
    - Cross Correlations, 392*
    - Evaluating Models, 395*
  - Data, 396
    - Attributes and Observations, 396*
    - Missing Data, 397*
  - Traditional ARIMA Estimation, 397
    - Goals, 398*
    - Tools, 398*
    - Results, 400*
  - Forecasts, 404*
  - Seasonality Evaluation, 405*
- Microsoft Time Series Estimation, 406
  - Goals, 406*
  - Data, 408*
  - Tools, 409*
  - Results, 412*
  - ARTxp Model, 413*
  - Forecasts, 415*
  - Seasonality Evaluation, 417*
- Cross Correlation and Linear Regression, 417
  - Goals, 418*
  - Data, 418*
  - Tools, 419*
- Comparison, 425
- Summary, 426
- Key Words, 427
- Review Questions, 428
- Exercises, 428
- Additional Reading, 431
- Specialized Tools, 432**
- Sequence Analysis and GIS, 433**
  - Introduction, 435
  - Sequence Analysis , 435
  - Business Situation, 437
  - Model, 438
    - Classification, 439*
    - Clustering: Business examples, 440*
  - Data, 442
    - Attributes and Observations, 442*
    - Continuous and Discrete Data, 444*
    - Missing Data, 444*
    - Web Log Data Files, 445*
  - SQL Server Sequence Clustering, 447
    - Goals, 448*
    - Data, 448*
    - Tools, 449*
    - Results, 452*
    - Prediction, 457*
  - Other Tools, 458
  - Sequence Summary, 458
  - Geographic Analysis, 459
  - Business Situation, 461
  - Data, 461

- Model, 462
- Microsoft MapPoint, 464
- Other Tools, 467
  - Esri*, 468
  - Google*, 471
  - Bing*, 472
  - Federal Government*, 473
- Geographic Summary, 474
- Key Words, 474
- Review Questions, 475
- Exercises, 476
- Additional Reading, 479
- MDX, 480**
  - Introduction, 482
  - OLAP Cube Review, 482
    - Dimensions and Hierarchies*, 484
    - Rolling Thunder Bicycles Cube*, 484
  - Definitions and Concepts, 487
  - Main Syntax, 489
  - Basic Examples, 490
    - A First Example*, 491
    - Adding a WHERE Condition*, 492
    - Displaying Specific Dimension Values*, 494
    - Cross Join*, 494
  - Calculated Measures, 495
  - Complex Computations, 496
    - Percentages*, 497
    - Compute Changes*, 498
    - ParallelPeriod Function*, 499
  - Some MDX Functions, 501
    - EXCEPT: Taking Values Out of Totals*, 501
    - Conditions with IIF and CoalesceEmpty*, 502
    - TopCount Function*, 504
    - Year to Date*, 505
    - Moving Averages*, 507
  - Summary, 509
  - Key Words, 510
  - Review Questions, 511
  - Exercises, 512
  - Additional Reading, 515

---

# Introduction

## Chapter Outline

Introduction, 3	<i>Statistical Tools, 22</i>
Business Situation, 4	<i>Production Systems and Scaling, 22</i>
<i>Finance, Risk, and Costs, 4</i>	Potential Dangers, 24
<i>Marketing, 6</i>	<i>Human Errors , 24</i>
<i>Production and Supply Chain Management, 11</i>	<i>Insufficient Data, 25</i>
<i>Human Resources Management, 12</i>	<i>Bad Data , 25</i>
Perspectives, 14	<i>Over Fitting, 26</i>
<i>Probability and Statistics, 15</i>	<i>Random Chance, 27</i>
<i>Machine Learning, 16</i>	<i>Estimation Instability, 28</i>
<i>Computer Science: Challenges of Large Data Sets, 16</i>	<i>Model Instability, 29</i>
<i>Management Applications, 16</i>	Introduction to Cases, 30
Database, 17	<i>Rolling Thunder Bicycle Company, 30</i>
<i>Traditional Transactions Processing, 18</i>	<i>Diner, 31</i>
<i>Data Warehouse and Analytical Processing, 19</i>	<i>Corner Med, 31</i>
<i>Data Sources, 19</i>	<i>Basketball, 32</i>
<i>Data Extraction, Transformation, and Loading, 19</i>	<i>Bakery, 32</i>
Software Tools, 20	<i>Cars, 32</i>
<i>Data Mining Techniques, 20</i>	Summary, 33
<i>Data Mining Tools, 21</i>	Key Words, 34
	Review Questions, 34
	Exercises, 34
	Additional Reading, 37

## What You Will Learn in This Chapter

- What is data mining and what do managers need to know about it?
- How is data mining used in business?
- What do managers need to know about data mining?
- What statistical and data mining tools are used in this book?
- What can go wrong?
- What are the cases used in the book?



### Wal-Mart

The amount of digital information increases ten times every five years, and Cisco estimates that in the year 2013, 667 exabytes of data will travel over the Internet. [Economist 2010]

In 2004, Wal-Mart had 460 terabytes of sales and customer data stored at its Bentonville headquarters. Wal-Mart's CIO Linda M. Dillman challenged her staff to help Wal-Mart's Florida stores before the onset of Hurricane Frances. Specifically, what products would residents want to buy before the storm hit? The analysts turned to the historical sales data for stores in the paths of prior hurricanes and applied some early analytical tools. While some might expect that survival items such as flashlights would be popular, it turns out the two biggest-selling items before a hurricane are: strawberry Pop-Tarts and beer. Ms. Dillman noted that "We didn't know in the past that strawberry Pop-Tarts increase in sales, like seven times their normal sales rate, ahead of a hurricane. And the pre-hurricane top-selling item was beer." So, the company loaded the trucks and shipped beer and Pop-Tarts to the stores in the path of the new storm—quickly selling most of the products stocked for the storm. [Hays 2004] By 2011, Wal-Mart's database was estimated to contain 2.5 petabytes of data and the company was expanding its use of business intelligence and data mining tools. [Rogers 2011] Wal-Mart makes some of its sales data directly visible to vendors—leaning towards a pass-through system where vendors monitor sales and build and ship products to match forecasts based on end-point sales.

Even with years of experience, managers and analysts can miss details and changes. Asking questions and analyzing data are critical to making better decisions.

*The Economist*, "Data, Data Everywhere," February 25, 2010. <http://www.economist.com/node/15557443>

Constance L. Hays, "What Wal-Mart Knows About Customers' Habits," *The New York Times*, November 14, 2004. <http://www.nytimes.com/2004/11/14/business/yourmoney/14wal.html>

Shawn Rogers, "Big Data is Scaling BI and Analytics," *Information Management*, September 1, 2011. [http://www.information-management.com/issues/21\\_5/big-data-is-scaling-bi-and-analytics-10021093-1.html](http://www.information-management.com/issues/21_5/big-data-is-scaling-bi-and-analytics-10021093-1.html)

## Introduction

---

**What is data mining and what do managers need to know about it?** **Data mining** is the process of using analytical tools to scan large data sets for patterns and provide insight to analysts and managers. Data mining can also be considered as machine or statistical learning, where data is used to train a system to identify categories and patterns so these can be used to make future decisions. Different names are sometimes used, including business intelligence and analytics. Some people categorize data mining and business intelligence as different tools. Data mining tends to focus on the database and algorithm topics while business intelligence emphasizes the business applications and evaluation of the tool results. This book serves as an introduction to all of the related topics, so it covers the data tools, some useful aspects of the algorithms, and the business applications. The goal of the book is to provide enough background to get the reader started in using the common tools with a basic ability to read and apply the results to business problems.

It is critical to recognize that data mining is different from statistical research. Statistical theory is used in many data mining techniques, but the goals of data mining and statistical research are radically different. The goal of statistical research is to prove (or disprove) hypotheses. Specific processes must be followed when evaluating results statistically in order to apply the fundamental theorems. The goal of data mining is much looser—it is to help analysts and managers explore the data and spot potential patterns. These patterns might be small and below the thresholds required for statistical testing. Applying multiple analyses to the same set of data, searching for small patterns, usually violates key assumptions required in statistical research. Data used for data mining can help analysts and managers identify potential patterns and hypotheses to test later, but the same data should not be used for formal research.

Why are data mining tools necessary? Can managers simply examine the data and find patterns on their own? Yes, some patterns are large enough to see simply by examining the data and basic charts. Chapter 3 shows how **hyper cubes** are used to explore data subtotals and enable managers to browse the data and compare various subgroups. But some patterns are complex and difficult or time-consuming to identify. Statistical data mining tools are much more efficient at examining many factors and finding patterns and important relationships. Most statistical tools provide measures of the strength of relationships, making it possible to focus on key properties in the data. Statistical tools are also useful for **forecasting**—extending existing data to predict potential outcomes as the data change.

As the cost of computers and data storage has declined over the years, the amount of data collected by organizations has dramatically increased. Managers can access terabytes and petabytes of **data** on every aspect of business including production, costs, marketing, and human resources management. But do you know which data is important? Max Hopper, CIO of American Airlines, developed a hierarchy that describes how managers can collect data, use it to create information and develop knowledge, which ultimately should lead to wisdom.

Can the data be converted into **information**, where data provides answers to questions and is used to make decisions? A **database management system (DBMS)** is often used to retrieve data to answer questions. Chapter 2 describes how queries are created to answer business questions. But queries require managers to know which questions to ask. The next step is to use information to develop

**knowledge**, or a deeper understanding of the data including rules and patterns. This level is the main target of data mining—using semi-automated tools to find patterns and rules in the data. How this knowledge is used and applied can lead to **wisdom**—finding new knowledge and making decisions in the presence of a changing environment. Data mining goes beyond basic database concepts and focuses on finding useful patterns in the data, as opposed to simply looking up values. Many of the data mining tools are intended to be interactively used by managers.

## Business Situation

---

**How is data mining used in business?** This book is largely written for business managers and analysts. The sample datasets, applications, and exercises are all based on common business problems. Yet, the examples and discussion cover only a small sampling of the possible business applications. There is almost no limit to the type of business problem that can be handled by data mining tools. As long as sufficient data points are available, it should be possible to find a data mining tool that can help identify patterns and rules. A key aspect of data mining is to understand each of the main tools so you know which tool to apply to specific types of data.

Many companies use data mining in multiple areas of business—some examples are publicly reported, others are carefully guarded secrets. It is helpful to consider a few examples of how data mining can be used in actual companies, so this section describes a few cases where information is publicly available.

Data analysis is critical to business decisions. Without data and critical analysis, people make mistakes. Relying on intuition in business decisions is usually a recipe for disaster. No matter how much experience a person has, it is critical to “do the math” and evaluate all of the data.

### Finance, Risk, and Costs

Finance, particularly investment managers, has made heavy use of analytical tools to identify patterns and rules. Predicting future changes is critical to investments. Some systems use highly complex models to determine how financial and economic variables move together and how patterns might move into the future. Some systems are simpler but are completely automated. They monitor millions of transactions around the world, looking for small differentials in markets and making trades faster than humans. Other issues are more complex—because they involve people. Predicting how people will perform in the future is one of the classic questions in finance. For example, determining how much money to lend to people and what interest rate to charge are key questions for bank loans, mortgages, and credit cards.

Investments are tricky and you can find some of the reasons in the theory of finance and economics. But, to illustrate a specific problem related to data mining, consider a classic scam. Do not attempt the scam; just think about it for a minute. A con-artist chooses a stock. He sends e-mails to 1,000 people stating that he can predict the future and that the stock is going to increase in price. He sends messages to another 1,000 people that the stock is going to drop in price. The next week, if the stock has increased in price, he sends a new message to 500 of the first group with a message that a second stock will increase in price and the reverse message to the second half of the group. If the stock price had fallen, he would switch to the second group and do the same thing. After three weeks, the

swindler has a group of 125 people convinced that he can accurately predict stock prices. Who else can predict three events in a row? At that point, the swindler asks the people for money. Does the story seem absurd? What does it have to do with data mining? On a broader scale, the same problem exists if you examine stock-picking data for brokers or mutual funds. Given a large enough sample, some company or person is likely to have a successful string of selections simply by random chance. Out of the millions of investors and thousands of firms, this one lucky firm stands out and receives all of the attention from the press. Does this firm or person truly have the “formula for success,” or is it merely chance? The answer to the relationship with data mining is that with enough data, a data mining approach will eventually find that one firm or one person who has been successful more times than anyone else. But, did the data mining find a useful set of rules or merely the result of chance applied to the huge amount of data?

### *Washington Mutual*

Washington Mutual, or WaMu as it called itself, was a large savings and loan bank with hundreds of branches on the West coast. The bank largely focused on lower-income customers with an emphasis on free checking, low fees, and liberal lending policies. As the real estate market heated up from 2003-2006, WaMu was one of the leaders in providing mortgages to thousands of borrowers. As Goodman and Morgenson (2008) reported, one interesting loan involved a mariachi singer claiming a six-figure income. John D. Parsons, a WaMu supervisor could not verify the income so he asked for a photograph of the singer in his mariachi costume standing in front of his house—and then approved the loan. With the “Power of Yes” campaign, WaMu’s home-lending unit generated almost \$2 billion in revenue in 2003. In 2005, WaMu purchased Provident, one of the leaders in selling credit cards in the subprime market. Ultimately, the goal of the combined company was to obtain higher interest rates on riskier loans. Like many banks, the company used financial data on customers to identify potentially risky loans; but unlike many banks, Provident and WaMu managers believed they had found the “best of the bad,” (Koudsi 2002). Loan applications were entered into the bank’s computer system for initial approval. Loans not approved automatically were overridden by humans. Many of the loans were repackaged and sold to investors. Yet, midway through 2008, the value of WaMu’s bad loans had reached \$11.5 billion—triple the level from the year before (Goodman 2008). In September 2008, WaMu was sold to JPMorgan Chase for the fire sale amount of \$1.9 billion.

The WaMu situation was not unique. The resulting crash in the housing industry drove the U.S. into the start of the Great Recession as it was called. From a data mining perspective, the key elements are that (1) the financial data mining models used by the company did not adequately estimate the risks involved, and (2) the models were apparently ignored and overruled by human managers in many cases. Other financial institutions also struggled, but many survived by having better models and better understanding of the data and patterns.

### *High-Frequency Trading*

Automated trading systems have existed for several years. Over time, the systems have become more sophisticated and faster. The early systems concentrated on arbitrage—identifying risk-free transactions to take advantage of small differences

in price. For example, if the price of a currency in the Tokyo market differs slightly from its price in the London market, a computer system can quickly identify the difference, and buy a currency on one market, resell it on the other and gain the difference as a profit. As the markets themselves moved to automated processing, several trading firms wrote data analysis algorithms to monitor trades and spot patterns instantly. The tools rely on the ability to monitor data in real time and submit trades before anyone else can react. In most cases, these trading systems are hosted on high-speed networks in the same building as the trade computers. According to (Duhigg 2009), stock exchanges suggest that by 2009, a handful of these trading systems accounted for more than half of all trades. The systems monitor all trades looking for patterns—particularly rising demand for a specific stock. Within milliseconds, the computer systems buy up shares of the desired stock and turn around and resell it to slower-ordering humans, making a profit on the difference as the price rises. Some of the systems issue and cancel small orders just to gauge the extent of the market to determine the maximum price slow traders are willing to pay. Even if the difference is only pennies a share, multiply the difference by thousands of shares across multiple stocks every day, and the profits multiply—perhaps to as much as \$21 billion in 2008 (Duhigg 2009 citing Tabb Group research).

Autonomous systems that can analyze data automatically are relatively rare, but they can be important. As the data mining tools improve, more autonomous systems will be added to financial analyses and other business problems.

## Marketing

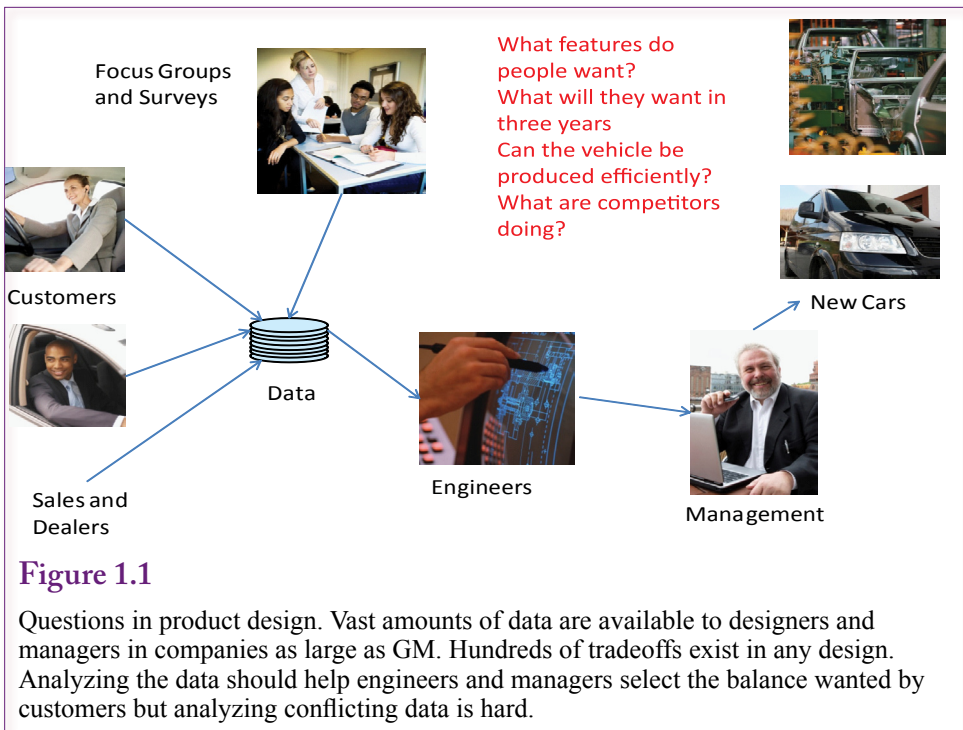
Many data mining tools are useful in marketing decisions. Predicting consumer demand is a key element of almost any organization. Designing, building, and shipping products often require time, and companies need some idea of the demand before production begins. But forecasting exactly how customers are going to behave is difficult. Any pattern or forecast can give an edge to the company that understands the data and implications.

Data mining is also used to assist customers and cross sell products. The recommendation tools used by bookstores, music systems, and movie rental companies such as Netflix are useful for encouraging customers to purchase related items. These systems rely on data mining to identify similar products that might appeal to each customer.

### *General Motors*

The issues of determining customer demand are critical in the automobile industry. It takes about three years to design and develop a new automobile. It takes several days and advance planning to physically produce a car. The cost of a car is driven down through mass production—making thousands of almost identical cars at the same time. Minor changes can be customized as a car is built—interior, the choice of radios, tires and so on are relatively easy to make as the car is built. Major changes—particularly colors are difficult to customize and retain low costs. Consequently, cars are designed and built long before they are sold. Many economic changes can occur before the cars make it to dealer showrooms. Selecting design features and forecasting demand for each model is critical in this environment. Figure 1.1 shows some of the types of data available and the basic questions faced by designers and managers. Designing complex products is difficult because of the hundreds of tradeoffs. Yes, customers want powerful engines, but they also





want high gas mileage and low prices. Obviously, one car does not fit everyone. But exactly what features should be included on each type of car and how will the collection of attributes be perceived?

A precursor of GM's design evolved in the early 1980s. The 1970s were hard on GM—a huge increase in the price of gas resulted in a dramatic drop in sales of their over-powered gas-guzzling cars. GM's response was to build an entirely new set of cars for the 1980s. Interestingly, all four major divisions at that time (Chevrolet, Pontiac, Buick, and Oldsmobile) designed almost exactly the same car. Why did each division build the same car when different people want different vehicles? Jump forward to the early 2000s and almost the same problem emerged—each division producing identical vehicles. Throw in the gas price increases of 2007 and GM's gas guzzling trucks and it starts to become obvious why GM was forced into bankruptcy and bailouts by the U.S. and Canadian governments in 2009. GM's designers have tons of data and computing power. How is it that GM was unable to identify customer needs when Toyota and Honda were so much better? See (Taylor 2008) for more details. As a result of the initial questions raised in the 1980s, (Barabba and Zaltman 1991) examined issues in collecting data and evaluating data. The chapter on human biases is particularly useful. Just collecting data is not enough. Even if good data mining tools are used, managers must still understand and believe the results. The GM case clearly indicates the importance of analyzing data, forecasting demand, and overcoming personal biases when designing products.

### *Pfizer*

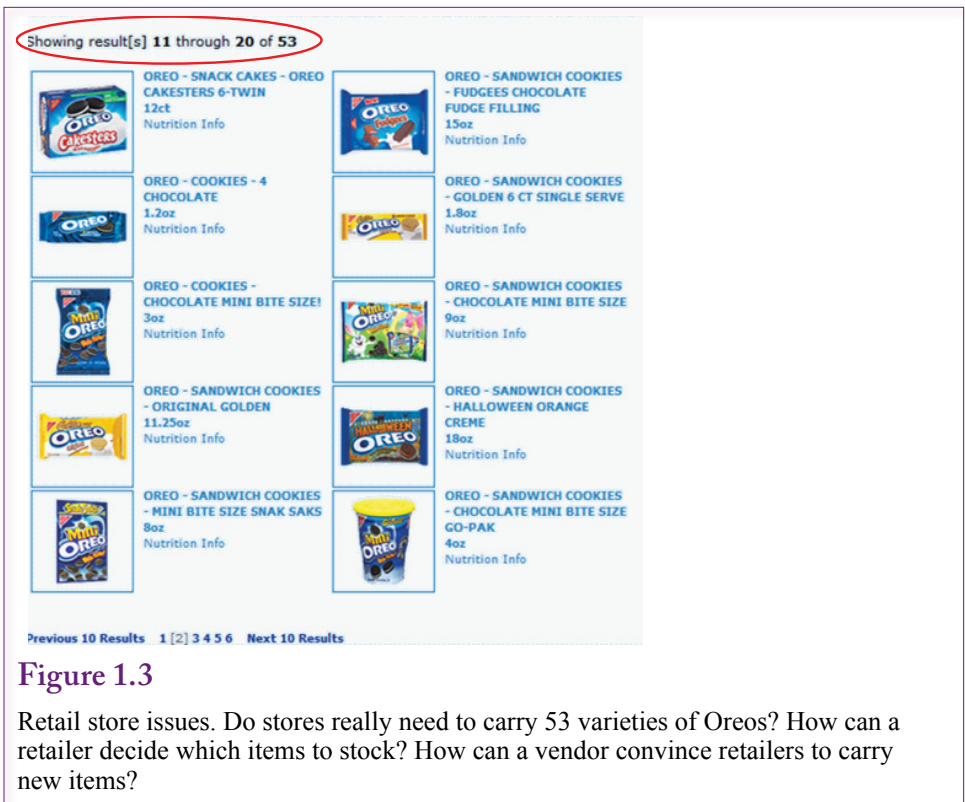
Another interesting marketing design case involves Pfizer, one of the largest pharmaceutical companies in the world. See (Johnson 2007) for details about how



Pfizer wrote off \$2.8 billion in a project to create an insulin inhaler. Pfizer spent 11 years trying to develop an inhaler that could be used by diabetics to put insulin into the blood stream instead of using injections. After passing all medical tests, the project made it to the market and Pfizer was pushing the product through physicians who specialized in the treatment of diabetes. After lackluster sales of a few million—compared to the projected billions—of dollars, Pfizer pulled the plug on the new product in 2007. In the end, it appears that patients rejected the device because of potential complications with inhaled insulin, because it cost twice as much as injections, and because the device resembled a bong for smoking marijuana. Also, over the course of the 11-year development, insulin pens became available that are substantially easier to use than old-style syringes. As noted in Figure 1.2, pharmaceutical companies often focus on clinical drug trials to generate data and analyses for the Food and Drug Administration. These trials rely on research methods to prove the efficacy and safety of potential new treatments. But, increasingly, pharmaceutical manufacturers such as Pfizer also need to consider the business aspects of treatments, including the ability to manufacture quality products at low prices and to produce items that customers and physicians will prefer over existing treatments. The point is that marketers and managers did not correctly identify the attributes that were important to customers. Identifying attributes and relationships are key aspects in data mining.

### *Retail Stores*

Think about some of the critical decisions for retailers. One of the most challenging problems is identifying which products to stock on shelves and the quantity to carry of each item. As a retailer, it is difficult or impossible to sell products you do not have. Running out of stock or not carrying a popular product is going to result in customers going to other stores and not coming back. Carrying too many



**Figure 1.3**


Retail store issues. Do stores really need to carry 53 varieties of Oreos? How can a retailer decide which items to stock? How can a vendor convince retailers to carry new items?

products costs shelf space and stocking seldom-sold items takes up space on the shelf and warehouse that could be used for popular products. The challenge lies in identifying exactly which products fall into each category, and predicting the demand for thousands of varieties. For many large retailers, the decision for years was to lean towards multiple products. Figure 1.3 from Nabisco's Web site shows an extreme case: 53 varieties of Oreo cookies. By 2008, a typical food retailer carried 47,000 different products and 47,113 new product variations or sizes were introduced in 2008. By 2009 (Brat et al. 2009), supermarkets, pharmacies, and other retailers began culling their product lists. For example, Walgreen decided that 25 versions of superglue was overkill and reduced the list to 11. Wal-Mart reduced the number of tape measures from 24 down to 20.

Huge amounts of data exist on retail sales. Bar-code scanners and loyalty cards provide data on every transaction by basket, time of day, store, and type of customer. But data mining is needed to find associations, trends, and forecast future demands.

### *Netflix*

The rise of e-commerce has increased the interaction with customers. As customers browse products, recommendation systems can evaluate each selection and compare it to products that other customers have selected. Particularly for subjective items such as books, music, and video, customers often enjoy the assistance provided by seeing what other customers purchase. Perhaps the author or artist has a new release, or a related product might often work well with a specific item. Netflix has a library of tens of thousands of movies. As shown in Figure 1.4,



The screenshot shows the Netflix homepage with a red header containing the 'NETFLIX' logo. Below the header, there are two main sections: 'Start Your 1 Month Free Trial' and 'Browse Selection'. The 'Browse Selection' section is divided into a vertical sidebar on the left and a grid of movie posters on the right. The sidebar lists various categories: 'Recently Added', 'Popular Picks' (highlighted in red), 'TV', 'Action', 'Anime', 'Children's Movies', 'Classic Movies', 'Comedies', 'Cult Movies', 'Docs', 'Dramas', 'Faith and Spirituality', 'Foreign Movies', 'Gay & Lesbian Movies', 'Horror', 'Indie Movies', 'Musicals', and 'Music'. The grid of movie posters includes titles such as 'TARZAN AMERICA', 'WARRIOR', 'THE SCORPION', 'HUNTER', 'THE MINDY PROJECT', 'NATIONAL SECURITY', 'TRANSFORMERS', 'THOR', and 'ALPHAS'.

Data  
Customer rentals.  
Customer ratings.  
Movie ratings/sales.

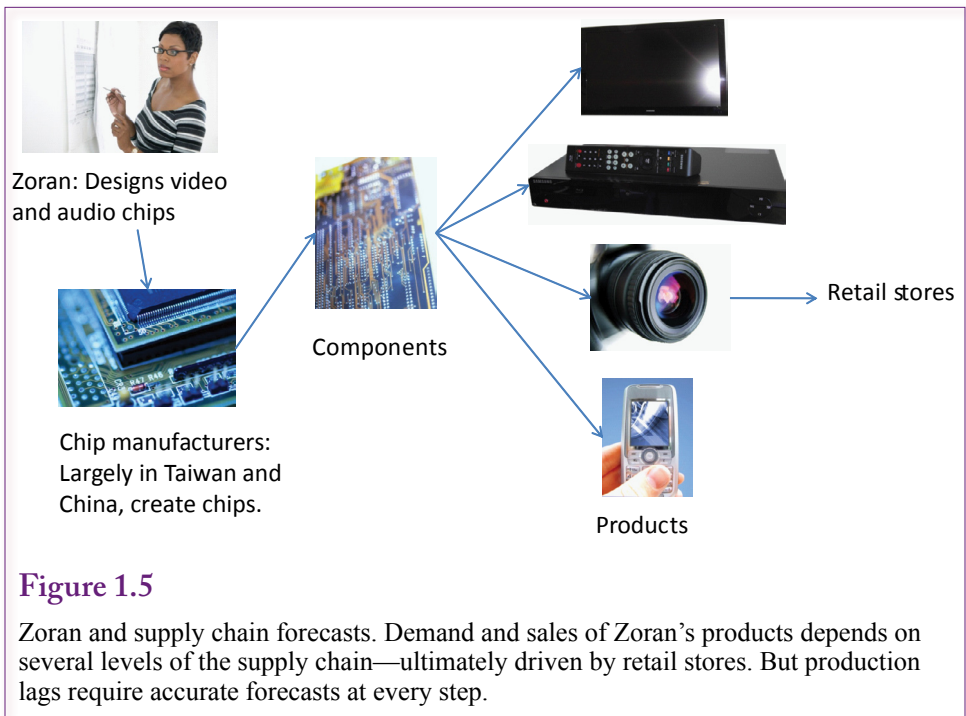
Goal  
Recommend movies that customers will enjoy.  
Convince customers to rent older movies.

**Figure 1.4**

Netflix recommendations. Using customer ratings of existing movies, the system must automatically find similar movies that the customer might enjoy. The ultimate goal is to convince customers to rent movies beyond the current releases.

Netflix needs to convince customers to rent movies beyond the handful of latest releases. The Netflix recommendation system is a critical element in the business model of the company. In 2006, Netflix initiated a \$1 million competition—open to anyone who could create a system that could improve recommendation results by at least 10 percent. Any tools could be used but the resulting tools had to be published and licensed to Netflix. A sample data set scrubbed of personal information was provided to contestant teams and results were measured against an internal set of data using a traditional measure (root-mean-square-error). In early results, individual teams made minor progress. In 2009, one team took the step of combining multiple data mining approaches and averaging the results. Based on public results, it accomplished the 10 percent gain, but other teams then combined their approaches in an attempt to duplicate or surpass the results. See (Lohr 2009) for details.

Other than the potential for winning a million dollars, the interesting aspect of the Netflix competition is that a combination of data mining techniques was the winning strategy. Often, analysts and researchers get fixated on a single approach. Observe that it took almost three years for someone to test multiple approaches. As you learn the different techniques, particularly the evaluation methods in Chapter 7, you might wonder why so many different methods exist. Keep in mind that some problems might require multiple approaches.



## Production and Supply Chain Management

Adam Smith, one of the early economists, in 1776 wrote about the value of specialization and exchange. Individuals and firms could increase production by specializing. Each person (firm or nation) could produce more of a single item or single step in production by focusing on that one item. One challenge with specialization is the cost of coordinating and paying each specialist. By 2000, information technology reduced transaction costs far enough so that the tasks of building complex products was scattered across many companies and multiple nations. Consider the case of Zoran, a company that designs video and audio processing chips that are used to control digital televisions, cameras, and other gadgets. With the 2009 switch to digital television in the U.S., demand for digital products that use Zoran's chips was booming. As shown in Figure 1.5, demand for Zoran's products depends on several levels of the supply chain for digital products. In a few short weeks in 2008, demand for consumer products evaporated to nothing. As the recession loomed, manufacturers predicted declines in sales, so at every stage of the process, vendors and producers slammed the brakes on production. Rick Tsai, CEO of Taiwan Semiconductor Manufacturing (TSMC) noted that consumer purchases of electronic products in the U.S. fell 8 percent in the last quarter of 2008, but shipments of chips dropped 20 percent (Dvorak 2009). The extent of the drop surprised most producers, but it also meant that as product demand increased, demand for chips also increased early. However, David Pederson, vice president of marketing at Zoran noted that forecasting demand for chips was difficult—sometimes sales are redirected to other uses, such as chips purchased for televisions being installed in digital photo frames. In other cases, growth forecasts are difficult because multiple customers of Zoran's chips might be competing for the same end contract, and only one or two might win. Consider the case of DVD players. Best



Buy, one of the largest U.S. retailers places orders each week along with forecasts for future needs. In general, the company orders DVD players about six weeks in advance. But, the components cannot be produced that quickly, so suppliers need to predict demand and build parts in advance of the orders. Profit margins are extremely thin at each step, and no firm wants to get caught with excess inventory. When the financial crisis hit in the U.S. in 2008 and consumers disappeared, Michael Vitelli, Best Buy's merchandizing chief slashed orders, noting that "you actually had to pick a number with no knowledge whatsoever, because nobody knows anything," (Dvorak 2009). Shipments of audio and visual equipment fell 19 percent in November, 21 percent in December, and 58 percent in January. Producers such as Zoran responded by cutting even deeper. Mr. Pederson noted that "everybody under-cut a certain extent." TSMC, the company that manufactures chips for Zoran and others, slashed production to 35 percent of capacity.

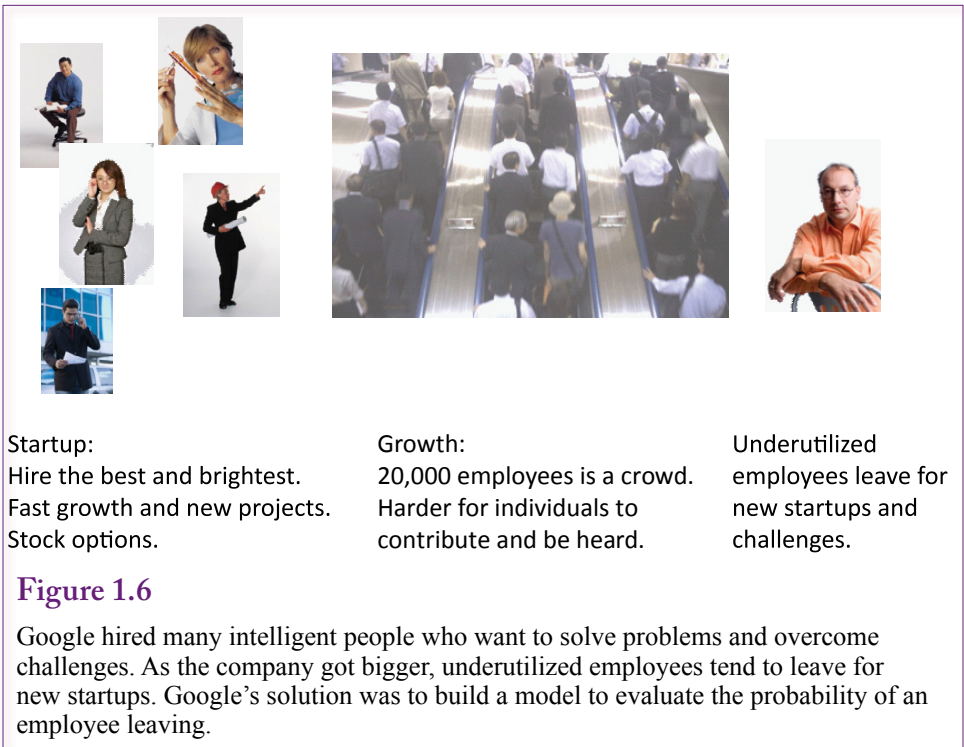
In terms of data mining, the case of Zoran and similar firms presents immense challenges. First, prediction is critical for all levels of production. **Just-in-time** systems require planning and building items in advance of when they will be needed. Second, most models were estimated in fairly stable economic environments. When data shifts far beyond normal ranges, model estimates may no longer work. Third, building models of complex supply chains with multiple interactions is extremely difficult. It might not be possible to obtain reliable data, and constantly changing firms can make it difficult to identify interactions, much less predict them. Yet, extreme changes can often provide useful data for estimating complex reactions. If the only data you ever see comes from the same constant environment, there will never be enough information to estimate radical changes.

## Human Resources Management

Employees are the most important part of many businesses. For service companies, employees interact with customers and hold the knowledge and skill to advance the company. Identifying and rewarding the best employees, supporting communication among employees and improving teamwork are often critical components to any business. In some cases, employees are a major component of costs. Determining efficient schedules requires predicting the number of workers needed at each point in time. In more extreme situations, monitoring employees for patterns of fraud or theft is also important. All of these activities use data mining to evaluate patterns.

### *Google*

At heart, Google is a knowledge company that depends heavily on its 20,000 employees. The company hires creative people and relies on employees to develop new products and new ideas. Most employees are given time to work on experimental ideas and new applications. But hiring and managing 20,000 employees is difficult—some employees are bound to slip through the cracks. As indicated in Figure 1.6, as the company gets bigger, the startup atmosphere and stock option growth rates disappear. In 2008 and 2009, several high-level and mid-level managers left to work on smaller companies such as Facebook and Twitter. Those workers who feel underused, or unable to contribute, have a tendency to leave. Finding new employees with the talent and creativity to help grow the company is difficult, time consuming, and expensive. So Google developed a formula based on employee surveys and pay raises among other data that identifies workers most likely to leave. Laszlo Bock, head of HRM for the company notes that the system

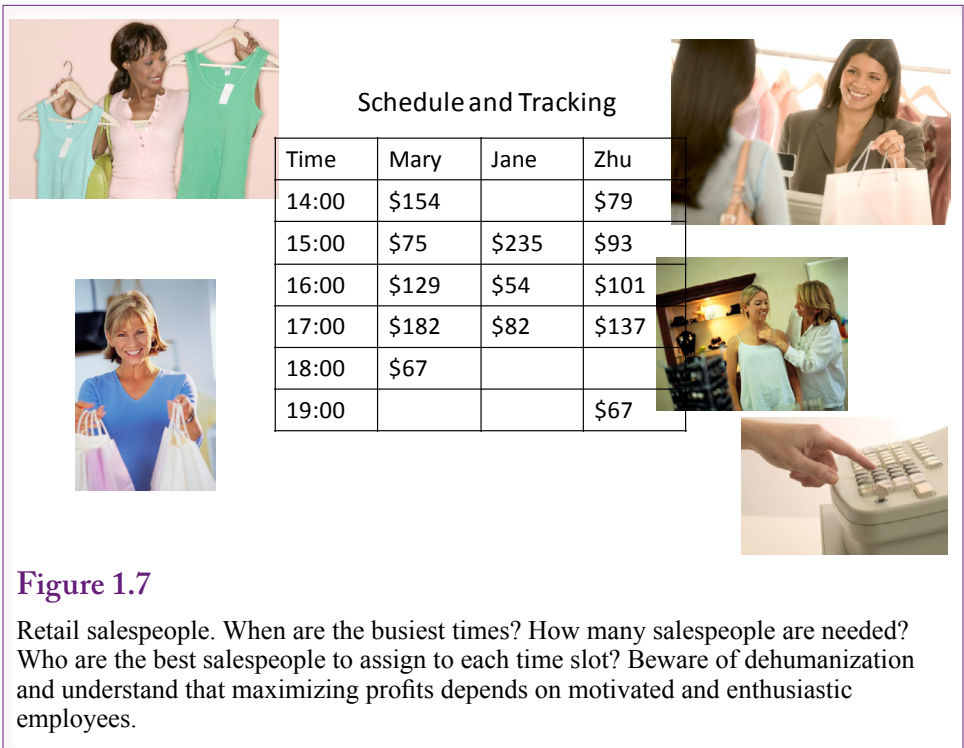


enables the company to “get inside people’s heads even before they know they might leave,” (Morrison 2009).

Applying data mining tools to human capital data is a relatively new experience. Google’s application is ahead of most other companies, and it does face somewhat unique circumstances. Most smaller companies would simply rely on human managers who work with employees on a daily basis to monitor attitudes and find useful tasks that keep workers engaged and creative. There is also the issue of how employees will perceive the use of data mining tools to monitor behavior. If a company does develop a powerful model to predict employee actions, it would probably be wise not to brag to the press.

#### *AnnTaylor Stores Corp.*

More traditional businesses have simpler problems—scheduling employees efficiently. Retail stores are classic examples. Having too few salespeople causes customers to walk away. Having too many workers increases costs and reduces profits. Knowing the proper number depends on predicting the number of customers at each point in time. It also depends on the productivity of each worker. As shown in Figure 1.7, AnnTaylor Stores Corp. installed a workforce management system in 2007 that monitors the performance of salespeople in terms of average sales per hour, dollars per transaction, and units sold. The system forecasts the busiest hours and schedules the most productive employees during those times (O’Connell 2008). Before the system was implemented, most store managers scheduled workers according to their preferences, and staffing rarely matched the peak time loads. The new system encourages workers to increase sales. However, this and similar systems often have the effect of shuffling worker schedules each

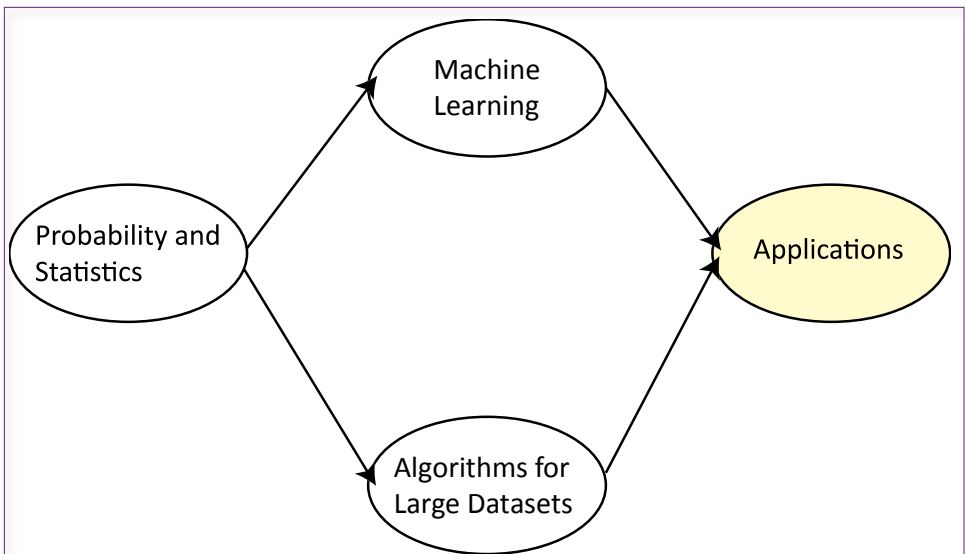


week—leading to unpredictability of hours for the workers and discontent over the loss of weekly wages and the dehumanization.

Scheduling employees is a question of prediction and of matching employee skills to the needs of the company. Data mining can make the process more efficient. But, any actions involving employees need to be carefully considered because people can be offended by mechanical decisions made by computers. In the end, companies need to evaluate the labor market as well as relationships with workers. Improving efficiency can be important, but in tight labor markets, keeping and improving existing employees is also critical to the success of the company. One of the challenges of data mining and optimization is identifying the true objectives to be maximized. Is it really best to maximize revenue per employee, or is it necessary to include employee hiring and retraining costs to maximize profits? Data mining and optimization can find patterns and improve specified goals, but it is vital to define the correct goals.

## Perspectives

**What do managers need to know about data mining?** The answer to this question is elusive because the technologies are still relatively young. Figure 1.8 outlines the basic perspectives on data mining. The theoretical foundations come from probability and statistics, so understanding the tools and results requires at least a basic knowledge of the mathematics of probability. Chapter 2 summarizes many of these concepts. Machine and statistical learning research led to the development of some powerful tools and concepts of identifying patterns. Computer science research has led to the development of efficient algorithms for searching large data sets. Many data mining tools seem straightforward mathe-



**Figure 1.8**

Perspectives on data mining. Data mining requires the integration of several complex topics. Probability and statistics form the foundation of the theory. Machine (and statistical) learning led to the development of important tools. Computer science research led to efficient methods to analyze large data sets. But the focus of this book is on understanding the tools to solve business problems.

matically, but with large data sets, they can be impossible to compute without some clever programming.

Several textbooks and courses exist in all three of these areas. In many programs, the emphasis lies in these areas because the concepts and tools are relatively new. Considerable work remains to be done in developing and improving the concepts and tools. Consequently, less work and even fewer books exist in the area of applying the tools to solving business problems. This last area, business applications, is the focus of this book. Many tools now exist and are easy to use without requiring detailed knowledge of statistics and computer programming. It is still important to understand some of the basic concepts of these underlying disciplines but it is not longer necessary to be an expert in those disciplines to participate in data mining projects.

### Probability and Statistics

In many ways, probability and statistics are the true fields of data mining. Statistics is concerned with identifying relationships—particularly with proving which relationships are strong enough to exceed random chance. By creating solid mathematical foundations and definitions, probability and statistical theory make it possible to discuss and explore complex relationships in precise terms. The drawback is that it takes several courses and months or years of work to understand the underlying concepts used by the high-level data mining techniques. If you have the mathematical background, check out (Hastie et al. 2009), which explains the statistical foundations of most of the common data mining tools.

## Machine Learning

Machine learning is an important area of research that developed computer algorithms for identifying patterns. The research led to the development of **neural networks** which is a way of modeling data that is designed to mimic the way the human brain works. The technique is explained in more detail in Chapter 7. It is a version of machine learning because the network has to be trained based on data with known outcomes. Essentially, a set of nonlinear relationships are estimated using input data with known outputs. The network is then useful at identifying patterns from new data—it has learned the relationships. The method has proven useful at solving complex problems that were difficult or impossible to solve with traditional programming techniques. In particular, it is commonly used for handwriting and speech recognition tasks. It can also be used to analyze business data. Much of the early work is described in the collected work of (Rumelhart and McClelland 1986). These volumes also contain interesting discussions on the nature of machine learning. The basic question is how to design algorithms that can use new data to improve the underlying models to make better decisions. Much of the work is related to statistics, but the machine part of the problem often focuses on the development of new algorithms or new ways to examine the problem.

## Computer Science: Challenges of Large Data Sets

Understanding the mathematics and statistics is only one aspect to developing data mining tools. Remember that data mining is often used for huge data sets—containing complex interactions. As fast as computers are, problems still exist that cannot be solved with brute force. It is just not possible to ask a computer to examine trillions of rows of data with thousands of attributes and come up with all possible relationships—unless you are willing to wait a few thousand years for an answer. This problem is the focus of computer science—finding ways to examine huge data sets efficiently. One of the classic data mining technologies (market basket or association rules covered in Chapter 6) gained prominence when Agrawal created the a priori algorithm in 1995 that could efficiently examine millions of combinations of attributes. Solving these types of problems is still an important aspect of computer science research. Even a few years ago, people working in data mining needed to understand some of the algorithms and tricks because they often had to write or customize programs to analyze the data.

Today, most of the existing basic data mining methods are well understood and common programs exist to handle even relatively large problems. A few examples in this book have millions of rows of data and they are solvable in a few seconds. However, be careful. Super large problems still require the assistance of experts trained to optimize the hardware and software. Experts have the hardware and knowledge to reduce computation times from days or weeks to a few minutes or hours. This book contains a few hints about where to expect problems, and the occasional hint of how some tools can be tweaked to improve performance. But, these are merely warnings. This book does not cover the optimization of the tools or applying them to truly gigantic data sets. In general, if you have that much data available, you will need huge servers, and you can afford to hire experts.

## Management Applications

The focus of this book is on how to use data mining tools in business applications. It explores the basic tools in terms of business tasks. It contains several examples



<u>SaleID</u>	SaleDate	CustomerID	EmployeeID
1001	9/12/....	15	7
1002	9/12/....	61	3
1003	9/13/....	83	7
1004	9/13/....	15	2

<u>SaleID</u>	<u>ItemID</u>	Quantity	SalePrice
1001	9301	2	5.95
1001	2932	4	12.39
1002	9301	6	5.75
1002	3351	1	12.15
1002	8371	2	16.39

**Figure 1.9**

Relational tables. Sales data is broken into small pieces stored separately in each table. Updates are fast and can be protected from common problems such as power failures. But retrieving data requires linking the tables.

of realistic data and applies the tools to explain the configuration of the data. It also explains how to understand and analyze the results from various tools.

Using data mining tools requires (1) understanding the purpose of the tool, (2) configuring the data to match the layout required for the tool, and (3) understanding and evaluating the results to make decisions. The main chapters in this book focus on these three steps, and the chapters are organized by the purpose of the tool. In each chapter, study the examples, set up the sample data, and run the same problems. Configuring the data and the tool options is often an important step in the process. You will learn the methods faster by performing the basic steps.

## Database

Most companies store data in a DBMS, typically one that relies on the relational model. The DBMS is typically optimized for transactions processing—to store incoming data as quickly and efficiently as possible. These systems are capable of handling large numbers of transactions simultaneously and of protecting the data in case of hardware or even power failures. The details of designing and building database systems are covered in database textbooks (such as Post 2011).

Databases also have query systems to retrieve individual sets of data. **SQL** is the most common query language in use and it has many powerful capabilities. SQL is useful for programmers developing applications, and it can be used to answer ad hoc questions that involve simple subtotals or details. Chapter 3 covers the basics of SQL because it is a useful way to obtain answers to detailed questions. Questions such as: Which customers bought the most items last year, and which salespeople worked with those customers?

ID	LastName	FirstName	Phone	ID	LastName	FirstName	Email
32	Jones	Martha	111-3333	1009	Smith	Jon	J_Smith@gmail.com
35	Brown	Jack	222-3555	1010	Sanchez	Emir	E_San332@gmail.com
36	Smith	Jonathon	777-0222	1011	James	Jeff	Jeff_J009@live.com
39	Masters	Penny	333-4444	1013	Monday	Mary	MandM19@gmail.com
				1014	Stiles	Donna	Dstiles@live.com

NewID	LastName	FirstName	Phone	Email
1	Jones	Martha	111-3333	
2	Brown	Jack	222-3555	
3	Smith	Jonathon	777-0222	J_Smith@gmail.com
4	Masters	Penny	333-4444	
5	Sanchez	Emir		E_San332@gmail.com
6	James	Jeff		Jeff_J009@live.com
7	Monday	Mary		MandM19@gmail.com
8	Stiles	Donna		Dstiles@live.com

**Figure 1.10**  
Extraction, transformation, and loading. Matching data is a common problem. Once it is matched, ID values have to be assigned to enable automated matching for future updates. Numbers might have to be transformed (such as millions to billions). The entire process must be automated.

## Traditional Transactions Processing

**Online transaction processing (OLTP)** represents traditional database applications and is characterized by the need to add small amounts of data from multiple sources. The data has to be consistent and added quickly so actions from one transaction do not interfere with another. Also, the updates need to be protected so that if anything happens during the middle of the update, such as a power failure, the changes can be recovered and the data always remain consistent. One of the most common solutions to these problems is to use a relational database system that breaks data into small pieces stored in separate tables. For example, as shown in Figure 1.9, a typical Sale table might contain columns for an identifier (SaleID), a reference to the customer (CustomerID), the date (SaleDate), and possibly a salesperson (EmployeeID). Creating a new sale simply requires inserting one row of data with values for those attributes. A list of items sold will be stored in a second table (SaleItem) with columns such as SaleID, ItemID, Quantity, and SalePrice. Updating the tables with new data is fast and straightforward to protect.

The challenge lies in retrieving the data. Answering questions involving sales involves both of the main tables, plus probably a Products table and a Customer table. Each row from the tables has to be joined to data in another table. Query languages such as SQL make it easy to define the connections. However, the query engine has to retrieve data by matching values in one table with data in other tables. This process requires many lookups. Most systems seek to improve performance by creating **indexes** on the columns—almost always on the primary keys. But indexes need to be updated when rows are added to a table, so adding several indexes to the database means each update now requires many changes to indexes as well. What began as a single row insert suddenly becomes complicated and can potentially slow down all updates.

## Data Warehouse and Analytical Processing

In essence, storing data and performing intensive searches are conflicting uses of a database. One solution is to create a second database or **data warehouse** which is a copy of the transaction data. This copy is bulk-updated on a schedule—not for every new transaction. Consequently, the data warehouse can be stored in new formats that include multiple search indexes and even duplicate data to increase performance for data retrieval. New methods of retrieving data that do not require SQL queries are used to make it easier to retrieve data for exploration and analysis. These **online analytical processing (OLAP)** systems are explained in Chapter 4.

### Data Sources

One function of a data warehouse is to combine data from multiple sources. Many companies today use enterprise resource planning (ERP) systems to handle accounting and other data-intensive tasks. This data is usually stored in a relational DBMS and it is internally consistent. Many of these systems have the ability to transfer specific data to data warehouses. These systems are relatively easy to configure and use with OLAP and data mining systems.

On the other hand, companies also tend to have other sources of data that might be stored in separate databases or even individual files. Engineering, research, marketing, finance, HRM, and other areas of the company might have created unique systems to hold data. If this data is needed for analysis, it has to be located and identified. One challenge with finding data is that it is often unclear exactly how it is defined or what it means. It is even more confusing when departments use different terms—such as Client instead of Customer. A key aspect of data warehouses is the ability to retrieve data from diverse sources, including PC databases and spreadsheets. As you locate each of these sources, it is critical to maintain detailed records of the data definitions, formats, locations, ownership, and security constraints. Eventually, a process has to be created to extract the data, transform it, and load it into the warehouse—and all steps have to be handled automatically with minimal errors or human involvement.

### Data Extraction, Transformation, and Loading

With luck, most important data will come from an ERP system. A primary benefit of ERP systems is that the major work of integrating the data and ensuring consistency has already been performed. Data taken from other sources typically needs considerable work to ensure all of the pieces will match. Seemingly simple things such as geographic locations (City, State, and Country) can cause problems. If people are allowed to enter data into simple text boxes—the data is guaranteed to be inconsistent. People will always abbreviate and misspell names and never do it consistently. Similarly, if multiple sources exist for customer or employee data, matching the data is going to be painful. Consider two data files with contact information created by two executives that contain basic employee information (Last Name, First Name, and Address). If each file has an entry for John Smith, do they both refer to the same person? Perhaps you can compare the address or even a phone number or e-mail address; but what if one of the lists is older than the other and the person has moved? Figure 1.10 shows an example of a simple merge of contact data. What if one of the lists has “John Smith,” and the other “Jonathon Smith,” are they the same people? Similar problems exist with almost any type of list, but the contact problem is common today when managers create their own

personal lists. If the lists contain only a few dozen entries, the problem is reasonable. When the lists contain hundreds of entries, they become difficult to integrate. Carefully crafted database queries can help, but some names will still need to be checked by humans. Moving forward, if the separate lists will be maintained in the future, the matching process has to be automated. Typically, a master list is created with unique ID values, and the individual data sets are modified to include the ID number from the master list. Then the data can be merged in the future by matching the ID values.

The process of retrieving data, cleaning it to verify consistency, and loading it into the data warehouse is often referred to as **extraction, transformation, and loading (ETL)**. The examples given here provide only a glimpse into the challenges faced at this stage of designing a data warehouse. In most OLAP projects, developing the automated ETL processes takes 60 to 80 percent of the time and effort of the entire project. This process often requires programmers with advanced knowledge of databases and queries. But it also requires business analysts who talk to the managers and workers to find the data, determine its precise definition, and write the transformation steps.

## Software Tools

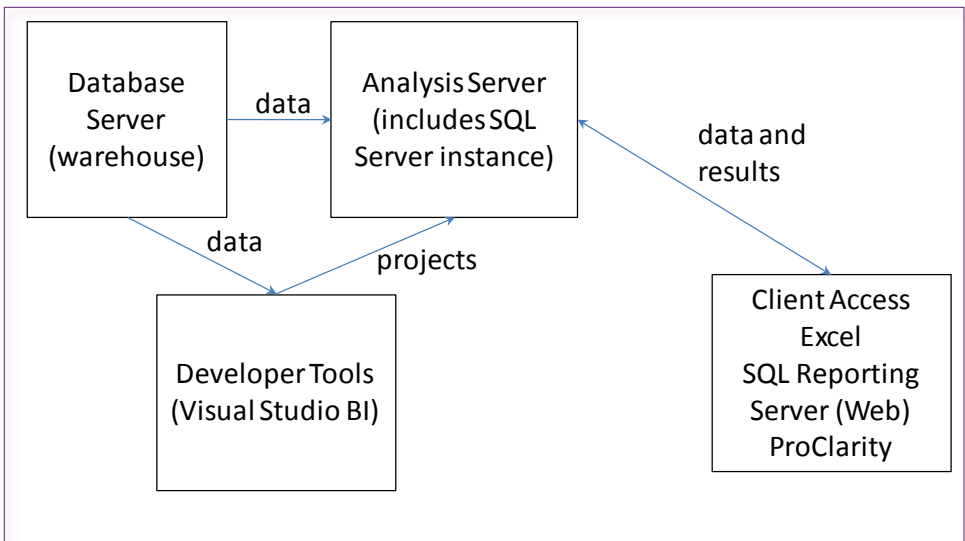
---

### What statistical and data mining tools are used in this book?

The tools available for data mining and statistical analysis have expanded considerably in the past few years. The big database vendors now provide tools—typically integrated with their database systems. Oracle has a data mining add-on for the enterprise suite. Microsoft has SQL Server Analysis Services (SSAS), and IBM purchased the statistical tool vendor SPSS in 2009. Standalone tools also exist, and a couple of them are used in this book. In general, the tools integrated with a DBMS are easier to use—particularly in a production environment. Although they are often more expensive than standalone tools, the integrated environment can save time and money during development. Still, the tradeoffs are difficult decisions that need to be made when designing a data mining strategy.

### Data Mining Techniques

Many techniques can be classified as data mining tools. This book concentrates on the core group of methods readily available in common tools. Rather than focus a chapter on each specific technique, this book is organized by the application goals. It begins with techniques that can run largely unsupervised—where the tools learn from the data and make decisions without requiring the analyst to build models and carefully guide the tool. Chapter 5 examines the issues of clusters. How close are data points to each other? What groups can be formed from the data where items within the group have similar attributes compared to items in other groups. Chapter 6 examines the classical data mining technique of association or market basket analysis. It addresses questions such as which items are commonly purchased together? It also applies to any events that might happen at the same time. Chapter 7 focuses on the evaluation of dimensions or attributes. It examines several techniques that require interaction or supervision with the analyst, including regression analysis, Naïve Bayes, Decision trees, and neural networks. All of these tools are defined and described in that chapter. Chapter 8 examines common techniques for evaluating and predicting time series data. Time series data consists of observations over time. Patterns often exist in terms of seasons, months, or daily changes. Chapter 9 introduces geographical analysis—which focuses on the visual display of data related to location.



**Figure 1.11**

SQL Server Analysis Services components. In a production environment, these pieces are usually installed on separate machines. For development, it is easiest to install everything on a single computer.

## Data Mining Tools

The primary data mining tool used in this book is Microsoft's **SQL Server Analysis Services (SSAS)** data warehouse and data mining tool. Through Microsoft's MSDN Academic Alliance program, the tools are relatively inexpensive for educational purposes. The database and other tools are easy to install, can run on simple hardware including laptops, and are comparatively easy to use. (Although the existing documentation is weak and a few tricks are needed, but they are explained in this book.) The tools and projects can be scaled up to production environments, and Microsoft's developers have reported the ability to handle huge data sets—with the appropriate hardware (server farms), and expert tuning. Other tools exist and can also be obtained on educational licenses (e.g., Oracle and IBM), but their tools tend to be more difficult to install and administer. It is also straightforward to get demonstration copies of the software for free use for several months, so students can load copies onto their own computers..so straightforward to get demonstration copies of the software for free use for several months.

Figure 1.11 shows the basic layout of SSAS. Each of the components is necessary to develop any projects—except the separate client tools are not needed for initial development and testing. In a development system, it is easiest to install the database server, analysis server, and the development tools (Visual Studio) all on the same computer. Basically, use the SQL Server installation process and choose the option to install everything. Putting everything on one machine simplifies the security permission issues. Be sure to add the appropriate users as administrators for the servers. If the pieces need to be split, moving the database server to a separate machine is the easiest—particularly if mixed authentication is used so that users can log in with a SQL Server username and password. Creating a separate

Analysis Services Server is more complex because developers need administrator access to the service, which means establishing user logins across the network. The process is straightforward if all of the machines and users are logged into Active Directory, but it still requires time to set up the permissions. Also, every project created requires modification of the project's deployment property to specify the new server—the default server is always “localhost.”

Several standalone tools exist for data mining. Some are expensive, others are free. The two leading free tools are the R System (<http://www.r-project.org/>) and Weka (<http://www.cs.waikato.ac.nz/ml/weka/>). The R System is actually more of a programming language and requires some time to learn—although a few books and sample code exist on the Internet. The Weka system is relatively easy to use and is demonstrated in a few cases in this book. If you want the Windows version, check carefully to find the current version.

## Statistical Tools

Several standalone statistical packages can be used for traditional statistical analyses as well as data mining. The specific features vary by tool and the prices vary enormously. The high-end packages SPSS, SAS, and Stata are commonly used in research disciplines and are often available at universities. Their usability, configuration, and results tend to be specific to the individual packages. Their main drawback is the price and the fact that data has to be specially configured to work with the tools. These packages are not covered in this book, although SAS does sell one of the leading data mining tools.

A few open-source statistical tools are available now. They also require data to be stored in specific formats, but they are low cost and relatively easy to install and use. One of the interesting tools is gretl (<http://gretl.sourceforge.net/>) which is useful for econometrics (logistic) and time series analysis. It is available free from source forge and is easy to install and run.

Most standalone tools can read data that is stored in comma-separated-values (CSV) formats. These files are text based and are relatively easy to create for small datasets. They are more difficult to deal with when the number of data rows gets into the millions. Most of the tools have some ability to use Microsoft's ODBC database connection methods, so it is possible to retrieve data directly from the database. However, for the relatively small examples in this book, the CSV format is simpler because it requires less configuration effort and fewer problems with security issues.

The standalone tools are useful in this book for a few reasons. First, to show that data mining can actually be performed at relatively low cost. Many people use MySQL or PostgreSQL as the database systems to reduce costs even further. Second, Microsoft's Analysis Services has some slightly unusual approaches to many tools. The methods work, and they are designed to run relatively unsupervised. However, the quirks lead to output that is somewhat different from traditional data mining tools. Hence, it is useful to compare the Microsoft results to those from the traditional or theoretical models. To follow along, you should obtain and install both the Weka and gretl packages.

## Production Systems and Scaling

Data mining and OLAP tools are designed to handle huge data sets—into the terabytes of storage. The tools used here (particularly Microsoft SQL Server Analysis Services and Weka) have all been used on giant problems. However, these huge



<b>Acquisition/Input</b>		
<b>Bias</b>	<b>Description</b>	<b>Example</b>
Data availability	Ease with which specific instances can be recalled affects judgments of frequency.	People overestimate the risk of dying due to homicides compared to heart disease.
Illusory correlation	Belief that two variables are related when they are not.	Ask any conspiracy buff about the death of JFK.
Data presentation	Order effects.	First (or last) items in a list are given more importance.
<b>Processing</b>		
Inconsistency	Difficulty in being consistent for similar decisions.	Judgments involving selection, such as personnel.
Conservatism	Failure to completely use new information.	Resistance to change.
Stress	Stress causes people to make hasty decisions.	Panic judgments and quick fixes.
Social pressure	Social pressures cause people to alter their decisions and decision-making processes.	Majority opinion can unduly influence everyone else: mob rule.
<b>Output</b>		
Scale effects	The scale on which responses are recorded can affect responses.	Ask a group of people to rate how they feel on a scale from 1 to 10. Ask a similar group to use a scale from 1 to 1,000.
Wishful thinking	Preference for an outcome affects the assessment.	People sometimes place a higher probability on events that they want to happen.
<b>Feedback</b>		
Learning from irrelevant outcomes	People gain unrealistic expectations when they see incomplete or inaccurate data.	In personnel selection you see how good your selection is for candidates you accepted. You do not receive data on candidates you rejected.
Success/failure attributions	Tendency to attribute success to one's skill and failure to chance.	Only taking credit for the successes in your job.

**Figure 1.12**

Biases in decision making. Without models, people tend to rely on simplistic “rules of thumb” and fall prey to a variety of common mistakes. These errors can be minimized with training and experience in a discipline. They can also be minimized by having computer systems perform much of the initial analysis.

problems require specialized hardware and careful configuration to function in realistic times. Although, some bloggers on the Web have reported running Weka for several months on a single problem, so it depends on how long you are willing to wait. The point is that the high-end tools such as SSAS can scale to handle huge problems. This scaling is typically accomplished by running the SQL Server database on a specialized server farm with multiple computers and high-speed

networked drives. The Analysis Services are run on a separate server farm with multiple high-speed processors. The enterprise versions of Windows Server and SQL Server are required to distribute the processing load across multiple computers in the server farm.

The high-end tools often utilize parallel processing to split computations into pieces or threads that can be run on different computers. Open source tools sometimes lag behind in support for parallel processing. However, with the source code available, you could always find a programmer to improve the performance.

In most cases, it is easiest to develop and test data mining models for SSAS on a single developer computer. If the datasets are huge, the initial versions can be built and tested on reasonably-sized samples pulled from the main data. Once the model is developed, the project can be deployed to the main analysis server by changing the project's deployment property. The database connections can also be changed simply by editing a connection string to point to a different server.

## Potential Dangers

---

**What can go wrong?** With any project that is used to guide decisions, it is useful to be cautious and recognize the potential dangers and errors that can arise. Obviously, many things can go wrong, but this section focuses on a few of the problems that are specific to data mining. Recognizing problems means they can be monitored. Even if some cannot be completely avoided, at least the level can be assessed and the results can be interpreted in terms of their usefulness and reliability. For example, decision makers need to know if results are highly variable and sensitive to small changes in assumptions or data.

### Human Errors

Humans can make many errors at every stage in the analysis process. Basic mechanical errors such as recording data incorrectly, converting numbers with the wrong values, recording results with errors, and even typographical errors in the analysis can have serious consequences on the analysis and interpretation of the data. These errors are best avoided through careful procedures. In many cases, it is helpful to use pairs or teams of people to cross-check the work. Also, experience is useful because analysts learn to evaluate data and results in terms of "reasonableness." If extreme variations or outliers exist in the data, they should be verified. When results are highly unusual or surprising, they should be validated. Experience with the tools and with the specific data helps analysts recognize what is "unusual" and overly "surprising."

More subtle human errors arise because the results are interpreted and applied through the human lens of perception. Particularly with complex problems, people are often selective and see the results that are familiar or that they want to see. Remaining neutral, evaluating all possibilities, and considering new conclusions is difficult. Barraba and Zaltman (1991) contains an excellent chapter on the perils of human decision making. Even if data is readily available, and even if analysts use comprehensive data mining tools, ultimately the results are evaluated by humans. And humans make many mistakes in evaluating data and results. Figure 1.12 shows a partial list of the more than 100 biases explored by Barabba and Zaltman. Data and careful analysis can help reduce some of the biases. Using multiple, independent people to evaluate data can help; but it is difficult to find independent people, and beware of "group think." Other studies have shown that

groups tend to make suboptimal decisions—with members often agreeing to go along with weak choices simply to avoid causing conflicts.

A big problem in data mining is applying the appropriate tool for the data and the type of decision. Some errors are blatant; others are subtle and might require a statistician or econometrician to explain the problem (for instance, using simple regression with limited dependent variables). It is also easy to misinterpret results or to extend results to decisions that no longer meet the assumptions. Classic examples include trying to forecast data too far into the future or to time intervals that do not match the underlying data. No simple rules exist to avoid these problems. Only that if the problem is critical, it would be wise to consult a statistician to verify the approach and overall tools.

### **Insufficient Data**

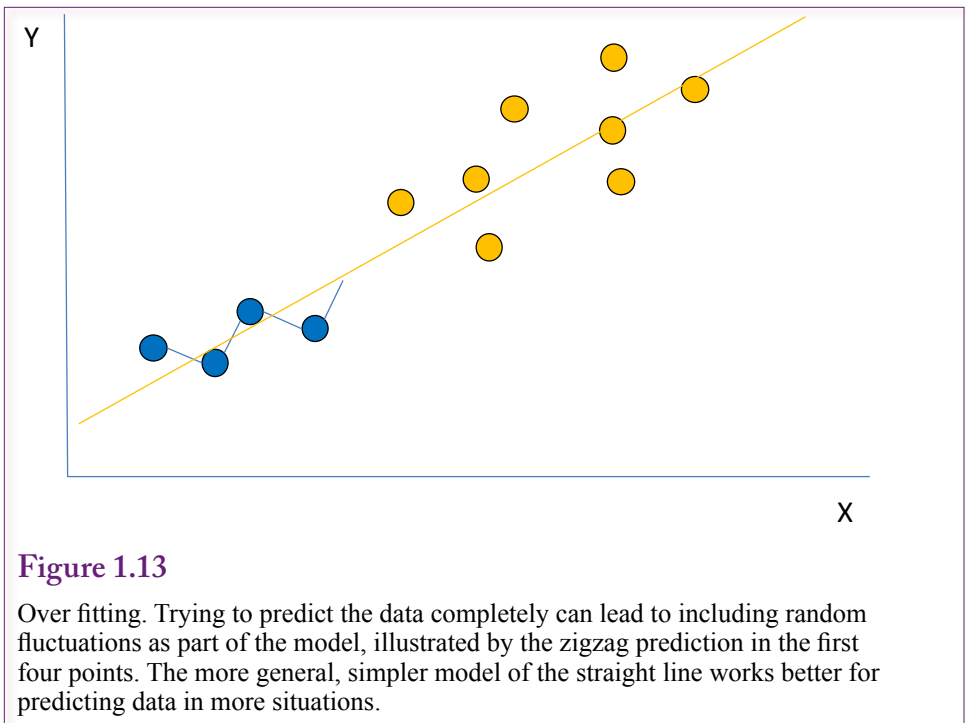
A problem that can arise in research is the issue of insufficient data. In business data mining situations, the opposite problem is more common—too much data. If a company has been operating for a while, usually a large amount of data exists. However, in some cases, it is possible to summarize the data down so far that it reduces the value. For example, it would be best to avoid evaluating data at the annual level. Even if a company has been operating for dozens of years, you would be throwing away detailed quarterly or monthly details that would be useful. Operating on annual data might not provide enough observations to generate robust results—particularly because the underlying economic and structural foundations can change considerably over a couple of decades.

The amount of data needed depends heavily on the type of problem being analyzed. In some cases, it is possible to estimate the minimum number of observations needed, but this approach is rarely needed for data mining. Typically, regression and time series problems are relatively robust with a hundred observations or more, and these levels are easy to achieve in most business problems. But, research involving customers or employees might have more limited data. With smaller data sets the choice of methodologies becomes more critical. Avoid making conclusions based on small sets of data. When in doubt, consult a statistician with knowledge and experience in research techniques. Remember that no matter what techniques might be used to compensate for small data sets, the results ultimately are based on a limited amount of information and can easily be affected by random peculiarities within that set.

A common technique used for 25 years in research is bootstrapping, which draws thousands of random samples from the existing data and generates new data sets. In theory, these new datasets should have the same distribution as the original data, and the expanded amount of data makes it easier to use some data mining techniques. However, a study in 2008 indicated that bootstrapping (and cross validation explained in another section) are ineffective and misleading in small samples (Isaksson et al. 2008).

### **Bad Data**

Most corporate data is probably sound. Sales and financial data tend to be critical to the company and investors, with accountants and auditors examining and verifying the accuracy of much of the data. However, bear in mind that some publicly-released accounting data is not audited—such as quarterly or monthly accounting statements. Even if the base data is accurate, some unaudited data might contain errors.



Still, financial data is going to be better than many other kinds. Asking customers, and even employees, to respond to surveys can lead to questionable data. Although few studies exist, it is likely that as much as 60 or 70 percent of the personal data collected via Web sites is inaccurate. Asking customers for age, income, and other demographic data might irritate customers. In attempts to protect their privacy, people will randomly click entries. When creating surveys, try to avoid asking questions that people are unwilling to answer. Try to provide incentives for people to be accurate.

Integrating data is also a difficult problem in some companies. When data is stored in diverse systems across the company, it is challenging to clean the data and ensure that everything matches. Mismatched data can lead to serious errors in the analysis and results. It is one of the reasons the ETL stage takes so much time—it is crucial that only correct data be loaded into the data warehouse. In almost every case, it is better to leave out data than to include data that can be wrong.

## Over Fitting

Over fitting is one of the classic issues with data mining. In fact, it is the reason that the term data mining was originally pejorative. Research statisticians are warned against data mining because it can violate the assumptions required for statistical testing. Think about a small problem with sales data from a couple hundred customers. With data mining, it is tempting to throw every possible tool on the data and see what shows up. With some tools, it is even possible to tweak the parameters to more closely fit the specific data. In the end, this process can generate a model that almost perfectly describes the specific set of data. But, what if this specific set of data is not completely representative of all of the customers or every situation? The model is over fitted to the specific data.

Another way to describe the problem is to note that most situations include three major elements: (1) A dependent variable to be forecast or modeled, (2) A set of independent variables that represent changes in underlying factors, and (3) Random error. The goal of statistics is to find the relationship between the dependent variables and independent variables while measuring (and ignoring) the random error. Over fitting a problem results in a model that incorporates the random error into the factor relationships. These random effects are likely to be different in other times and other cases, but the over-fit model treats them as if they will always be the same. Figure 1.13 illustrates the problem. Over fitting the first four points leads to a zigzag model that is unlikely to fit other points and can lead to strange predictions. The simpler, more general model of a straight line does not perfectly fit the four points, but it does a much better job of applying to additional data.

In any case, a model that is over-fit will work well on the specific training data, but perform poorly on any other version of the data. In other words, it is dangerous, because the results are not applicable to any other situation. And what is the point in describing and predicting something that already happened? The goal is to create a model that describes the overall situation and can be applied to other cases (subject to random error).

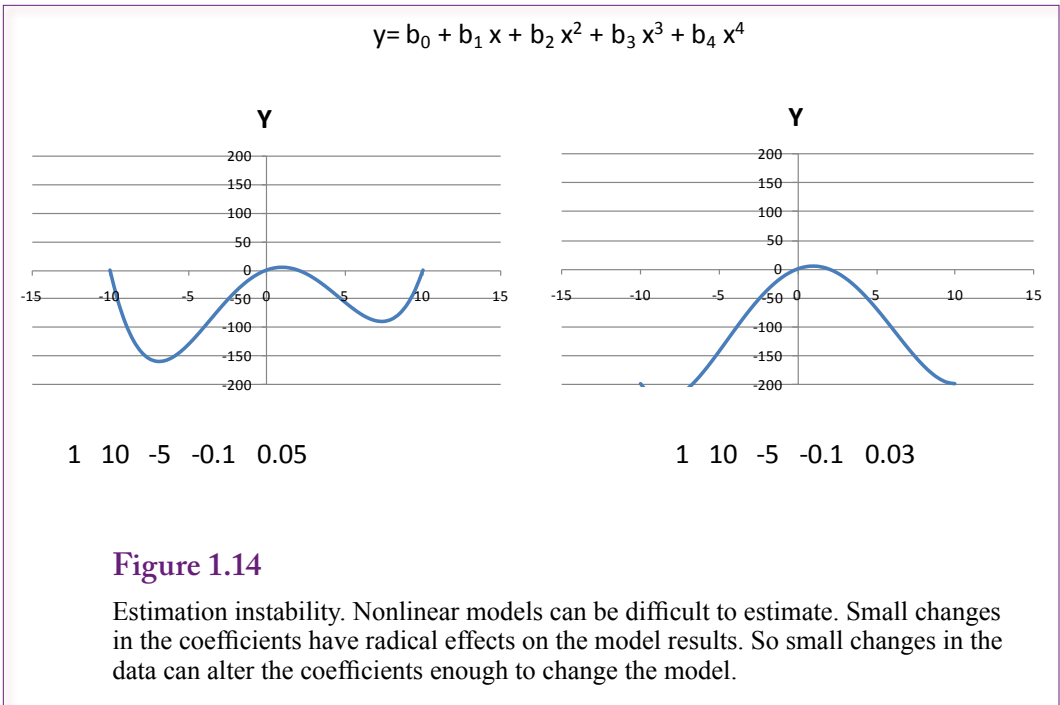
One method commonly used to test for over fitting is to withhold a random sample of data from the training set. By default, SSAS reserves 30 percent of the observations for this testing sample. The model is estimated on the main 70 percent of the data. It can then be tested on the withheld 30 percent to see if the results are close to the same. If they are radically different, the model is probably over fit and needs to be discarded.

A related approach is to split the data into multiple sets (typically 10 sets) and estimate models on combinations that leave out one of the sets at a time—resulting in ten different model estimates, each based on a different 90 percent of the data. This **cross validation** approach is a useful way to test for over fitting. However, it still applies to only the data in the entire sample. If the data is not representative of the entire set of possible data, the model can still be over fit to that specific time period or set of data.

## Random Chance

All events are affected by random chance, so real-world data contains random errors. As shown in Chapter 2, most random errors fall within a fairly tight range, but extraordinary events can always arise—particularly with thousands or millions of observations. Consider simple events such as lotteries. The vast majority of people lose every week, but eventually someone wins. This winning selection gets all of the press, which distorts perceptions, but the point is that even rare events happen. Do not mistake random events for causation. Consider the slightly more complex case of financial investments. Check any financial newspaper or Web site. You will see rankings and stories on investment firms that have successful track records. But, do these firms, or their data mining models, truly have a method that predicts in all cases, or is it the same as the lottery winners? A random winner gets all of the press attention. At any point in time, some person or model can have amazing results—simply by random chance.

Zweig (2009) makes the point by being critical of data mining tools in the investment community. He mentions examples from David Leinweber's book *Nerds on Wall Street*. To prove his point, Mr. Leinweber built a model that



**Figure 1.14**

Estimation instability. Nonlinear models can be difficult to estimate. Small changes in the coefficients have radical effects on the model results. So small changes in the data can alter the coefficients enough to change the model.







showed over a 13-year period 75 percent of the variation in U.S. stock prices was explained by the annual production of butter in Bangladesh. He increased the percentage to 99 percent by including U.S. cheese production and the total population of sheep in the U.S. and Bangladesh. The model is clearly nonsensical. Yet, Mr. Leinweber notes that he still gets calls from would-be investment managers who ask for details on the model to create a new investment fund. The point is that random events do arise in practice, and random correlations are going to show up if enough data is examined in a large number of combinations. All results need to be examined for reasonableness and tested against other sets of data.

### Estimation Instability

A difficult problem that arises in complex data is that the results can be unstable. Small changes in the data result in radical shifts in the model coefficients and results. If the estimation is run only one time, this instability can be hard to spot. Changing the data set by removing observations or estimating the model on new datasets will reveal these instabilities—simply compare the coefficients and results from different data sets.

Instability occurs frequently with non-linear models. Figure 1.14 shows what happens with a relatively simple quartic function (fourth-degree polynomial). Notice the difference in the single coefficient from 0.05 to 0.03. The resulting model is quite different. Given random changes in data, it is possible that either value could be estimated with slightly different versions of the data. Some tools are even more nonlinear and are sensitive to small changes in the data. Even if it is possible to estimate the model reliably, it would be risky to apply the model to new situations where it might predict radically different results.



	Company	ZIP size	DB size
	Rolling Thunder Bicycle Company	11 MB	78.8 MB
	Diner	0.6	11.2
	Corner Med	1.5	55.3
	Basketball	1.3	29.7
	Bakery	47	349
	Cars	0	3

**Figure 1.15**

Cases and database sizes. ZIP files are the size of the file to download. DB Size is the typical size of the MDF database file. It is best to configure the database with the option for Recovery Mode set to Simple to eliminate the transaction log.

A second common situation that results in unstable coefficients is created when multicollinearity is high among the attributes. **Multicollinearity** means that the data from multiple variables can be expressed as a linear combination. For example,  $X_1 = 5X_2 + 4X_3$ . Using all three  $X$  variables in a model will fail in most cases because there are only two pieces of information instead of three. If the data are not quite perfectly linear, the values can be estimated, but they tend to be unstable. Small changes in the data can result in highly different values for the coefficients.

Some techniques exist for finding stable estimates (such as ridge regression), but they are not covered in this book. The most important point is to test for instability. The simple solutions are to reduce the number of attributes in the model and move closer to linear models.

### Model Instability

Model instability is another potential problem, and it is different from estimation instability. It is possible to estimate coefficients that are stable and consistent, yet lead to a model that is unstable. For example, models that include observations over time often use difference equations, such as  $Y_t = aY_{t-1} + bX$ . Some values of the coefficients will cause the model to be unstable—for example, increasing values of  $Y$  will lead to ever larger values and the forecasts will quickly lead to huge numbers that are out of control.

These models can work for limited ranges of data but extrapolating them into the future results in unrealistic projections. It is important to examine every model over a wide range of data. Projections that are unrealistic will help determine the

valid range of data for the model. Some models will work over a wide range of data; others will work for only a small set—such as one or two years. The key is that you need to know the limits before trying to use the model in real predictions.

## Introduction to Cases

---

**What are the cases used in the book?** As with many other mathematical and business concepts, data mining is easiest to understand by working with examples. Six different cases are used through this book to illustrate various techniques in data mining. Some of the cases are relatively general; others were chosen to illustrate specific concepts. Four of the cases use data that was generated based on realistic data, but represent fictional companies. It is difficult to obtain actual detailed data from operating companies. Two of the cases do use actual data, but the amount of data is limited as a result.

The databases can be downloaded from the book's Web site. The files are stored in CSV text files that have scripts to build the database in SQL Server and load the table data. Some security permissions are needed to create the tables and bulk load the data. In general, it is easiest to create the databases with DBA permissions. Databases are built and loaded one at a time. Load the main script file into the SQL editor. As indicated in the file, change the database name and the path that points to the data files. Then run the script to create the tables and load the data.

Figure 1.15 summarizes the cases in the book. Note the database sizes. The ZIP size is the size of the downloadable file—it will affect the time it takes to transfer the data. The DB size is the base size of the SQL Server MDF file. Note that the Bakery case contains a table with over seven million rows. When you create these databases, you should be careful to set the Recovery Mode to Simple—it is an optional parameter when creating the database and it can be changed after the database is created. If the mode is left on the default Full method, every data change will be written to a transaction log file—which quickly becomes substantially bigger than the original data file. This file is as much as seven gigabytes for the Bakery database. With most data mining projects, either the Simple or Bulk Load option is preferred because it holds down the size of the log file and improves performance. It is also unnecessary because the data can always be reloaded from the original transaction sources.

### Rolling Thunder Bicycle Company

The Rolling Thunder Bicycle Company is a fictional company that builds and sells custom bicycles. Most of the data is realistic, such as prices and weights for components. Sales data is configured to match some economic data. Bicycle prices might seem high to people unfamiliar with the industry, but they are relatively accurate for custom bicycles—possibly even too low. The focus of the data is on customization, sales, and production. Sales start in 1994. Some additional description of the company is available on the base Web site: [www.JerryPost.com](http://www.JerryPost.com), under the Rolling Thunder links. For people unfamiliar with the bicycle industry, it might be helpful to download the Microsoft Access copy of the database on the Web site. It contains the transaction forms for the company, so it is possible to work through sample cases of configuring and building bicycles, as well as placing and receiving orders from suppliers. The SQL Server files contain only the data—not the forms. The slides for this chapter contain the relationship diagram of the tables used for the Rolling Thunder Bicycle database. The Bicycle and Customer tables are the most important, but several tables are used as lookups. Note

Code	Name
FG	Field goals made
FGA	Field goals attempted
TP	Three pointers made
TPA	Three pointers attempted
FT	Free throws made
FTA	Free throws attempted
ORB	Offensive rebounds
DRB	Defensive rebounds
TRB	Total rebounds
AST	Assists
STL	Steals
BLK	Blocked shots
TOV	Turnovers
PF	Personal fouls
PTS	Points (total)

**Figure 1.16**

Player statistic abbreviations. Three point shots TP and TPA are sometimes abbreviated 3P and 3PA, but SQL server requires brackets if columns begin with numbers.

that the Access database also contains forms to generate sales of new bicycles, so it is possible to create new patterns. However, the process requires some thought to get reasonable patterns because of the complexity of the options.

## Diner

The Diner database is fictional data that contains a single table. This table (Diners) lists the date, day of week, meal time, gender of the group, the number of people at the table, the total bill, and whether or not dessert was ordered. Treat the data as if it came from a relatively high-end restaurant. The specific meals ordered are not critical because the chef changes the menu depending on the availability of some items, and on his preferences. Dessert is important because desserts have higher profit margins, and because people who stay for dessert often spend more money on drinks, with even higher profit margins.

## Corner Med

Corner Med is a small healthcare database for a fictional company that has a store-front treatment office. Several physicians, nurses, and assistants staff the site. Patients arrive with common conditions and are treated. The data is fictional but it includes the ICD10 codes for diseases and treatment. The standard drug table is also included. More importantly, the incidence of diseases and treatments are drawn from the U.S. physician's survey so the combinations of items are accurate representations of real cases. As with all of the databases, the names, addresses, and phone numbers are false. The slides for this chapter contain the relationship diagram for the Corner Med database. The primary tables are the Patient, Visit, Employee, VisitDiagnoses, VisitProcedures, and VisitMedications. The goal of the case is to examine data as a business manager—not as a medical expert. The diagnoses codes are fun to read, but not particularly relevant to the business issues

in the case. Medical researchers would use similar data to examine public health trends and explore the effects of treatment options. Business managers are more interested in costs, revenue, and employee management.

## Basketball

The basketball database contains actual data for three seasons of the NBA including playoffs: 2008 through 2011. It lists game statistics for all of the players for every game. The tables are: Teams, Players, Games, GameResults, and PlayerGameStats. From a database perspective, the Games table has redundancies because it lists every game twice—once for the home team and once for the visiting team. Technically, the GameResults table contains the same data, but to track player statistics, it was easier to use the Games table to create the shared GameID. Be careful to avoid double counting when examining team statistics. The included team View is designed to minimize this problem, so use it instead of the base tables. The player statistics are straightforward, but the abbreviations might be unfamiliar to people with limited basketball experience. Figure 1.16 shows the list of abbreviations and their names.

If it seems strange to use sports data in a business case, consider that professional sports is a huge business. No, the case does not focus on ticket sales—although those are likely related to the won/loss record. Instead, think of the management questions involving worker performance and salaries. Sports agents make money by analyzing performance data for their clients and presenting the players (workers) in the best possible light to justify raises. Given the large amount of sports statistics available, data mining is a useful tool for identifying patterns and understanding relationships.

## Bakery

The Bakery case is a straightforward example of business sales. The data could almost come from any type of business. The tables consist of: Sale, SaleItem, Product, and ProductCategory. The company does not track employees or customers. Instead, the focus is on the items and categories sold. The SalesDate in this case is also tracked down to the specific time that a sale was made. Sales in the case run from January 1, 1995 through December 31, 2012. The data is fictional, but a relatively sophisticated mechanism was used in the data generator to ensure that interesting results and patterns arise.

The key feature of this database is its size. It contains almost 2 million sales with almost 8 million rows in the SaleItem table. Although the size is still relatively small in terms of data mining problems; it would be difficult to distribute anything larger. If you are interested in larger and more realistic datasets, search for the Wal-Mart data. Wal-Mart is trying to encourage the teaching of data mining by providing a cleaned set of sales data through a university program. There are restrictions, and some complications in its use, but it is a large set of data. Also, Netflix provided scrubbed rental data for its recommendation contest. Again, the data has restrictions on its use and distribution, but it is a large set of realistic data.

## Cars

The Cars database contains real data. It is a smaller database than the others, but most people are familiar with attributes for cars and the real data make it interesting and easier to comprehend for certain types of problems. The primary Cars database consists of features for over 300 automobiles in the 2012 model year. Basic

data includes weight, number of cylinders, miles per gallon, number of seats, and list price. In most cases, the data represents values for the base model of the car, so it is not meant as a tool to compare all possible versions. Many vehicles have dozens of variations, and students are encouraged to add more data if they want to compare specific models. If you want to use more data, an older copy of the file contains data for 2009 models.

The data file includes a secondary file (*CarSalesMonthly.csv*) that does not load automatically into the SQL Server database, although it is straightforward to import it. It contains data from U.S. government Web sites that lists total vehicle sales by month from 1967-01 through 2012-01. Total sales are broken down into domestic and foreign vehicles for each month. This set of data is quite different from the detailed attributes for 2012 models, but it provides a useful set of data for time series analysis. The sales are a count of the number of vehicles sold in that month (in thousands).

## Summary

---

Data mining is different from statistical analysis of research data. DM is designed for the exploration of data—providing insights, visualization, and early statistical comparisons of data. It has been successfully used in all areas of business—to analyze trends, identify patterns, understand relationships among attributes, and predict results..

Data mining requires a basic understanding of probability and statistics as well as some knowledge of database concepts and tools—particularly queries. Data warehouses are often created to hold data for analysis and exploration. The designs for retrieving and analyzing large amounts of data are different from the designs for storing large amounts of transaction data. Consequently, most organizations build processes that extract data from the transaction processing systems, clean and format the data, and store it in a specialized data warehouse.

Many different software tools exist to analyze data, from traditional statistical packages to dedicated data mining and business intelligence tools. This book focuses on the readily available data analysis tools in SQL Server 2008, but a few more traditional tools are also used to compare the results.

Many things can go wrong in data mining—from human errors, to bad or insufficient data, failing to account for random chance, estimation and model instability, and over fitting a model to a small set of data. It is important to understand the common types of errors and recognize when one of them appears. Knowing what to watch for helps ensure that the problems are caught early and the risks minimized.

This book contains sample cases for six different types of organizations. The size of the databases varies in terms of the number of rows and the number of attributes. The cases are useful for illustrating the application of the tools. It is also important to work on the chapter exercises to gain experience with configuring and running the tools and to understand the results. Cases are important for practice. One of the complications of learning data mining is trying to understand and interpret results. Many situations require some understanding of the underlying organization. For the most part, the cases used in this book focus on business problems, so a basic background in business is a good starting point. For more detailed interpretations and for making decisions, you should do some background research on the specific industries to understand the terms and relationships.

## Key Words

---

cross validation	just-in-time
data	knowledge
data mining	multicollinearity
data warehouse	neural networks
database management system (DBMS)	online analytical processing (OLAP)
extraction, transformation, and loading (ETL)	online transaction processing (OLTP)
forecasting	SQL
hyper cubes	SQL Server Analysis Services (SSAS)
indexes	wisdom
information	

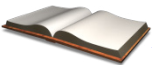
## Review Questions

---

1. What is the purpose of data mining?
2. Why are data, statistics, and data mining important in the decision making process?
3. What are the background disciplines used to develop data mining technologies?
4. What is ETL, why is it so important, and why does it take so much time to configure it?
5. What strengths does SQL Server have as a data mining tool and platform?
6. What are the main dangers in any data mining project?

## Exercises

---



### Book

1. If necessary, install the SQL Server database, SQL Server Analysis Studio, and the Business Intelligence Visual Studio components.
2. Identify a decision you have made in the past year or two and describe which human errors and biases you faced while making the decision.
3. Watch a TV game show where contestants must make decisions (not just answer questions), and identify human biases that exist and that are pressed by the host.
4. Choose an industry or specific firm and identify a common decision that must be made. Specify the type of data that should be collected and what types of patterns might be helpful in analyzing the data?



5. Identify the current version of a commercial data mining/business intelligence tool. Summarize its features and estimate the price of the software.
6. Find a business example of a problem that could benefit from the use of data mining. Identify the data available and the specific decisions that need to be made. If possible, interview a manager in that area or find a business case.
7. Use a Netflix account or Amazon search to evaluate the accuracy of the recommendation engine. Do you agree with the proposed matches?



### Rolling Thunder Database

8. Create the database and install the data. If the database and tables were already created for you on a central server, test your log in and display the data in the Employee table.
9. Identify at least one major decision that must be made by the managers of this company. Looking at the table definitions, what data could be used to help make this decision?
10. List each table in the database and briefly describe the purpose or data in the table. Keep the description general; it is not necessary to explain every column.
11. If the company emphasized online ordering, what additional data would be available to use for data mining?



### Diner

12. Create the database and install the data. If the database and tables were already created for you on a central server, test your log in and display the data in the first 10 rows of the Diners table.
13. Identify at least one major decision that must be made by the managers of this company. Looking at the table definitions, what data could be used to help make this decision?
14. List each table in the database and briefly describe the purpose or data in the table. Keep the description general; it is not necessary to explain every column.



## Corner Med

15. Create the database and install the data. If the database and tables were already created for you on a central server, test your log in and display the data in the Employee table.
16. Identify at least one major decision that must be made by the managers of this company. Looking at the table definitions, what data could be used to help make this decision?
17. List each table in the database and briefly describe the purpose or data in the table. Keep the description general; it is not necessary to explain every column.



## Basketball

18. Create the database and install the data. If the database and tables were already created for you on a central server, test your log in and display the data in the Teams table.
19. Identify at least one major decision that must be made by the managers of a team. Looking at the table definitions, what data could be used to help make this decision?
20. List each table in the database and briefly describe the purpose or data in the table. Keep the description general; it is not necessary to explain every column.



## Bakery

21. Create the database and install the data. If the database and tables were already created for you on a central server, test your log in and display the data in the Cars table.
22. Identify at least one major decision that must be made by the managers of this company. Looking at the table definitions, what data could be used to help make this decision?
23. List each table in the database and briefly describe the purpose or data in the table. Keep the description general; it is not necessary to explain every column.



## Cars

24. Create the database and install the data. If the database and tables were already created for you on a central server, test your log in and display the data in the Employee table.
25. Identify at least one major decision that must be made by the managers of an automobile manufacturer. Looking at the table definitions, what data could be used to help make this decision?
26. List each table in the database and briefly describe the purpose or data in the table. Keep the description general; it is not necessary to explain every column.



## Teamwork

27. Form teams for future assignments. Obtain contact information and determine a method of communication. If available, find a way to work on files together, such as via SharePoint, Groove, or a Web site such as Google Docs.
28. Each person should select a Web business and identify a major decision that needs to be made by that company and the data available. Share the information and then vote on which decision would be the most difficult to answer.

## Additional Reading

---

Barabba, Vincent P. and Gerald Zaltman, *Hearing the Voice of the Market*, 1991, Harvard Business School Press: Boston. [Marketing perspective on collecting and understanding data—triggered from GM's mistakes.]

Brat, Ilan, Ellen Bryon, and Ann Zimmerman, "Retailers Cut Back on Variety, Once the Spice of Marketing," *The Wall Street Journal*, June 26, 2009. [Identifying products to carry.]

Duhigg, Charles "Stock Traders Find Speed Pays, in Milliseconds," *The New York Times*, July 24, 2009. [High frequency trading data.]

Dvorak, Phred, "Clarity is Missing Link in Supply Chain," *The Wall Street Journal*, May 18, 2009. [Supply chain challenges with Zoran.]

Goodman, Peter S. and Gretchen Morgenson, “By Saying Yes, WaMu Built Empire on Shaky Loans,” *The New York Times*, December 28, 2008. [WaMu mortgage lending collapse.]

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman, 2001, *The Elements of Statistical Learning*, Springer: New York. [An outstanding book on data mining, with an emphasis on statistical theory. A graduate-level book that requires a strong mathematics background.]

Isaksson, A., M. Wallman, H. Göransson, M.G. Gustafsson, “Cross-Validation and Bootstrapping are Unreliable in Small Sample Classification,” *Pattern Recognition Letters* Volume 29, Issue 14, 1960-1965, October 15, 2008. [Analysis of the performance of bootstrapping and cross-validation in small samples, showed that the results are weak and the recommend a Bayesian approach instead.]

Johnson, Avery, “Insulin Flop Costs Pfizer \$2.8 Billion,” *The Wall Street Journal*, October 19, 2007. [A good summary of the issues and costs faced by Pfizer and the inhaled insulin project.]

Koudsi, Suzanne, “Sleazy Credit,” *Fortune*, February 19, 2002. [WaMu early warnings.]

Lohr, Steve, “Netflix Challenge Ends, but Winner Is in Doubt,” *The New York Times*, July 27, 2009. [Initial results of the Netflix competition.]

Lohr, Steve, “Netflix Competitors Learn the Power of Teamwork,” *The New York Times*, July 28, 2009. [A more detailed examination of the NetFlix competition and combining algorithms.]

Morrison, Scott, “Google Searches for Staffing Answers,” *The Wall Street Journal*, May 19, 2009. [Summary of Google’s use of data to evaluate employee turnover.]

O’Connell, Vanessa, “Retailers Reprogram Workers In Efficiency Push,” *The Wall Street Journal*, September 10, 2008.

Rumelhart, David E. and James L. McClelland, 1986, *Parallel Distributed Processing*, Vol. 1 and 2, MIT Press: Cambridge, Massachusetts. [A classic collection of work on the foundations and start of machine learning (neural networks). Includes descriptive and technical content.]

Taylor III, Alex, “GM: Death of an American Dream,” *Fortune* November 25, 2008. [A good but optimistic business history of GM’s mistakes.]

Zweig, Jason, “Data Mining Isn’t a Good Bet for Stock-Market Predictions,” *The Wall Street Journal*, August 8, 2009. [Comments on problems with data mining in investments.]

# Fundamental Tools

## What are the foundations of data mining?

How is data stored and retrieved with a database management system? How can data be retrieved more efficiently in a way that is easier for managers to use? Are events and correlations random or are they due to fundamental relationships?

One of the main steps of data mining is to be able to retrieve data efficiently. Queries, typically based on the SQL standard, are a powerful way to look up information in most standard database systems. They are often used to retrieve and format data for analysis and can also be used to find specific pieces of information. However, complex SQL queries are often misunderstood by managers and can return values that might not correspond to the question perceived by the manager.

Managers need a method to explore data—not just retrieve fixed reports and ad hoc queries. In particular, a hyper cube browser is useful for interactively examining subtotals, filtering data to examine specific instances, and drilling down through hierarchies of dimensions.

Probability and statistics are fundamental tools for exploring data. They can determine the difference between random chance and useful relationships. Several basic concepts form the foundation for data mining tools. Most data mining tools are based on probability and statistical foundations.

---

<b>Chapter 2</b>	Finding and Storing Data
<b>Chapter 3</b>	OLAP Cubes
<b>Chapter 4</b>	Probability and Statistics Summary

---

# Database Management Systems

## Chapter Outline

- Introduction, 42
- Relational Databases, 43
  - Tables, 43
  - Data Types, 45
- Four Questions to Retrieve Data, 47
  - What Output Do You Want to See?, 48
  - What Do You Already Know?, 48
  - What Tables Are Involved?, 48
  - How Are the Tables Joined?, 48
- Query Basics, 49
  - Single Tables, 50
  - Introduction to SQL, 52
  - Sorting the Output, 52
  - Criteria, 53
  - Useful WHERE Clauses, 56
- Computations, 57
  - Basic Arithmetic Operators, 57
  - Aggregation, 58
  - Functions, 60
- Subtotals and GROUP BY, 61
  - Conditions on Totals (HAVING), 63
  - WHERE versus HAVING, 64
  - The Best and the Worst, 65
- Multiple Tables, 66
  - Joining Tables, 66
  - Identifying Columns in Different Tables, 67
  - Joining Many Tables, 68
  - Views: Saved Queries, 69
  - LEFT JOIN, 71
  - UNION, 71
- Data Manipulation, 72
  - UPDATE, 73
  - INSERT, 73
  - DELETE, 74
- SQL Server Reports, 75
  - Administration Configuration, 75
  - Creating a Report, 76
- Database Design Concepts, 82
  - Notation, 84
  - First Normal Form, 86
  - Second Normal Form, 86
  - Third Normal Form, 87
- Summary, 88
- Key Words, 89
- Review Questions, 90
- Exercises, 90
- Additional Reading, 93

## What You Will Learn in This Chapter

- What are databases and why are formal query systems necessary?
- How is data stored in a relational database?
- What is the basic structure of a query?
- How do you create a basic query?
- What types of computations can be performed in SQL?
- How are subtotals computed?
- How do you use multiple tables in a query?
- How are reports created in SQL Server?
- How do you create a new database?



### FAA: Air Safety

For years, the Federal Aviation Administration (FAA) and major airlines have had teams dedicated to analyzing air crashes and using the results to improve air safety. The teams have been successful and their results have undoubtedly made flying safer. But, they have faced an interesting problem—with improved safety and fewer crashes, there is little opportunity for investigation. So, they have turned to analyzing data on all flights and pilot reports. As part of the process, the FAA and the airlines have changed their reporting culture—encouraging pilots and controllers to report all potential problems with no fear of blame. These “quality assurance programs” are designed to collect massive amounts of data to be analyzed for “precursors” or potential problems. Modern planes capture detailed flight data (not just the “black boxes” but ongoing reporting from instruments). For example, Southwest Airlines, which started collecting data in 2003, by 2008 had data on more than one million flights. Don Carter, senior manager of Southwest’s flight safety program noted that “We are always asking ourselves, ‘What should we be asking this data that we haven’t thought of yet?’” [Wilber 2008] US Airways used similar data to identify an unusually high number of “unstabilized” approaches where the planes come in too fast or sink too quickly at the last stages of landing. The carrier changed its training and landing checklists and reduced the rate of unstable approaches by more than 70 percent. They used similar data to rewrite charts to improve landings and visibility at McCarran International airport in Las Vegas. Tom Lulkovich, US Airways director of flight safety noted that “everything is about identifying risk here.”

Collecting and organizing huge amounts of data is an important first step. Analysts must also know which questions to ask.

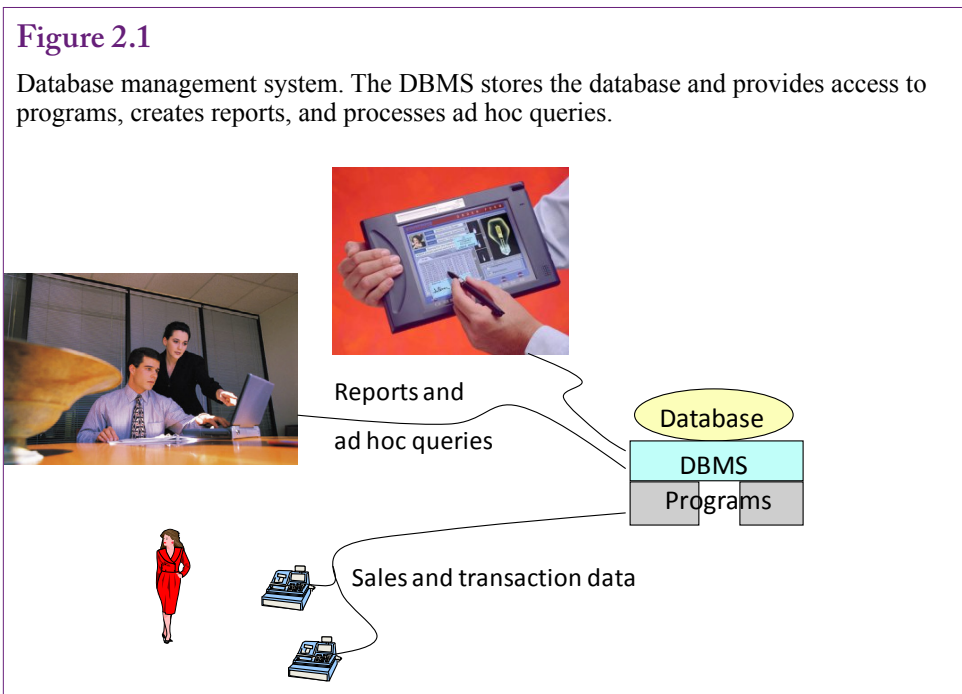
Del Quentin Wilber, “Avoiding Plane Crashes By Crunching Numbers,” *The Washington Post*, January 13, 2008. <http://www.washingtonpost.com/wp-dyn/content/article/2008/01/12/AR2008011202407.html>

## Introduction

**What are databases and why are formal query systems necessary?** Most companies use a relational **database management system (DBMS)** to hold transaction data. These databases form the foundation for applications such as accounting systems and Web servers. Many companies have integrated applications or **enterprise resource planning (ERP)** systems that store all data in a DBMS. These transaction databases are designed to be efficient at storing data, and to ensure that the data remains safe even when multiple operations are performed at the same time and even if power fails half way through a process.

A **database** is a collection of related data. The database is typically stored in specific formats and controlled with a DBMS which provides access to the data through standardized connection methods. Without databases, programmers store data in proprietary structures so the data is available only to a specific program. For example, word processing documents are commonly stored in proprietary formats. It usually works for word documents because most people are willing to use the same word processor to edit a document. However, it is critical for basic business data to be accessible to other programs. Most databases have a relatively standard method to provide data to people and to other software. This process usually involves a **query system**, and most existing query systems are based on the **SQL** standard. Following standards is important because it makes it easier to change the underlying DBMS if necessary, and it is easier for programmers and managers to learn a single query method that can be used in most situations.

Figure 2.1 shows the basic roles of the database and DBMS. The DBMS is software that controls access to the database, supports programs, creates reports for standardized data, and has a query language to process ad hoc queries.



Most major DBMS vendors have implemented a “natural language” query system at some point. Most have abandoned the work in favor of the SQL standard. The problem with natural languages, such as English, is that they are inherently ambiguous. The risk is relatively high that the computer will not understand a question the same way it was intended by the manager asking the question. The DBMS does the best it can to produce results, and it is difficult for the manager to verify that the computer had the correct understanding of the question. Versions of this problem exist with any query language, but formal systems are defined to work in a specific way. By learning the basic rules of the formal query system, it is possible to ensure that the DBMS returns exactly the data needed to answer a specific question. However, you must learn those basic rules.

The main goal of this chapter is to show how to create common queries in SQL Server. Queries are useful for answering ad hoc questions. In some ways, data mining tools covered in Chapter 3 are easier to use than traditional SQL queries. However, queries are still used to look up basic facts and they are used to configure data for some of the data mining tools. Queries and data warehouses are complementary—a good analyst needs to be able to use both tools. Queries provide direct access to data and support row-by-row computations. Data warehouse tools make it easy to summarize data and compare subtotals and explore data.

## Relational Databases

**How is data stored in a relational database?** To learn to write queries to retrieve data, it is important to understand how data is stored in the database. The most important thing to know is that all data in a relational database is stored in tables. To answer business questions or configure data for analysis, it is necessary to determine which tables hold the data needed. This section introduces the basic concepts of tables, and assumes that the tables and data already exist. It does not attempt to explain how to create tables. Relational databases have to be carefully designed or they do not work well. These design issues are briefly covered in a later section of this chapter, but learning how to design a database is covered in other textbooks (Post 2011).

### Tables

A **table** describes a single concept or object. Its **columns** consist of attributes or properties of the object. Data is stored as a row—where each row represents one instance of data. Figure 2.2 shows part of a table for Customers. Customers are

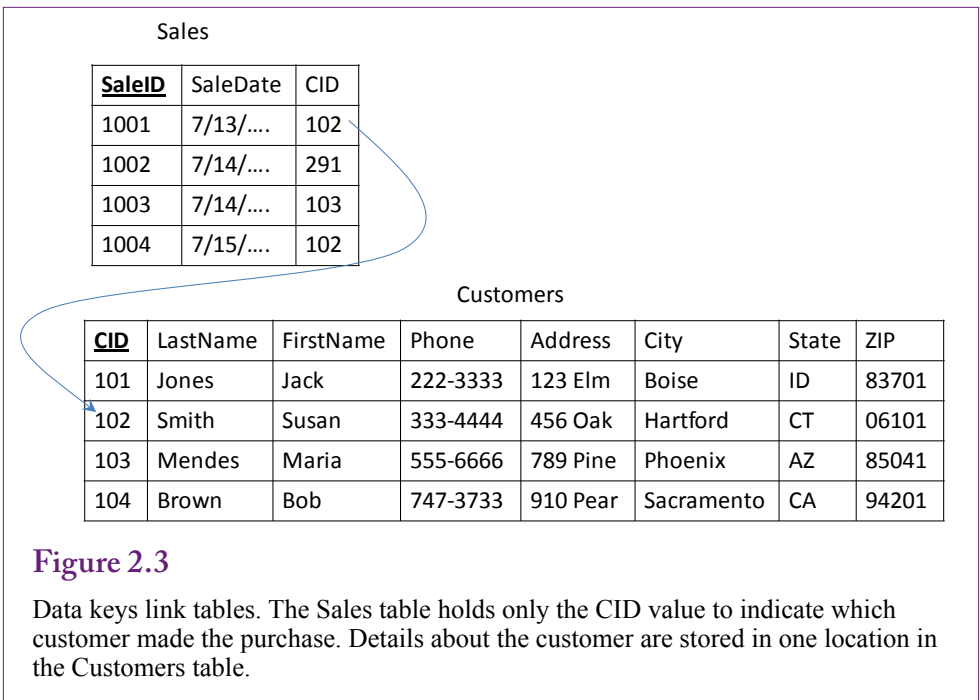
**Figure 2.2**

Sample table: Customers. A table represents a single object or event. Columns are properties or attributes that describe the object. Each row is one instance of data.

primary key: Identify a row

Columns: Properties/Attributes

<u>CID</u>	LastName	FirstName	Phone	Address	City	State	ZIP
101	Jones	Jack	222-3333	123 Elm	Boise	ID	83701
102	Smith	Susan	333-4444	456 Oak	Hartford	CT	6101
103	Mendes	Maria	555-6666	789 Pine	Phoenix	AZ	85041
104	Brown	Bob	747-3733	910 Pear	Sacramento	CA	94201



defined in terms of their name, phone, address, city, state, and ZIP or postal code. Most companies collect additional properties about customers, such as e-mail address. Each row represents a single customer, and all data for that customer is stored in that row. The row is identified by the CustomerID (CID) value—it is the primary key for the table. A **primary key** is a column or set of columns that uniquely identifies a row. Every table must have a primary key. In cases of simple objects, the key is usually created within the DBMS. For example, SQL Server uses an **identity** to generate new key values that are guaranteed to be unique. In the example of the Customers, if the ID value 104 is provided, the DBMS can quickly find the data for customer Brown.

Primary keys are often used to link tables. Figure 2.3 illustrates the process with a Sales table. The Sales table holds a value for CID to indicate which customer made the purchase. Details about the customer are stored in a single row in the Customers table with the matching CID value (102). The main benefit to this approach is that data for each customer is stored in only one location—making it easy to find and easy to change. The drawback is that the DBMS needs a method to quickly match data from multiple tables.

Relational databases can have any number of tables—all related through the key values. Corporate databases can easily include hundreds or even thousands of tables. Creating queries to answer business questions ultimately requires that you know which table to use—which requires knowing exactly what each table represents and knowing the meaning of each column. Figure 2.4 shows the relationship diagram for Rolling Thunder Bicycles. Even this relatively basic set of tables is difficult to fit on a single page. Most business queries focus on a few basic tables, such as Bicycle, Customer, City, Employee, BikeParts, Components, Manufacturer, PurchaseOrder, and PurchaseItem. Some tables, such as CustomerTrans, ManufacturerTrans, ModelSize, and BikeTubes are used for internal accounting

### Figure 2.4

Relationship diagram for Rolling Thunder Bicycles. Creating queries requires understanding all of the tables and columns, although most business questions focus on a few key areas.

and manufacturing purposes and are rarely queried directly. But, it takes knowledge of the industry and the company to recognize the purpose of each table. One of the first steps in analyzing data is to understand exactly what data is available. Looking through the list of tables provides a first glimpse of that data.

### Data Types

Computers deal only with binary data, so all data must be assigned a **data type** that specifies how the data is stored and handled. The basic data types are: Numeric, Text, Date, XML, and binary Object, such as pictures. The catch is that many subtypes exist. Figure 2.5 lists the basic data types in SQL Server. First, note that all text today should use the Unicode (“National”) data type—which supports characters in multiple languages. Second, the appropriate data type should be chosen for each data column. Integer values cannot contain decimal data and the size chosen must be able to hold the largest possible value. Monetary values should be stored using the money type, never integers and never float or real which can round off some values. Dates must always be stored with the datetime data type because it supports searches and subtraction of dates to find the number of days between two dates.

Data Type	SQL Server	Size
Text		
Fixed	char	8K
Variable	varchar	8K
Unicode	nchar, nvarchar	4K
Memo	text	2G, 1G
XML	xml	2G
Number		
Byte	tinyint	255
Integer	smallint	+/- 32767
Long	int	+/-2B
64-bits	bigint	18 digits
Fixed precision	decimal(p,s)	p: 1 to 38
Float	real	+/- 1E 38
Double	float	+/- 1E 308
Currency	money	+/- 900.0000 trillion (8 bytes)
Yes/No	bit	0/1
Date/Time	datetime smalldatetime	1/1/1753 – 12/31/9999 (3 ms) 1/1/1900 – 6/6/2079 (1 min)
Interval	interval year, ...	
Image	image	2GB

**Figure 2.5**

Data types in SQL Server. Use Unicode whenever possible. Be careful with number types to ensure the correct type is chosen for each piece of data.

In most cases, the database structure and data types will already exist for any data that managers use. However, managers might be involved in the design and construction of a new data warehouse that consolidates data from other sources. Hence, it is helpful for managers to know the types of data available to hold the consolidated data.

The binary data types are rarely used in data mining. Data mining has been successfully applied to images and large-text files, but these investigations typically use specialized tools and data storage. Relational databases are generally too slow to use for large collections of huge text and image files.

Extensible markup language (XML) files are increasingly common in business, but they present additional complications in data mining. XML data consists of text that is tagged; similar to HTML tags, but any tag names can be used. The drawback to XML is that multiple types and levels of data are stored within a single XML list. Figure 2.6 provides an example of an XML segment used for an order. To analyze this data, the individual elements must be extracted and stored into new tables. Data mining tools, particularly those in SQL Server, are designed to work with relational tables. It would be too slow to extract data from XML segments on the fly. SQL Server supports XQuery and other XML tools used to extract data from XML files, but these tools need to be run once during the setup to unload the data from XML and transfer it into standard tables.



```

<order>
  <orderID>111</orderID>
  <customer>
    <clD>99</clD>
    <lastName>Smith</lastName><firstName>Mary</firstName>
  </customer>
  <itemList>
    <item>
      <itemid>290</itemid><description>red dress</description>
      <salePrice>132.99</salePrice><quantity>1</quantity>
    </item>
    <item>
      <itemid>171</itemid><description>shoes</description>
      <salePrice>79.89</salePrice><quantity>1</quantity>
    </item>
  </itemList>
</order>

```

**Figure 2.6**

Sample XML data. To analyze this data the individual elements must be extracted and stored into separate tables and rows.

## Four Questions to Retrieve Data

**What is the basic structure of a query?** Every attempt to retrieve data from a relational DBMS requires answering the four basic questions listed in Figure 2.7. The difference among query systems is how you fill in those answers. You need to remember these four questions, but do not worry about the specific order. When you first learn to create queries, you should write down these four questions each time you construct a query. With easy problems, you can almost automatically fill in answers to these questions. With more complex problems, you might fill in partial answers and switch between questions until you completely understand the query.

Notice that in some easy situations you will not have to answer all four questions. Many easy questions involve only one table, so you will not have to worry about joining tables (question 4). As another example, you might want the total sales for the entire company, as opposed to the total sales for a particular employee, so there may not be any constraints (question 2).

**Figure 2.7**

Four questions to create a query. Every query is built by asking these four questions.

What output do you want to see?

What do you already know (or what constraints are given)?

What tables are involved?

How are the tables joined?

## What Output Do You Want to See?

In many ways, this question is the most important. Obviously, the database engine needs to know what you want to see. More importantly, you first have to visualize your output before you can write the rest of the query. In general, a query system answers your query by displaying rows of data for various columns. You have to tell the DBMS which columns to display. However, you can also ask the DBMS to perform some basic computations, so you also need to identify any calculations and totals you need.

You generally answer this question by selecting columns of data from the various tables stored in the database. Of course, you need to know the names of all of the columns to answer this question. Generally, the hardest part in answering this question is to wade through the list of tables and identify the columns you really want to see. The problem is more difficult when the database has hundreds of tables and thousands of columns. Queries are easier to build if you have a copy of the class diagram that lists the tables, their columns, and the relationships that join the tables.

## What Do You Already Know?

In most situations you want to restrict your search based on various criteria. For instance, you might be interested in sales on a particular date or sales from only one department. The search conditions must be converted into a standard Boolean notation (phrases connected with AND or OR). The most important part of this step is to write down all the conditions to help you understand the purpose of the query.

## What Tables Are Involved?

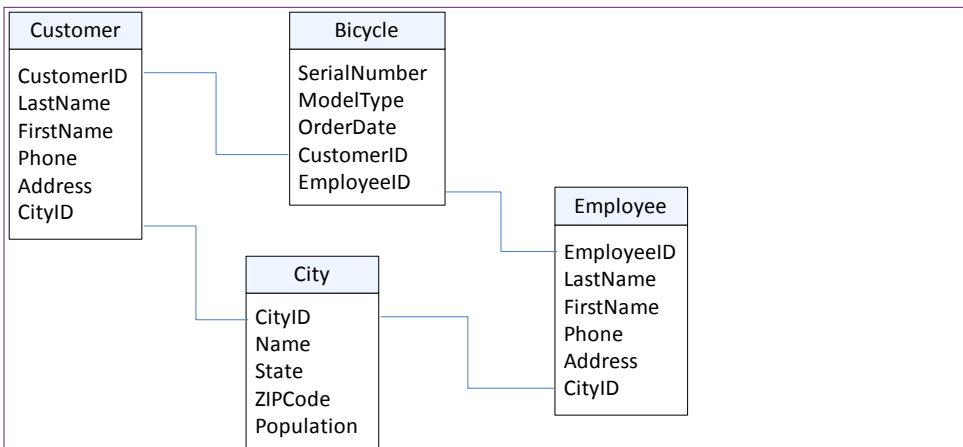
With only a few tables, this question is easy. With hundreds of tables, it could take a while to determine exactly which ones you need. A good data dictionary with synonyms and comments will make it easier for you (and users) to determine exactly which tables you need for the query. It is also critical that tables be given names that accurately reflect their content and purpose.

One hint in choosing tables is to start with the tables containing the columns listed in the first two questions (output and criteria). Next decide whether other tables might be needed to serve as intermediaries to connect these tables.

## How Are the Tables Joined?

In a relational database, tables are connected by data in similar columns. For instance, as shown in Figure 2.3, a Sales table has a CustomerID column. Corresponding data is stored in the Customer table, which also has a CustomerID column. In many cases matching columns in the tables will have the same name (e.g., CustomerID) and this question is easy to answer. The join performs a matching or lookup for the rows. You can think of the result as one giant table and use any of the columns from any of the joined tables. Note that columns are not required to have the same name, so you sometimes have to think a little more carefully. For example, an Order table might have a column for SalesPerson, which is designed to match the EmployeeID key in an Employee table.

Joining tables is usually straightforward as long as the database design is sound. In fact, the query system will automatically use the design to join any tables whenever possible. However, two problems can arise in practice: (1) You should verify that all tables are joined, and (2) Double-check any tables with multiple join conditions.



**Figure 2.8**

Loops with joins usually cause problems. This sample query would return customers **ONLY** if they live in the same city as the employee who placed the order. Delete the connection from Employee and City to solve the problem.

Technically, it is legal to use tables without adding a join condition. However, when no join condition is explicitly specified, the DBMS creates a **cross join** or Cartesian product between the tables. A cross join matches every row in the first table to every other row in the second table. For example, if both tables have 10 rows, the resulting cross join yields  $10 \times 10 = 100$  rows of data. If the tables each have 1,000 rows, the resulting join has one million rows! A cross join will seriously degrade performance on any DBMS, so be sure to specify a join condition for every table.

Sometimes table designs have multiple relationship connections between tables. For example, Figure 2.8 shows that the Rolling Thunder Bicycles database joins Customer to City and City to Employee. If a query is built that includes the tables: Customer, City, Bicycle, and Employee; the query builder will automatically include a join relationship between Customer and City as well as a join between Employee and City. Including both joins causes a problem—only data that meets both conditions will be displayed. In the Customer/Employee example, Customers will be returned only if they live in the same city as the employee who sold the bicycle. This query is rarely going to be useful. The solution is to remove the join between Employee and City.

## Query Basics

**How do you create a basic query?** It is best to begin with relatively easy queries. This section presents queries that involve a single table to show the basics of creating a query. Then it covers details on constraints, followed by a discussion on computations and aggregations. Groups and subtotals are then explained.

Figure 2.9 presents several business questions that might arise at the Rolling Thunder Bicycles company. Most of the questions are relatively easy to answer. With a small enough number of rows, it might be possible to hand-search the Bicycle table to find answers. However, the point of this section is to start with relatively easy queries to focus on the basics of creating queries and entering conditions.

- Which bicycles were ordered between 12/1/2008 and 12/15/2008?
- Which race bicycles in 2008 were larger than 61 cm?
- Which race bicycles in 2007 had a list price over 7000?
- In December 2008, which mountain or mountain full suspension bicycles sold for more than 5500?
- What is the total value of all bikes sold in December 2008?
- What is the total value of items on purchase order 101?
- How many Race bikes were sold in November 2008?
- What is the number of bicycles sold of each model type in November 2008?
- Who is the best sales person?
- List the CustomerID of everyone who bought a bicycle on December 1, 2008.
- List Names and Phone numbers for everyone who purchased a bike on 01-DEC-2008.
- List customers from Miami, FL who purchased bicycles in December 2008.
- Which model types were not sold in December 2008?

### Figure 2.9

Sample questions for Rolling Thunder Bicycles. The essence of building a query is to convert the business question into the structure required by the DBMS.

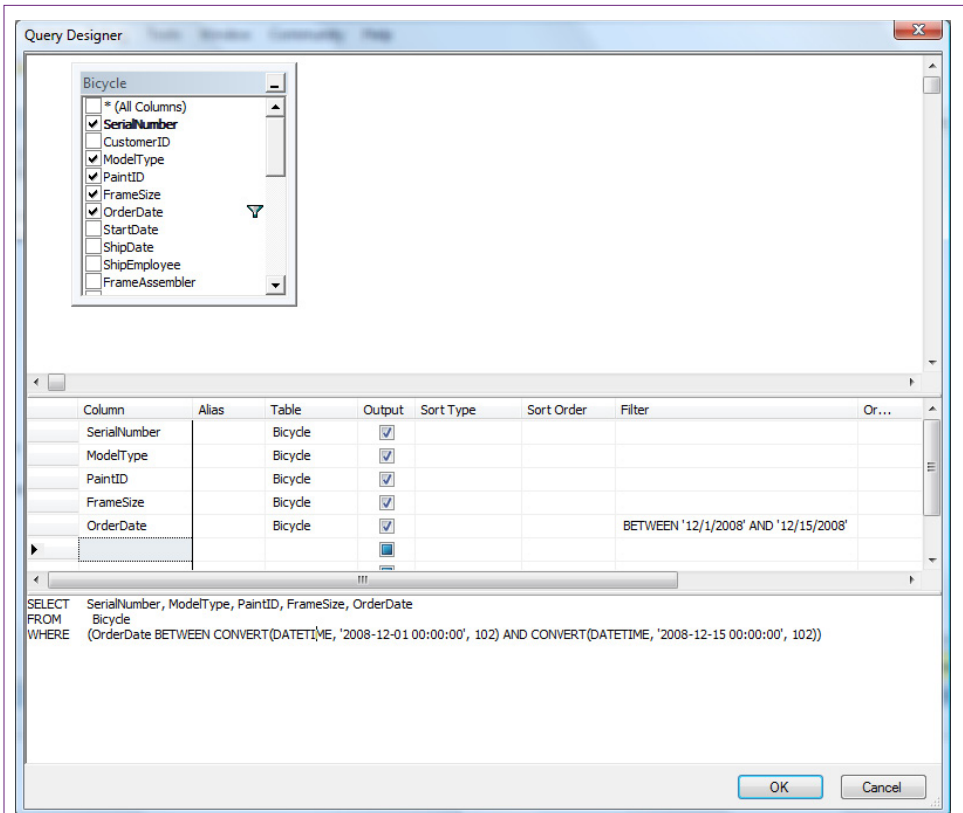
The foundation of queries is that you want to see only some of the columns from a table and that you want to restrict the output to a set of rows that match some criteria. For example, in the first query (animals with yellow color), you might want to see the AnimalID, Category, Breed, and their Color. Instead of listing every animal in the table, you want to restrict the list to just those with a yellow color.

As with most tools today, SQL Server has a query editor to help build queries visually. Ultimately, queries are written in SQL. SQL Server shows the SQL statement in the editor and you can switch back and forth between the two. In many cases, the SQL is easier to read, but the visual editor can be used to create queries with less typing. The query editor can be run through the SQL Server Management Studio. Chapter 3 also shows how the query editor can be used from within the analysis services. Start the Management Studio and log in. Expand the list of databases and right-click the RT database to select the New Query menu item. If the “New Query” button is used on the main menu, you must always start a query with the line: *USE RT;* to ensure the proper database is used. By default, the query editor is set for SQL. Right click and choose the “Design query in Editor” menu option to open the visual designer.

### Single Tables

The first query to consider is: *Which bicycles were ordered between 12/1/2008 and 12/15/2008?* Figure 2.10 shows the design editor and the SQL. The two methods utilize the same underlying structure. The designer approach saves some typing, but eventually you need to be able to write the SQL statements. If you write down the SQL keywords, you can fill in the blanks—similar to the way you fill in the designer grid.

The designer will ask you to choose the tables involved. This question involves only one table: Bicycles. You know that because all of the data you want to see and the constraint are based on columns in the Bicycle table. With the table displayed, you can now choose which columns you want to see in the output. The



**Figure 2.10**

Sample query shown in QBE and SQL. Since there is only one table, only three questions need to be answered: What tables? What conditions? What do you want to see?

business question is a little vague, so select SerialNumber, ModelType, PaintId, FrameSize, and OrderDate.

The next step is to enter the criteria that you already know. In this example, you are looking for bicycles ordered between two specific dates. Date conditions can be entered with standard comparison operators (<, >, >=, and so on). However, dates are usually given as a range and ranges are easiest to enter with a BETWEEN condition. On the same row as the order date, scroll to the right, in the filter column enter: BETWEEN '12/1/2008' AND '12/15/2008'. Dates must be enclosed in single parentheses and entered in the data format set for the local computer. Dates can also be entered as '01-DEC-2008' to avoid confusion between month and day numbers. The SQL statement uses the CONVERT function to explicitly define the date format. Click the OK button to close the designer. Click the "Execute" button to run the query and see the bicycles that match the date condition.

The four basic questions are answered by filling out blanks on the query design grid. (1) The output to be displayed is placed as a field on the grid. (2) The constraints are entered as criteria or conditions under the appropriate fields. (3) The tables involved are displayed at the top (and often under each field name). (4) The

SELECT	columns	What do you want to see?
FROM	tables	What tables are involved?
JOIN	conditions	How are the tables joined?
WHERE	criteria	What are the constraints?

**Figure 2.11**

The basic SQL SELECT command matches the four questions you need to create a query. The uppercase letters are used in this text to highlight the SQL keywords. They can also be typed in lowercase.

table joins are shown as connecting lines among the tables. The one drawback to query design systems is that you have to answer the most difficult question first: Identifying the tables involved. The query design system uses the table list to provide a list of the columns you can choose. Keep in mind that you can always add more tables as you work on the problem.

## Introduction to SQL

SQL is a powerful query language. However, unlike the design editor, you generally have to type in the entire statement. Perhaps the greatest strength of SQL is that it is a standard that most vendors of DBMS software support. Hence, once you learn the base language, you will be able to create queries on all of the major systems in use today. Note that some people pronounce SQL as “sequel,” partly treating it as a descendant of vendor’s early DBMS called quel. Also, “Sequel” is easier to say than “ess-cue-el.”

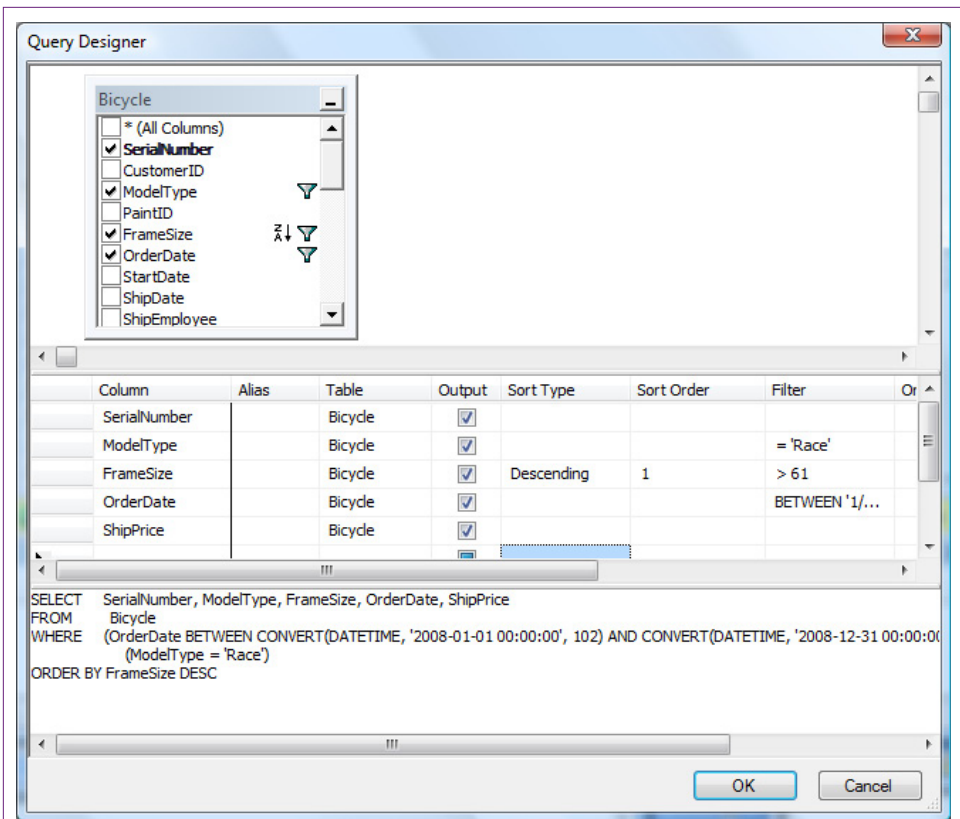
The most commonly used command in SQL is the SELECT statement, which is used to retrieve data from tables. A simple version of the command is shown in Figure 2.11, which contains the four basic parts: **SELECT**, **FROM**, **JOIN**, and **WHERE**. These parts match the basic questions needed by every query. In the example in Figure 2.11, notice the similarity between the editor and SQL approaches. The four basic questions are answered by entering items after each of the four main keywords. When you write SQL statements, it is best to write down the keywords and then fill in the blanks. You can start by listing the columns you want to see as output, then write the constraints in the WHERE clause. By looking at the columns you used, it is straightforward to identify the tables involved. You can use the class diagram to understand how the tables are joined.

## Sorting the Output

Database systems treat tables as collections of data. For efficiency the DBMS is free to store the table data in any manner or any order that it chooses. Yet in most cases you will want to display the results of a query in a particular order. The SQL **ORDER BY** clause is an easy and fast means to display the output in any order you choose. As shown in Figure 2.12, simply list the columns you want to sort. The default is ascending (A to Z or low to high with numbers). Add the phrase **DESC** (for descending) after a column to sort from high to low. In QBE you select the sort order on the QBE grid.

In some cases you will want to sort columns that do not contain unique data. For example, many customers will have the same last name (such as Smith). In





**Figure 2.12**

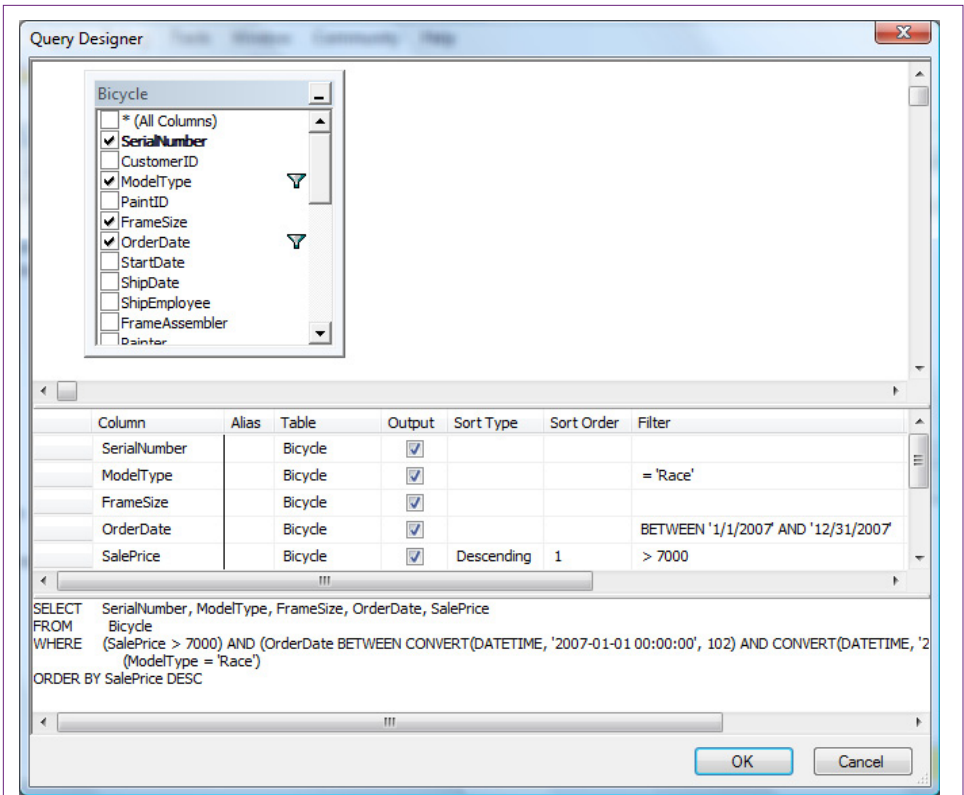
The ORDER BY clause sorts the output rows. The default is to sort in ascending order, adding the keyword DESC after a column name results in a descending sort. When columns like Category contain duplicate data, use a second column (e.g., Breed) to sort the rows within each category.

these cases, a secondary sort column is usually added. Names are generally sorted by LastName and then FirstName. So all customers with the last name Smith will be subsorted by the first name. This additional column is only used if the rows in the prior column contain identical data. In SQL, simply listed the columns left to right in the order to be sorted, such as ORDER BY LastName, FirstName. In the design editor, the priority is given by an index in the Sort Order column..

## Criteria

In most questions, identifying the output columns and the tables is straightforward. If there are hundreds of tables, it might take a while to decide exactly which tables and columns are needed, but it is just an issue of perseverance. On the other hand, identifying constraints and specifying them correctly can be more challenging. More importantly if you make a mistake on a constraint, you will still get a result. The problem is that it will not be the answer to the question you asked—and it is often difficult to see that you made a mistake.

The primary concept of constraints is based on **Boolean algebra**, which you learned in mathematics. In practice, the term simply means that various conditions



**Figure 2.13**

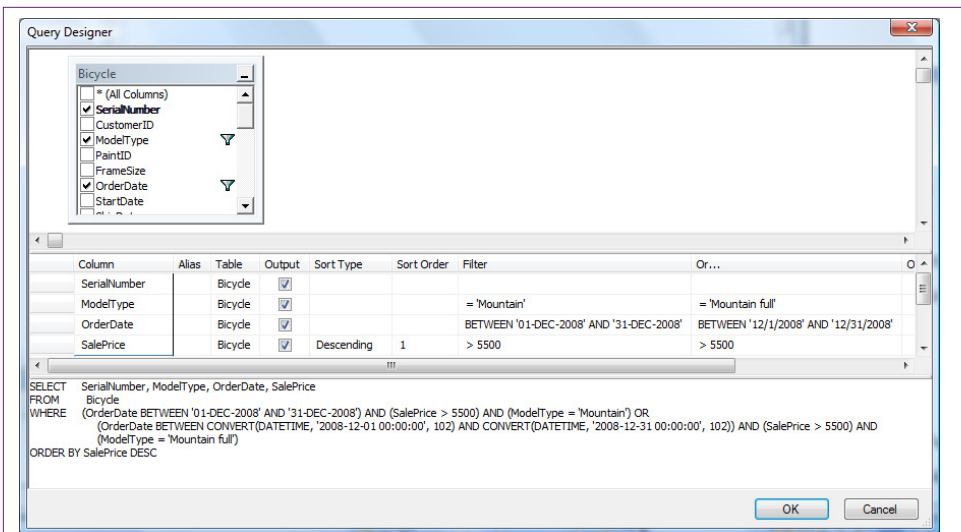
Criteria with AND connectors. Which race bicycles sold for more than 7000 in 2007?

are connected with AND and OR clauses. Sometimes you will also use a **NOT** statement, which negates or reverses the truth of the statement that follows it. For example, NOT (ModelType = 'Race') means you are interested in all models except Race.

Consider the example in Figure 2.13. The first step is to note that three conditions define the business question: date, model type, and price. The second step is to recognize that all of these conditions need to be true at the same time, so they are connected by AND. As the database system examines each row, it evaluates all three clauses. If any one clause is false, the row is skipped.

Notice that the SQL statement is straightforward—just write the three conditions and connect them with an AND clause. The designer is a little trickier. Every condition listed in the same filter column is connected with an AND clause. Conditions in different filter columns are joined with an OR clause. You have to be careful creating (and reading) designer statements, particularly when there are many different criteria columns.

Consider an example with an “OR” connector. In December 2008, which mountain or mountain full suspension bicycles sold for more than 5500? Figure 2.14 shows one way to build the query in the designer. The two ModelType conditions can be written in separate filter columns which creates the OR condition. However, with this approach, the date and price conditions have to be duplicated



**Figure 2.14**

Criteria with OR connectors. In December 2008, which mountain or mountain full suspension bicycles sold for more than 5500? Or conditions (Mountain or Mountain full) can be written in different filter columns. But the other conditions have to be copied along as well.

and listed in both filter columns. Each filter column is treated as a separate list of conditions. All conditions within that column are connected with AND statements. If you leave out the date and price conditions on the second filter column, it will match all full suspension bicycles regardless of date or price. In all cases, test the query! Run it and check the rows to ensure they meet all of the business conditions.

Figure 2.15 shows a way to simplify the query using SQL. Enter the ModelType condition with the OR connector in parentheses. Then add the date and SalePrice conditions and insert the AND connectors. By using parentheses to isolate the OR conditions, the other statements can be listed one time. Switch to the design editor to see how the editor handles this change. In the main query editor, highlight the entire query and then switch to the design editor. The ModelType statement with the OR connector is listed completely in a single filter cell. For

**Figure 2.15**

SQL criteria with OR connectors. Parentheses are used to isolate the OR condition for Mountain or Mountain full model types. Then the AND conditions are listed only once. Convert to design view to see how it is handled in the editor.

```

SELECT  SerialNumber, ModelType, OrderDate, SalePrice
FROM    Bicycle
WHERE   (OrderDate BETWEEN '01-DEC-2008' AND '31-DEC-2008')
        AND (SalePrice >5500)
        AND (ModelType='Mountain' OR ModelType='Mountain full')
ORDER BY SalePrice DESC;
  
```

Comparisons	Examples
Operators	<, =, >, <>, >=, BETWEEN, LIKE, IN
Numbers	SalePrice > 6000
Test	
Simple Match one character Match many	LastName > 'Jones' License LIKE 'A__82_' LastName LIKE 'S%'
Dates	SaleDate BETWEEN '01-DEC-2008' AND '15-DEC-2008'
Missing Data	LastName IS NULL
Negation	FirstName IS NOT NULL NT (ModelType='Race')
Sets	ModelType IN ('Race', 'Road', 'Tour')

**Figure 2.16**

SQL criteria with OR connectors. Parentheses are used to isolate the OR condition for Mountain or Mountain full model types. Then the AND conditions are listed only once. Convert to design view to see how it is handled in the editor.

complex statements, it is generally best to rely on SQL to create the conditions. The conditions are easier to create and easier to read in SQL. However, in all cases, build the query in steps and test it at each step.

Actually, the mountain bike model types in RT were specifically named to support another way to include both the Mountain or Mountain full models without the hassle of using an OR clause. Notice that both begin with the word “Mountain.” Consequently, whenever both types are desired, the condition can be written: ModelType LIKE 'Mountain%'.

### Useful WHERE Clauses

Most database systems provide the comparison operators displayed in Figure 2.16. Standard numeric data can be compared with equality and inequality operators. Text comparisons are usually made with the LIKE operator for pattern matching. For all text criteria, you need to know if the system uses case-sensitive comparisons. By default, Microsoft SQL Server is not case-sensitive, so you can type the pattern or condition using any case. If case-sensitivity is desired, the default collation method can be changed to one that is case-sensitive. If you do not know which case was used, you can use the UPPER function to convert to upper case and then write the pattern using capital letters.

The **BETWEEN** clause is not required, but it saves some typing and makes some conditions a little clearer. The clause (SaleDate BETWEEN '01-DEC-2008' AND '15-DEC-2008') is equivalent to (SaleDate >= '01-DEC-2008' AND SaleDate <= '15-DEC-2008'). The date syntax shown here can be used on most database systems. Some systems allow you to use shorter formats, but on others, you will have to specify a conversion format. These conversion functions are not standard.

```
SaleItem(SaleID, ItemID, SalePrice, Quantity)
SELECT SaleID, ItemID, SalePrice, Quantity,
       SalePrice*Quantity As Extended
FROM SaleItem;
```

SaleID	ItemID	Price	Quantity	Extended
24	25	2.70	3	8.10
24	26	5.40	2	10.80
24	27	31.50	1	31.50

**Figure 2.17**

Computations. Basic computations (+ - \* /) can be performed on numeric data in a query. The new display column should be given a meaningful name.

Another useful condition is to test for missing data with the NULL comparison. Two common forms are IS NULL and IS NOT NULL. Be careful—the statement (City = NULL) will not work with most systems, because NULL is not a value. You must use (City IS NULL) instead. Unfortunately, conditions with the equality sign are not flagged as errors. The query will run—it just will never match anything.

## Computations

**What types of computations can be performed in SQL?** The statistical computations used for data mining are handled later by the business intelligence tools which use a programming language. However, some simple computations can be performed directly within SQL. Queries are used for two types of computations: aggregations and simple arithmetic on a row-by-row basis. Sometimes the two types of calculations are combined. Consider the row-by-row computations first.

### Basic Arithmetic Operators

SQL and the designer can both be used to perform basic computations on each row of data. This technique can be used to automate basic tasks and to reduce the amount of data storage. Consider a common order or sales form. As Figure 2.17 shows, the basic tables would include a list of items purchased: SaleItem(SaleID, ItemID, SalePrice, Quantity). In most situations you would need to multiply SalePrice by Quantity to get the total value for each item ordered. Because this computation is well defined (without any unusual conditions), there is no point in storing the result—it can be recomputed whenever it is needed. Simply build a query and add one more column. The new column uses elementary algebra and lists a name: SalePrice\*Quantity AS Extended. Remember that the computations are performed for each row in the query.

Most systems provide additional mathematical functions. For example, basic mathematical functions such as absolute value, logarithms, and trigonometric functions are usually available. Although these functions provide extended capabilities, always remember that they can operate only on data stored in one row of a table or query at a time.

Sum	Avg	Min	Max	Count
StDev	StDevP	Var	VarP	

**Figure 2.18**

SQL Server Aggregation functions. Sum, Avg, and Count are most common. The standard format is Sum( column ) but it is also possible to use Count (DISTINCT column) to examine only the unique values.

## Aggregation

Databases for business often require the computation of totals and subtotals. Note that the data mining approach in Chapter 3 is a better way to examine many subtotals, but sometimes SQL subtotals are useful to configure data for other analyses. Hence, query systems provide functions for **aggregation** of data. The common functions listed in Figure 2.18 can operate across several rows of data and return one value. The most commonly used functions are Sum, Avg, and Count which are similar to those available in spreadsheets. SQL Server also supports the DISTINCT clause to examine only the unique values; such as using Count(DISTINCT ModelType) to count each model type only one time. Be careful when using Sum and Count. Count simply counts the number of rows regardless of the data value. Sum adds the values selected. Sum is useful for monetary or quantity columns, Count is often used for key or ID columns.

With SQL, the functions are simply added as part of the SELECT statement. Figure 2.19 shows the SQL command to compute the total value of bicycles sold in December 2008. These basic queries are often easier to write in SQL than in the designer.

The designer can be used to build queries with totals. As shown in Figure 2.20, the first trick is to right-click the table area and choose the option to “Add Group By” to add the column with the Group By options. Use the drop-down list to select Sum for the SalePrice data. The second big trick is to select the WHERE option for the OrderDate. By default, the option is GROUP BY, which means that the totals will be computed for each item in the column (day). To compute just a single total across all days, the WHERE option switches the condition to the WHERE clause of the SQL statement. This difference is explained in the next section.

The **row-by-row calculations** can also be combined with an aggregate function. The example in Figure 2.21 asks for the total value of a particular purchase

**Figure 2.19**

Aggregation functions. SQL statement to obtain total sales of bicycles in December 2008.

```
SELECT SUM(SalePrice) As TotalSales
FROM Bicycle
WHERE OrderDate BETWEEN '01-DEC-2008' AND '31-DEC-2008';
```

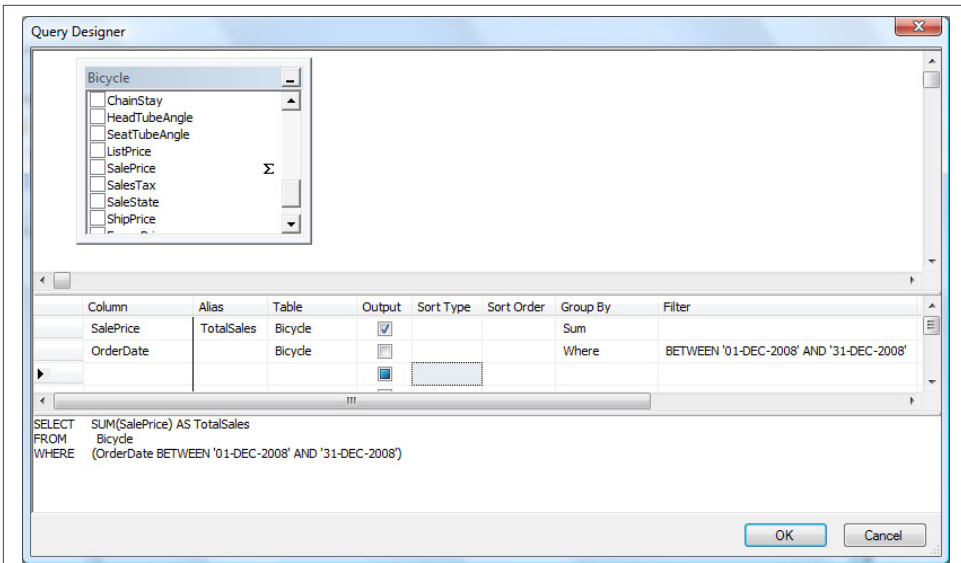
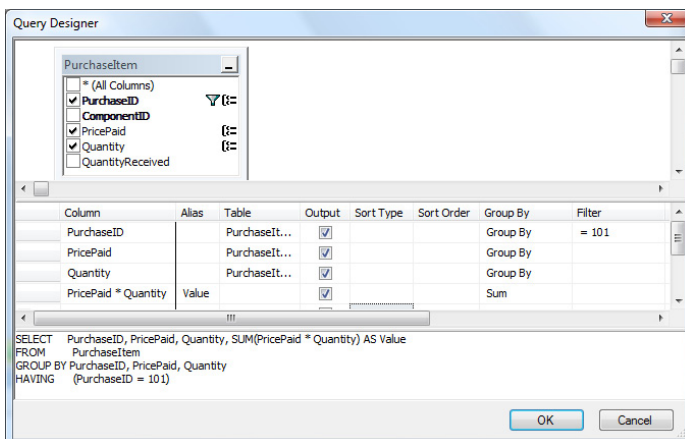


Figure 2.20

Aggregation functions in the designer. Right-click the table area and choose “Add Group By” to see the Group By column. Select the Sum option for SalePrice and the WHERE option for the date.

Figure 2.21

Multiply values and then compute totals. What is the total value of items on purchase order 101?



```
SELECT PurchaseID,
PricePaid, Quantity,
SUM(PricePaid * Quantity)
AS Value
FROM PurchaseItem
WHERE (PurchaseID = 101)
GROUP BY PurchaseID,
PricePaid, Quantity
```



Task	SQL Server
Strings	
Concatenation	FName + ' ' + LName
Length	Length(LName)
Upper case	Upper(LName)
Lower case	Lower(LName)
Partial string	Substring(LName,2,3)
Dates	
Today	GetDate()
Month	DateName(month,
Day	myDate), Month(myDate)
Year	DatePart(day, myDate)
Date arithmetic	DatePart(year, myDate), Year(myDate) DateAdd DateDiff
Formatting	Str(item, length, decimal) Cast, Convert
Numbers	
Math functions	Cos, Sin, Tan, Sqrt
Exponentiation	Power(2, 3)
Aggregation	Min, Max, Sum, Count,
Statistics	Avg, StDev, Var, LinRegSlope, Correlation

**Figure 2.22**

SQL functions. Hundreds of internal functions exist in SQL Server, but these are the most common.

order. To get total value, the database must first calculate Quantity \* PricePaid for each row and then get the total of that column. This example computes the total for just one specific order (101).

There is one important restriction to remember with aggregation. The query can display either the details or the totals—not both at the same time. This constraint can sometimes be avoided by using complex queries, but it is best to decide in advance if the SQL should display details or totals, then use other tools to manipulate the results.

Note that several aggregate functions can be computed at the same time. For example, the Sum, Average, and Count can be displayed at the same time: `SELECT Sum(Quantity), Avg(Quantity), Count(Quantity) From PurchaseItem`. In fact, if you need all three values, you should compute them at one time. Consider what happens if you have a table with a million rows of data. If you write three separate queries, the DBMS has to make three passes through the data. By combining the computations in one query, you cut the total query time to one-third. With huge tables or complex systems, these minor changes in a query can make the difference between a successful application and one that takes days to run.

## Functions

The `SELECT` command also supports functions that perform calculations on the data. These calculations include numeric forms such as the trigonometric func-

```
SELECT Count(SerialNumber) As Nbikes
FROM Bicycle
WHERE ModelType='Race'
      AND OrderDate BETWEEN '01-NOV-2008' AND '30-NOV-2008'
```

### Figure 2.23

SQL subtotal introduction. How many Race bikes were sold in November 2008? Use SQL and not the designer.

tions, string function such as concatenating two strings, date arithmetic functions, and formatting functions to control the display of the data. Unfortunately, these functions are not standardized, so each DBMS vendor has different function names and different capabilities. Figure 2.22 lists some of the common functions used in SQL Server.

String operations are relatively useful. Concatenation is one of the more powerful functions, because it combines data from multiple columns into a single display field. It is particularly useful to combine a person's last and first names. Other common string functions convert the data to all lowercase or all uppercase characters. The length function counts the number of characters in the string column. A substring function is used to return a selected portion of a string. For example, you might choose to display only the first 20 characters of a long title.

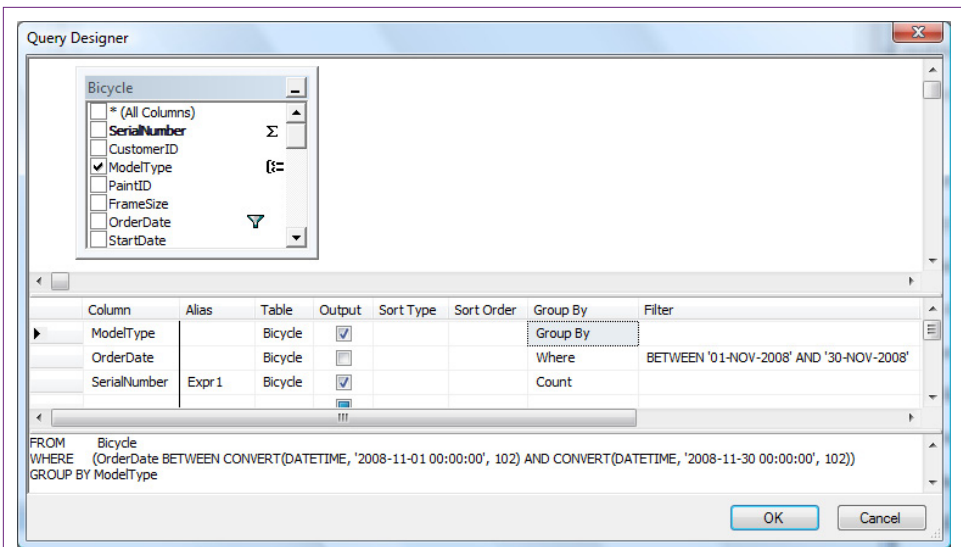
The powerful date functions are often used in business applications. Date columns can be subtracted to obtain the number of days between two dates. Additional functions exist to get the current date and time or to extract the month, day, or year parts of a date column. Date arithmetic functions can be used to add (or subtract) months, weeks, or years to a date. The Convert function can be used to specify the format of a date. It is also used for other types of data, such as setting a fixed number of decimal points or displaying a currency sign.

It is also possible to define custom functions. SQL Server uses T-SQL to enable you to create programming code in SQL that can define relatively complex operations on data and return a new function value. For even more complex calculations, it is possible to create new functions using a .NET procedural language such as C#. These tasks are covered in database textbooks and typically handled by database programmers instead of managers.

## Subtotals and GROUP BY

**How are subtotals computed?** The previous section hinted at the use of totals but one of the most powerful features of SQL was left for this section: subtotals. Many business questions involve the use of subtotals: Who are the best customers? Which employee sold the most bicycles in November? Which model type was the most popular in 2008? All of these questions require totals or counts for each value: The total sales for each customer, total sales for each employee, and count of bicycles for each model type.

To illustrate, consider the question: How many Race bikes were sold in November 2008? As shown in Figure 2.23, the SQL is straightforward—simply use the Count function and a WHERE clause for ModelType and OrderDate. To test the function, stick with the SQL and avoid the designer.



**Figure 2.24**

Subtotal with designer. How many bikes of each model type were sold in November 2008?

Before building the query in the designer, it is useful to generalize the question to require the use of subtotals: What is the number of bicycles sold of each model type in November 2008? The only difference is that this query does not require the ModelType to be constrained. Instead, it requires the DBMS to find each ModelType, restrict the date to the desired range, and count the number of each type of model. Although it sounds complex, SQL handles the setup easily.

Build a new query using the designer. Use the Bicycle table and add only the columns: ModelType, OrderDate, and SerialNumber. Figure 2.24 shows the setup. Add the Group By column and leave Group By as the mode for the ModelType because the business question calls for a value “for each” model type. Enter the date filter and set the Group mode to Where. If this mode is left on Group By, the

**Figure 2.25**

Subtotal in SQL with results.

```
SELECT  ModelType, COUNT(SerialNumber) AS NBikes
FROM    Bicycle
WHERE   (OrderDate
        BETWEEN CONVERT(DATETIME, '2008-11-01 00:00:00', 102)
        AND CONVERT(DATETIME, '2008-11-30 00:00:00', 102))
GROUP BY ModelType;
```

```
Mountain      13
Mountain full 57
Race          52
Road          47
Tour         13
```

```
SELECT ModelType, COUNT(SerialNumber) AS NBikes
FROM Bicycle
WHERE (OrderDate
      BETWEEN CONVERT(DATETIME, '2008-11-01 00:00:00', 102)
      AND CONVERT(DATETIME, '2008-11-30 00:00:00', 102))
GROUP BY ModelType
HAVING Count(SerialNumber) > 15;
```

### Figure 2.26

Limiting the output with a HAVING clause. The GROUP BY clause with the Count function provides a count of the number of animals in each category. The HAVING clause restricts the output to only those categories having a count greater than 15.

query will return values for each date—which is not called for in the business question. Finally, set Count for the SerialNumber. Almost any column could be counted since the DBMS simply counts the number of rows returned, but SerialNumber is best because it indicates that bicycles are being counted and because it is the primary key it avoids potential problems with duplicate or missing values.

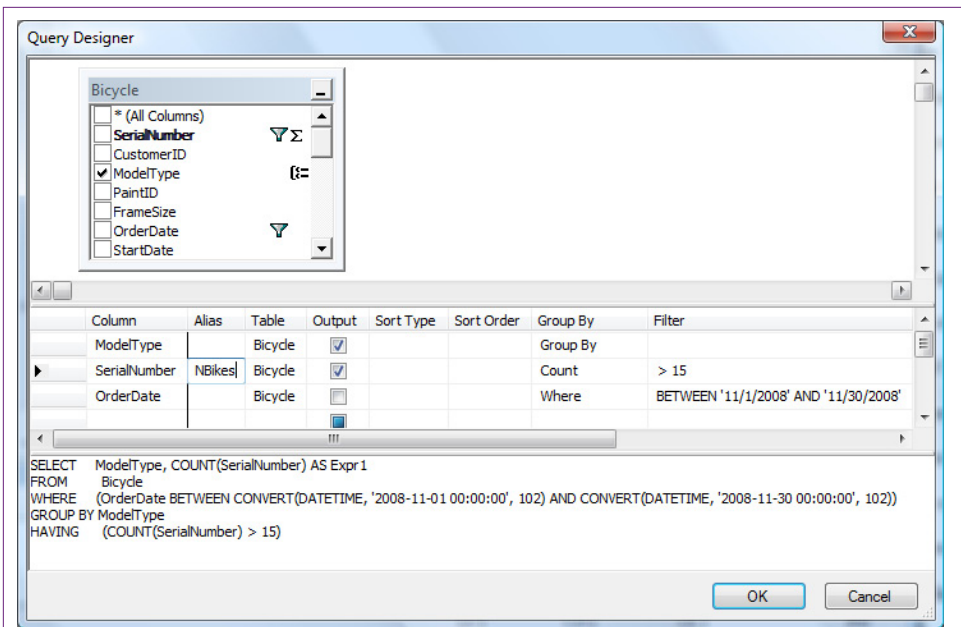
Figure 2.25 shows the SQL statement for the subtotal computation along with the results. Your results might differ slightly because the data in the database could have been altered. The alias (NBikes) has been added to provide a more descriptive title for the result column instead of Expr1. To obtain subtotals, the only new step is to add the GROUP BY clause. The **GROUP BY** statement can be used only with one of the aggregate functions (Sum, Avg, Count, and so on). With the GROUP BY statement, the DBMS looks at all the data, finds the unique items in the group, and then performs the aggregate function for each item in the group.

By default, the output will generally be sorted by the group items. However, for business questions, it is common to sort (ORDER BY) based on the computation. Be careful about adding multiple columns to the GROUP BY clause. The subtotals will be computed for each distinct item in the entire GROUP BY clause. Including additional columns (particularly date) might lead to a more detailed breakdown than desired.

### Conditions on Totals (HAVING)

The GROUP BY clause is powerful and provides useful information for making decisions. In cases involving many groups, you might want to restrict the output list, particularly when some of the groups are relatively minor. For example, the simple Mountain and Tour types are not ordered much. In analyzing sales the managers might prefer to focus on the top-selling categories.

One way to reduce the amount of data displayed is to add the **HAVING** clause. The HAVING clause is a condition that applies to the GROUP BY output. In the example presented in Figure 2.26, the managers want to skip any model type category that has fewer than 15 animals. Notice that the SQL statement simply adds one line. The same condition can be added to the criteria grid in the designer query. The HAVING clause is powerful and works much like a WHERE statement. Just be sure that the conditions you impose apply to the computations indicated by the GROUP BY clause.



**Figure 2.27**

WHERE versus HAVING. Count the bicycles sold in November 2008 by model type and show only those where the result is greater than 15. It is hard to see the difference between these conditions in the designer. SQL is often easier to see that WHERE conditions are applied before the GROUP BY and HAVING is applied to the results.

## WHERE versus HAVING

At first glance, WHERE and HAVING look very similar, and choosing the proper clause can be confusing. Yet it is crucial that you understand the difference. If you make a mistake, the DBMS will give you an answer, but it will not be the answer to the question you want. The key is that the WHERE statement applies to every single row in the original table. The HAVING statement applies only to the subtotal output from a GROUP BY query. To add to the confusion, you can even combine WHERE and HAVING clauses in a single query—because you might want to look at only some rows of data and then limit the display on the subtotals.

Figure 2.27 shows the query in the designer that includes both the WHERE and HAVING conditions. Looking at the layout, it is difficult to determine the difference between the two conditions. The WHERE entry in the GROUP BY column is required to shift a condition to the WHERE statement instead of the HAVING statement; and this change can be hard to remember. In most cases, you should check the SQL to ensure that the conditions are structured correctly. The WHERE clause is used to filter rows before any calculations take place. The HAVING condition simply limits the output of the GROUP BY results. In general, it is best to use WHERE clauses whenever possible because they immediately cut the number of rows to be investigated. The SQL Server query processor has a decent optimizer so making a mistake will not seriously hurt performance, but to understand the difference, it is best to think of the WHERE clause as the initial filter.

```
Try:
SELECT EmployeeID, Max(SalePrice)
FROM Bicycle
GROUP BY EmployeeID;
```

Which returns a list of employees and the most expensive bicycle sold by each employee. It does not return the totals.

```
Try:
SELECT EmployeeID, Max(Sum(SalePrice))
FROM Bicycle
GROUP BY EmployeeID;
```

Which does not even run.

```
Answer:
SELECT EmployeeID, Sum(SalePrice) AS Revenue
FROM Bicycle
GROUP BY EmployeeID
ORDER BY Sum(SalePrice) DESC
```

### Figure 2.28

Find the best salesperson. Several tempting methods do not work. The solution is to compute the total sales for each employee using the GROUP BY statement and use ORDER BY to sort the results. To display less than the entire list, use a HAVING clause or enter TOP 5 just after the SELECT clause.

## The Best and the Worst

Think about the business question, *Who is the best salesperson?* How would you build a SQL statement to answer that question? To begin, you have to decide if “best” is measured in quantity, revenue, or profit. For now, simply use revenue. A common temptation is to write a query similar to `SELECT EmployeeID, Max(SalePrice) FROM Bicycle GROUP BY EmployeeID`. This query will run. It will return a list of employees and the most expensive bicycle sold by each employee; but it will not sum the prices. A step closer might be `SELECT EmployeeID, Max(Sum(SalePrice)) FROM Bicycle GROUP BY EmployeeID`. But this query will not run because the database cannot compute the maximum until after it has computed the sum. So, the best answer is to use: `SELECT EmployeeID, Sum(SalePrice) AS Revenue FROM Bicycle GROUP BY EmployeeID ORDER BY Sum(SalePrice) DESC`. This query will compute the total sale prices for each employee and display the result in descending order—the best salespeople will be at the top of the list.

The advantage to this approach is that it shows other rows that might be close to the “best” entry, which is information that might be valuable to the decision maker. The one drawback to this approach is that it returns the complete list of items sold. Generally, most businesspeople will want to see more than just the top or bottom item, so it is not a serious drawback—unless the list is too long. In that case, you can use the HAVING command to reduce the length of the list. SQL Server also supports the TOP command to restrict a list without knowing anything about the data. Simply add TOP 5 to the SELECT statement: `SELECT TOP 5 EmployeeID, Sum(SalePrice As Revenue)...` Of course, the number 5 is arbitrary and any value can be used. In practice, the Min and Max functions are rarely used.

SELECT CustomerID FROM Bicycle WHERE OrderDate = '01-DEC-2008';	19114 31202 31335 31651 32631 32687
---	--

**Figure 2.29**

List the CustomerID of everyone who bought a bicycle on December 1, 2008. Most people would prefer to see the names and phone numbers of the customers—those attributes are in the Customer table.

## Multiple Tables

**How do you use multiple tables in a query?** All the examples so far have used a single table—to keep the discussion centered on the specific topics. In practice, however, you often need to combine data from several tables. In fact, the strength of a DBMS is its ability to combine data from multiple tables.

The essence of a relational database is to split data into separate pieces that are linked through the primary keys. This approach is efficient at storing data but it requires a query system to join the tables back together. From a query design perspective, SQL has a straightforward method of connecting tables. For example, the Bicycle table contains just the CustomerID to identify the specific customer. Most people would prefer to see the customer name and other attributes. This additional data is stored in the Customer table—along with the CustomerID. The objective is to take the CustomerID from the Sale table and look up the matching data in the Customer table.

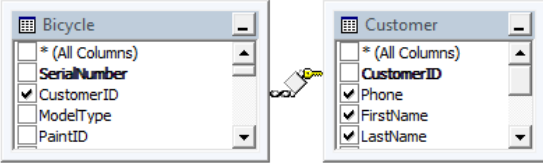
### Joining Tables

With modern query languages, combining data from multiple tables is straightforward. You simply specify which tables are involved and how the tables are connected. The query designer is particularly easy to use for this process. To understand the process, first consider the business question posed in Figure 2.29: list the CustomerID of everyone who bought something on 01-DEC-2008.

Most managers would prefer to see the customer name instead of CustomerID. However, the name is stored in the Customer table because it would be a waste of space to copy all of the attributes to every table that referred to the customer. If you had these tables only as printed reports, you would have to take the CustomerID from the sale reports and find the matching row in the Customer table to get the customer name. Of course, it would be time-consuming to do the matching by hand. The query system can do it easily.

As illustrated in Figure 2.30, the designer approach is somewhat easier than the SQL syntax. However, the concept is the same. First, identify the two tables involved (Bicycle and Customer). In the designer, select the tables from a list and they are displayed at the top of the form with the JOIN drawn as a line. In SQL, enter the table names on the FROM line. Second, specify which columns are matched in each table. In this case match CustomerID in the Bicycle table to the CustomerID in the Customer table. Most of the time the column names will be the same, but they could be different.





```

SELECT  Bicycle.CustomerID, Customer.FirstName,
        Customer.LastName, Customer.Phone
FROM    Bicycle
INNER JOIN Customer
        ON Bicycle.CustomerID = Customer.CustomerID
WHERE   (Bicycle.OrderDate = '01-DEC-2008')

```

CID	FirstName	LastName	Phone
19114	James	Dugan	(405) 015-0612
31202	Michael	Macdonald	(803) 408-1618
31335	Theresa	Raux	(540) 689-8730
31651	Kwok	Lawrence	(850) 136-9460
32631	Joseph	Despirito	(814) 732-0324
32687	B	Wagman	(863) 989-8229

**Figure 2.30**

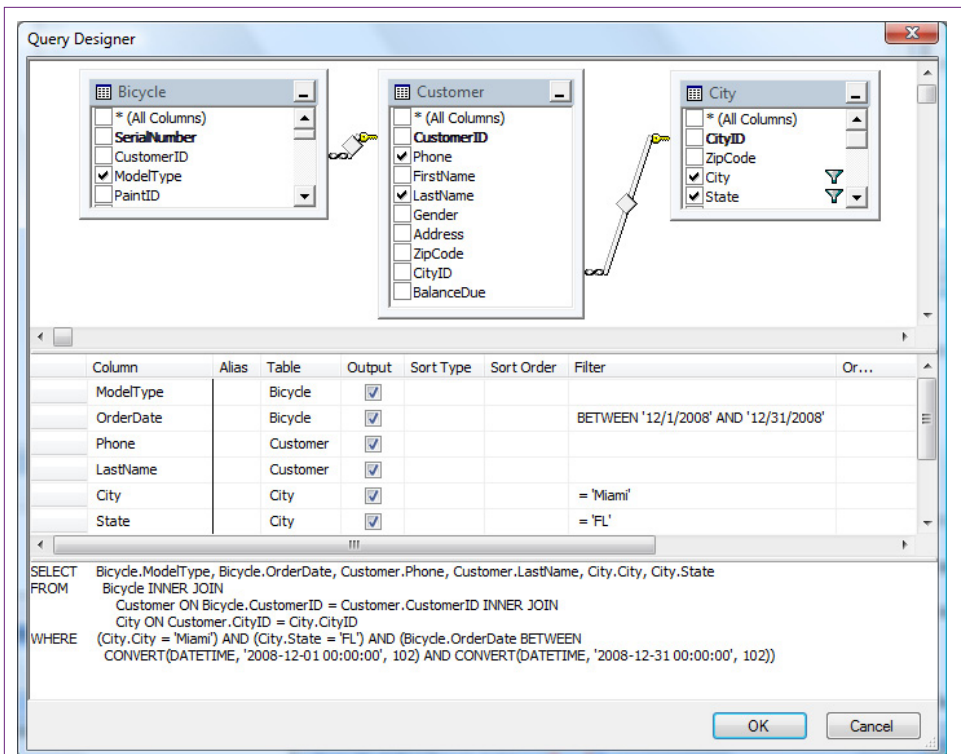
Joining tables causes the rows to be matched based on the columns in the JOIN statement. You can then use data from either table. The business question is, List the names and phone numbers of customers who bought a bicycle on December 1, 2008.

In SQL, tables are connected with the JOIN statement. The syntax for a JOIN is: FROM Bicycle INNER JOIN Customer ON Bicycle.CustomerID = Customer.CustomerID. The order of the tables does not matter. Notice that the concepts for SQL and the designer are the same: List both tables and which columns in the two tables are matched. SQL requires more typing. When multiple tables are involved, it is often easier to handle the joins in the designer. To add more tables in SQL, simply add another INNER JOIN statement with its associated ON clause to specify the columns.

### Identifying Columns in Different Tables

Examine how the columns are specified in the SQL JOIN statement. Because the column CustomerID is used in both tables, it would not make sense to write CustomerID = CustomerID. The DBMS would not know what you meant. To keep track of which column you want, you must also specify the name of the table: Sale.CustomerID. Actually, you can use this syntax anytime you refer to a column. You are required to use the full table.column name only when the same column name is used in more than one table.

SQL Server supports two more levels: the database and schema. The schema is typically a name assigned to a role, and in many cases it is dbo (short for database owner). So, a fully-named table would be: database.schema.table. For example: SELECT \* FROM RT.dbo.ModelType.



**Figure 2.31**

Joining multiple tables. The designer makes it easy to connect multiple tables—simply add them to the list and verify the column connections.

## Joining Many Tables

A query can use data from several different tables. The process is similar regardless of the number of tables. Each table you want to add must be joined to one other table through a data column. If you cannot find a common column, either the table design is wrong or you need to find a third table that contains links to both tables.

Consider the example in Figure 2.31: List customers from Miami, FL who purchased bicycles in December 2008. An important step is to identify the tables needed. For large problems involving several tables, it is best to first list the columns you want to see as output and the ones involved in the constraints. In the example, the name and phone number you want to see are in the Customer table. The city name and state are in the City table, and the OrderDate is in the Bicycle table. Fortunately, all three of these tables are connected to each other and you do not need to search for an intermediate connection table.

When the database contains a large number of tables, complex queries can be challenging to build. You need to be familiar with the tables to determine which tables contain the columns you want to see. For large databases, an entity-relationship diagram (ERD) or a class diagram can show how the tables are connected.

When you first see it, the SQL 92 syntax for joining more than two tables can look confusing. In practice, it is best not to memorize the syntax. When you are

```

SELECT      Bicycle.ModelType, Bicycle.OrderDate, Customer.Phone,
            Customer.LastName, City.City, City.State
FROM  Bicycle
INNER JOIN  Customer
            ON Bicycle.CustomerID = Customer.CustomerID
INNER JOIN  City
            ON Customer.CityID = City.CityID
WHERE (City.City = 'Miami')
      AND (City.State = 'FL')
      AND (Bicycle.OrderDate BETWEEN
            CONVERT(DATETIME, '2008-12-01 00:00:00', 102)
            AND CONVERT(DATETIME, '2008-12-31 00:00:00', 102))

```

**Figure 2.32**

Joining multiple tables. The designer makes it easy to connect multiple tables—simply add them to the list and verify the column connections.

first learning SQL, understanding the concept of the JOIN is far more important than worrying about syntax. Figure 2.32 shows the syntax needed to join three tables. To handle multiple joins use the FROM statement to list the first table and add an INNER JOIN new\_table ON table.column = new\_table.column statement for each table that needs to be added. SQL Server does not require parentheses to specify the order of the joins, but be careful when adding new tables. The column to be connected must already exist in the joined tables. In the Bicycle-Customer-City example, the joins have to be built either as Bicycle-Customer-City or City-Customer-Bicycle. A set of joins from City-Bicycle-Customer will not work because the Bicycle table does not include the CityID.

When in doubt, use the designer. Occasionally, it is necessary to change the joins created automatically in the designer. For example, the designer might include an extra join, such as connecting the Employee table to the City table. Deleting joins is straightforward—select the join and press the Delete key or right-click the join and choose the Remove option. Creating a new join is only a little harder. Select the column in one table (such as CustomerID), drag the column and drop it on top of the matching column in the second table. The direction of the drag is unimportant, but it often helps to expand or adjust the tables in advance so both tables clearly display the columns to be connected.

## Views: Saved Queries

Complex queries are often easier to solve by breaking them into smaller pieces. An initial query can be saved as a **view**, which is simply a saved query. New queries can use views as if they were another table, and the joins are the same. Views are also useful for controlling access to data. Instead of giving a user access to an entire table, a view can be created that retrieves a restricted set of rows and columns. Other workers can use this view without knowing anything about the underlying table. This approach is useful for combining multiple tables into a single view so that the joins are hidden within the view. To create a view that retrieves data, first build the query and test it. Then add one line to the top of the query: CREATE VIEW view\_name AS. Enter a unique and memorable name as

```
CREATE VIEW December2008Sales AS
SELECT  SerialNumber, CustomerID, ModelType, PaintID, FrameSize,
        OrderDate, StartDate, ShipDate, LetterStyleID, StoreID,
        EmployeeID, SalePrice, ListPrice, SalesTax, SaleState, ShipPrice
FROM    Bicycle
WHERE   (OrderDate BETWEEN
        CONVERT(DATETIME, '2008-12-01 00:00:00', 102)
        AND CONVERT(DATETIME, '2008-12-31 00:00:00', 102))
```

### Figure 2.33

Creating a View. Build and test the query—bicycle sales in December 2008. Then add the first line to create the view with a unique name.

the view\_name and run this new command. Instead of displaying the data, the system will create the query with the given name. Figure 2.33 shows how to create a view that lists bicycle sales only for December 2008. This view is needed for the next section.

Saved queries are useful when a problem has multiple parts. Consider the question: Which customers purchased bicycles in both 2007 and 2008? At first glance the question seems easy. You might try a single query:

```
SELECT *
FROM Customer
INNER JOIN Bicycle
    ON Customer.CustomerID=Bicycle.CustomerID
WHERE Year(OrderDate)=2007 AND Year(OrderDate)=2008;
```

This query will run but it will never return any matches because a date can never be both 2007 and 2008 at the same time. Instead, the question has to be split into two queries. SQL supports subqueries that enable the second part to be embedded into a single query, but subqueries are beyond the scope of this book. It is easier to simply create two views—one for each year. Save them as Customers2007 and Customers2008 and set the single appropriate year condition in each query. The list of customers can then be obtained by creating a new query that joins the two views. The JOIN condition establishes the “AND” condition that a customer fall into both lists:

```
SELECT *
FROM Customers2007
INNER JOIN Customers2008
    ON Customers2007.CustomerID=Customers2008.
CustomerID;
```

The key is to recognize when two separate queries are needed. No secret formula exists—you simply have to logically evaluate the business question and decide which conditions can be handled at one time and which ones require separate lists. Just remember that when you save a query as a view to give it a name that explains the data in the view so that it can be recognized later.

```
SELECT ModelType.ModelType
FROM ModelType
LEFT JOIN December2008Sales
  ON ModelType.ModelType=December2008Sales.ModelType
WHERE December2008Sales.SerialNumber Is Null;
```

**Figure 2.34**

Left Join example. Which model types were not sold in December 2008? Left Join is needed because Inner Join would display only model types that were sold.

## LEFT JOIN

You might be wondering why the syntax for the join is `INNER JOIN` instead of just the word “Join.” The difference is important because another form of the Join command exists. An inner join connects two tables by matching only the items in the first table that are equal to the items in the second table. Items in either table that do not match are ignored or dropped from the display. For most queries, this action is appropriate as a simple matching system. However, a special type of business question arises that is difficult to answer with the inner join approach. (It can be handled with subqueries, but these are not covered in this book.) The basic question is: Because the database records things that did happen, how can you find things that did not happen? This question seems strange at first, but consider the simple business question for Rolling Thunder Bicycles: Which model types were **not** sold in December 2008?

The `December2008Sales` view created in the previous section lists all bicycles ordered in December 2008, which are things that did happen. Where does the database store things that did not happen? Nowhere. The solution is to take the list of all model types (`ModelType` table), subtract out the model types that were sold, and the model types that remain are the ones that were not sold. Figure 2.34 shows the query needed to find the result. The `ModelType` table is connected to the `December2008Sales` view using a Left Join. The Left Join specifies that all rows from the left table are to be included in the results—even if no matching data exists in the right table. When an entry in the left, `ModelType`, table has no matching value in the sales table, the join enters a Null value for all columns in the sales table. Hence, the `Is Null` condition returns only those rows in the left table that have no matching values in the right (sales) table. In this case, the result is: Hybrid and Track model types. Figure 2.35 shows some random sample data from the query using the Left Join. The Null values for the Hybrid and Track model types are highlighted in red. All other model types have matching data, so only the two types meet the `Is Null` condition to indicate that they were not sold.

## UNION

Joins are used to select columns from multiple tables or views. On the other hand,, sometimes it is useful to combine rows of data from multiple tables or views. This trick is useful when similar data is stored in separate tables. For example, consider a company that has two divisions (East and West) and keeps a separate employee table for each division: `EmployeesEast` and `EmployeesWest`. Most queries are

ModelType	Serial	CID	ModelType
Mountain full	37774	10433	Mountain full
Mountain full	37486	31386	Mountain full
Mountain	37232	31132	Mountain
Race	38080	31980	Race
Race	38123	32023	Race
Road	38359	32259	Road
Hybrid	Null	Null	Null
Tour	38787	32687	Tour
Track	Null	Null	Null

**Figure 2.35**

Left Join example. Sample random rows from the left join. The Null values for the Model Types that do not exist in the Sales table are highlighted in red.

based on a single table, but sometimes the company wants to retrieve data from both tables. The UNION command is the solution:

```
SELECT EID, LastName, FirstName, Gender, Phone, 'East'
As Division
FROM EmployeesEast
UNION
SELECT EID, LastName, FirstName, Gender, Phone, 'West'
As Division
FROM EmployeesWest
```

Figure 2.36 shows the result of the query. Notice that the columns must match exactly. Also, note the use of the created column Division to track the division for each employee after the data has been merged. In most cases, Union queries are saved as views and used as the basis for other searches. Union queries are useful for merging data from multiple sources—hence they appear in data warehouse situations when data comes from many places.

## Data Manipulation

All of the queries covered so far have been SELECT statements—queries to retrieve data. From a data mining perspective, these are the most useful queries because they are used to set up data for analyses. They are also used to find answers to ad hoc questions. However, it is worth knowing that SQL is capable of other types of commands. Specifically, SQL has **data manipulation** and **data definition** commands. The data definitions commands are used to create tables, indexes, and other structures. For example, they are used to load the databases for this book, but the details are not covered in this book. On the other hand, the data manipulation commands are often used when configuring and loading data for a data warehouse. Even if you do not need to create the commands as a manager, it is worthwhile to get a glimpse of the power of the commands so you know what tasks can be handled by SQL. The three main commands are: UPDATE to change

EID	LastName	FirstName	Gender	Phone	Division
113	Jones	Jack	Male	2222	East
114	Smith	Sarah	Female	4444	East
225	Hart	Hank	Male	6624	West
256	Eccles	Ephraim	Male	4432	West

**Figure 2.36**

UNION example. Combining rows from two employee tables (East and West). Note that the columns in the two SELECT statements must match exactly.

data values, INSERT to copy data, and DELETE to delete rows of data. All three rely heavily on the WHERE concepts covered in the SQL command.

## UPDATE

The SQL UPDATE command alters data in existing rows. Typically, it is applied to a single table at a time. The command operates on a single row at a time, but the WHERE clause can be used to apply the operation to multiple rows of data. Figure 2.37 shows an example of the UPDATE command that increases the list price of components by ten percent. The WHERE clause restricts the updates to components introduced in 2006 or later.

The UPDATE command can change multiple columns at the same time—all on the same row of data. The basic syntax is to separate the columns with commas: SET Col1=x, Col2=y, Col3=6.

To be safe, all updates should be tested first as SELECT statements. This approach is useful for ensuring the WHERE clause and computations are correct. For example, the example would be tested as:

```
SELECT 1.10*ListPrice AS NewPrice
FROM Component
WHERE Year >= 2006
```

## INSERT

The INSERT command has two forms—one to insert single rows of data at a time and the other to copy rows of data from one table and insert them into a second table. The first method is useful for transactions, but the second approach is more useful for data warehouses where insert commands are useful for transferring data. Figure 2.38 shows an example of using INSERT to copy data. It assumes that a new table exists to hold bicycle sales data.

Always test the SELECT statement first! Then add the INSERT line at the top to send the results into the new table. For loading data warehouses, the SELECT statement is often used to make minor changes to the data as it is being transferred. For example, using an exchange rate table, monetary data could be converted to a standard currency.



```
UPDATE Component
SET ListPrice = 1.10*ListPrice
WHERE Year>=2006
```

### Figure 2.37

UPDATE command. Increase the List Price of components that were introduced in 2006 or later.

## DELETE

The SQL DELETE command is powerful—probably too powerful to trust. Avoid using it—besides how often do you really want to delete data? The basic format is:

```
DELETE
FROM Manufacturer
WHERE ManufacturerID =1000;
```

Do not run the command—the ID was specifically chosen because it does not match any manufacturers in the Rolling Thunder database, so it will not actually delete any rows. Any DELETE command should first be tested by writing it as a SELECT \* statement. When you are completely satisfied that the WHERE clause is correct and that the data absolutely must be deleted, the SELECT statement can be replaced with the DELETE line.

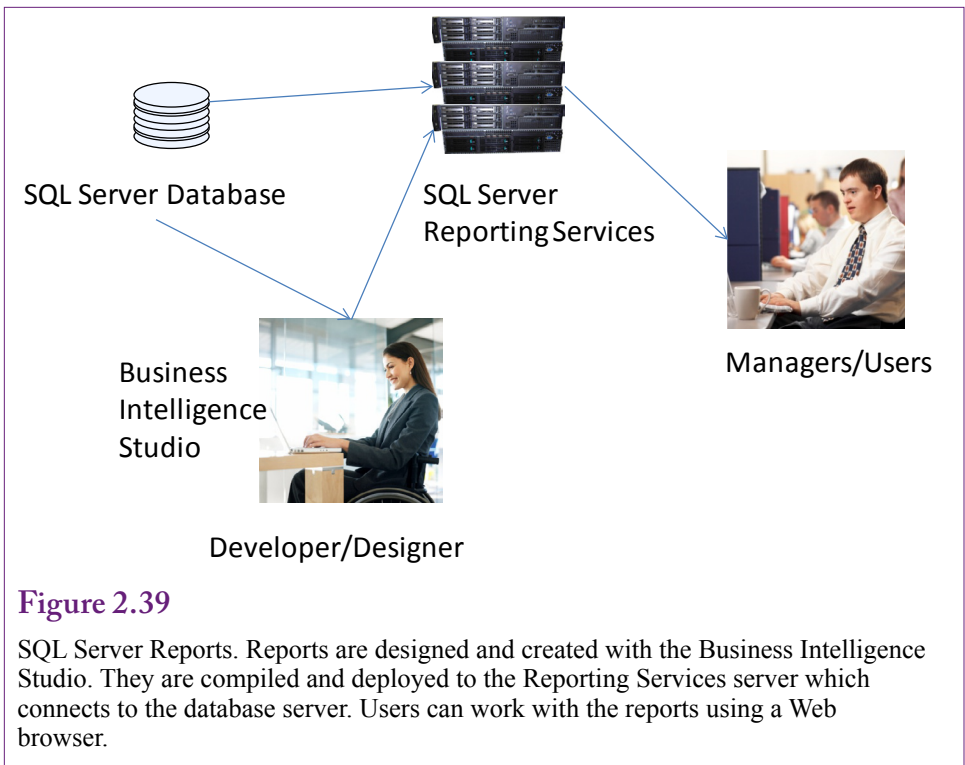
However, DELETE commands are dangerous. Tables are interrelated, and deleting a row from one table causes cascade deletes in the other tables. For instance, deleting a manufacturer deletes all purchase orders and components associated with that vendor. Deleting components removes them from the BikeParts table, so it could remove information on which parts were installed on a bicycle. Deleting customers or employees is even more dangerous because those objects affect many tables.

If data is accidentally deleted, it might be possible to recover by entering a ROLLBACK command, if it is entered early enough. SQL Server keeps a transaction log of changes and has the ability to reset the database to an earlier point in the log—but only if the changes have not been fully committed. In other cases, it might be necessary to find and restore a backup copy of the database. To be safe, avoid deleting data. Still, it is useful in data warehouses where some tables may be emptied before loading new data.

### Figure 2.38

INSERT command to transfer data. Test the SELECT query first and then add the INSERT command to send the results to a new table.

```
INSERT INTO newTable (SerialNumber, CustomerID, OrderDate, ModelType)
SELECT SerialNumber, CustomerID, OrderDate, ModelType
FROM Bicycle
WHERE OrderDate BETWEEN '01-JAN-2008' AND '31-DEC-2008';
```



**Figure 2.39**

SQL Server Reports. Reports are designed and created with the Business Intelligence Studio. They are compiled and deployed to the Reporting Services server which connects to the database server. Users can work with the reports using a Web browser.

## SQL Server Reports

**How are reports created in SQL Server?** Queries are useful but they do not have much in the way of formatting options. Reports combine queries with layout and format options to produce more useful information. In the “old days” reports were designed to be printed on paper. Now, SQL Server Reporting Services can run as a Web server and generate fixed reports as well as interactive reports that provide some initial exploration of the data to present both details and subtotals.

SQL Server reports require a computer running the Reporting Services and the **SQL Server Business Intelligence (BI)** (or Visual Studio) client tools. As shown in Figure 2.39, reports are created on a developer computer using the BI studio. The finished reports are compiled and deployed to a server running the SQL Server Reporting Services. This tool creates a Web interface, so managers can browse to the server and retrieve the reports using a standard Web browser. For development purposes, all of the components (SQL Server database, SQL Server Reporting Services, and the Business Intelligence Studio) can be installed on a single computer. However, if you are using a laptop, the Reporting Services are generally turned off by default to reduce the processing demands during typical use. To test reports, you will need to use the Computer Manager and Windows Services to start the Reporting Services running.

### Administration Configuration

Reporting Services has a couple of administrative twists. Particularly when Reporting Services is installed on a Windows 7 operating system, you will probably

- Start browser/Internet Explorer with: Run as Administrator
- Open report folder: `http://<server>/Reports`    `http://localhost/Reports`
- Use Tools/Options to add site as trusted site: `http://localhost`
- At reports site: Click the Properties tab
- Click button: New Role Assignment
- Enter your Windows user account: `<domain>\<user>`
- Click the option for: Content Manager
- Click the OK button
  
- Click the Site Settings link (top right)
- Select Security tab/option (left side)
- Click: New Role Assignment
- Enter your Windows user account: `<domain>\<user>`
- Check option: System Administrator
- Click OK

### Figure 2.40

SQL Server Reporting Services configuration. With Vista and Windows 7 it is usually necessary to add the developer account as an administrator to Reporting Services.

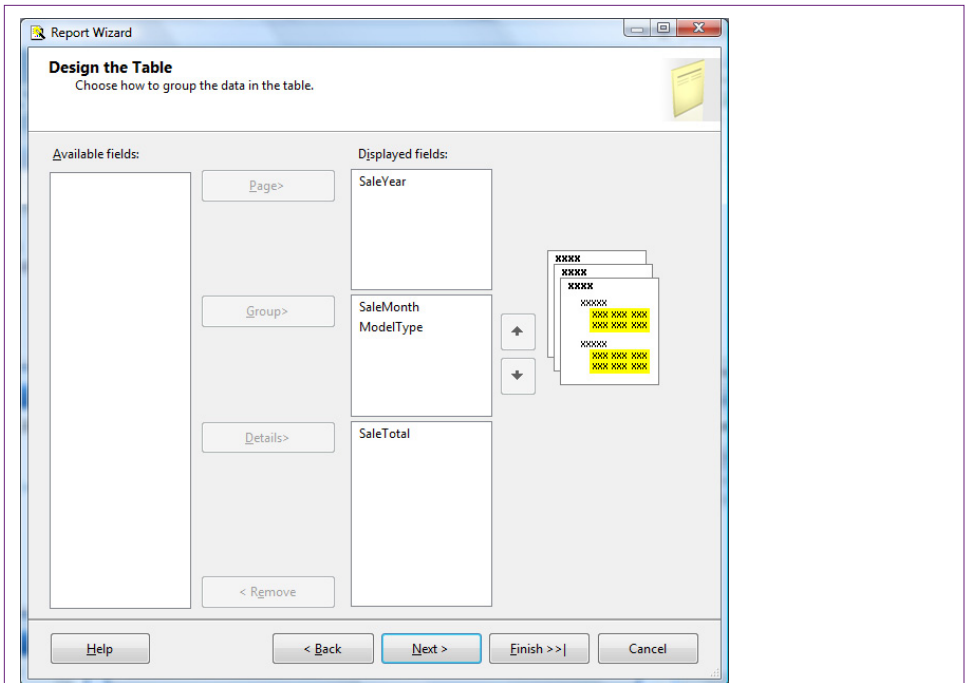
need to add the developer account (yours) as an administrator to the Reporting Services. The first twist is that this process is handled through a Web browser, not through the standard Database console. Figure 2.40 shows the basic steps. Reporting Services typically has given high-level permissions only to the Windows Administrator account, so the browser must be started with “Run as Administrator.” Use the localhost name if the Reporting Services are running on the development computer—otherwise, replace localhost with the name of the server running the services. Setting these security permissions is required to enable you to create and save reports on the server.

### Creating a Report

The Business Intelligence Studio has a report wizard that makes it relatively painless to create a basic report. The wizard is useful for establishing the database connections and defining the structure of the report. Once the report has been created, you can edit the design, improve the design and layout, and add more features if needed. Begin by starting the BI Studio and creating a new Report project. Choose a location to store the project files on your computer and enter a name that describes the project.

Visual Studio has several windows that display menus, code, properties, and report designs. All of these windows can be moved, resized, or closed. If a window is closed, search the menu options to open it when needed. To create a new report, in the Solution Explorer window, right-click the Reports entry and choose the option to add a new report. Follow the basic Wizard steps. As shown in Figure 2.41, the first step in creating a report is to establish a connection to the database. A connection requires the name of the server running the SQL Server DBMS, which could be a separate server. For most teaching purposes, the database is installed on the developer’s computer, so localhost usually works well as the server name. The login credentials must be entered for connecting to the database. Using the Windows account works for databases on the local computer. Choose the name of





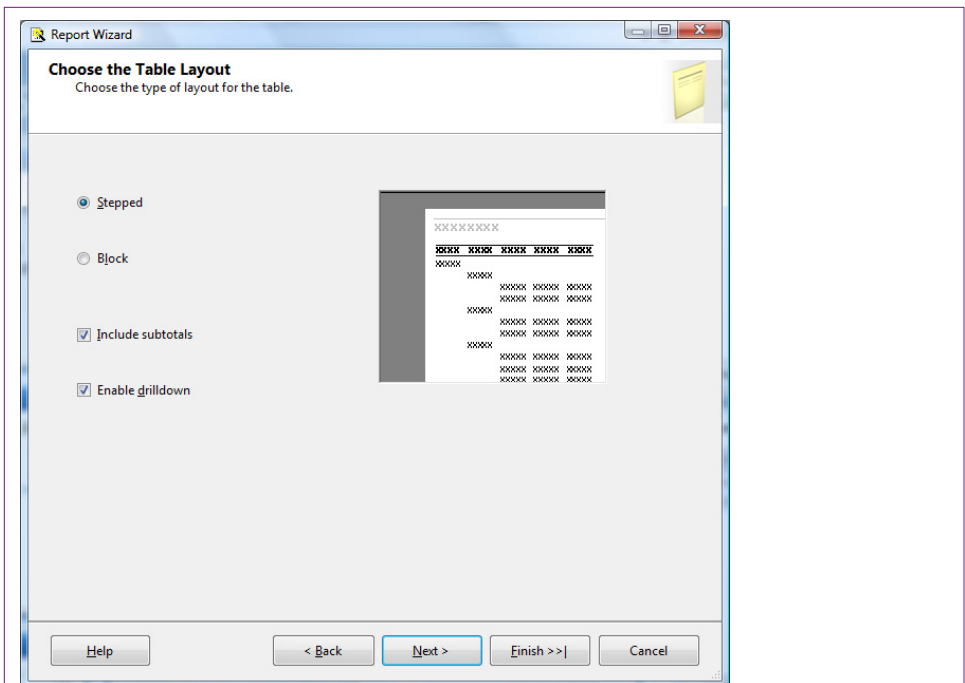
**Figure 2.43**

Report structure or levels. Before this step, choose the Tabular report layout instead of Matrix. Move columns from the Available fields panel into the appropriate level. SaleYear is the page break, SaleMonth and ModelType are the main Group breaks, and SaleTotal is the detail level.

the database and always click the button to test the connection. The data source can be saved as a shared data source (check box) so that it is available for multiple reports.

The most important aspect of a report is to build a query that retrieves the data needed for the report. The key to configuring the query is to retrieve the detail data needed for the report. Reports can have many levels with headings and subtotals; but ultimately, each report has a certain level of detail. The query needs to return the detail-level of data. The report itself will handle subtotals and formats. As an example, the goal is to create a report that displays total monthly sales of bicycles by model type. The total refers to value (of SalePrice) as opposed to the count of the number of bicycles. The goal is to create a report that lists one year on a page then the totals for each month with the detail level of sales by model type for that month. Consequently, the detail level needed in the query is the sum of SalePrice by month. As shown in Figure 2.42, the query also needs to return the Year value of the OrderDate to use as a page variable. The SQL statement is:

```
SELECT      YEAR(OrderDate) AS SaleYear,
           YEAR(OrderDate) * 100 + MONTH(OrderDate) AS
SaleMonth, ModelType,
           SUM(SalePrice) AS SaleTotal
FROM      Bicycle
GROUP BY  YEAR(OrderDate), YEAR(OrderDate) * 100 +
MONTH(OrderDate), ModelType
ORDER BY  SaleMonth, ModelType
```



**Figure 2.44**

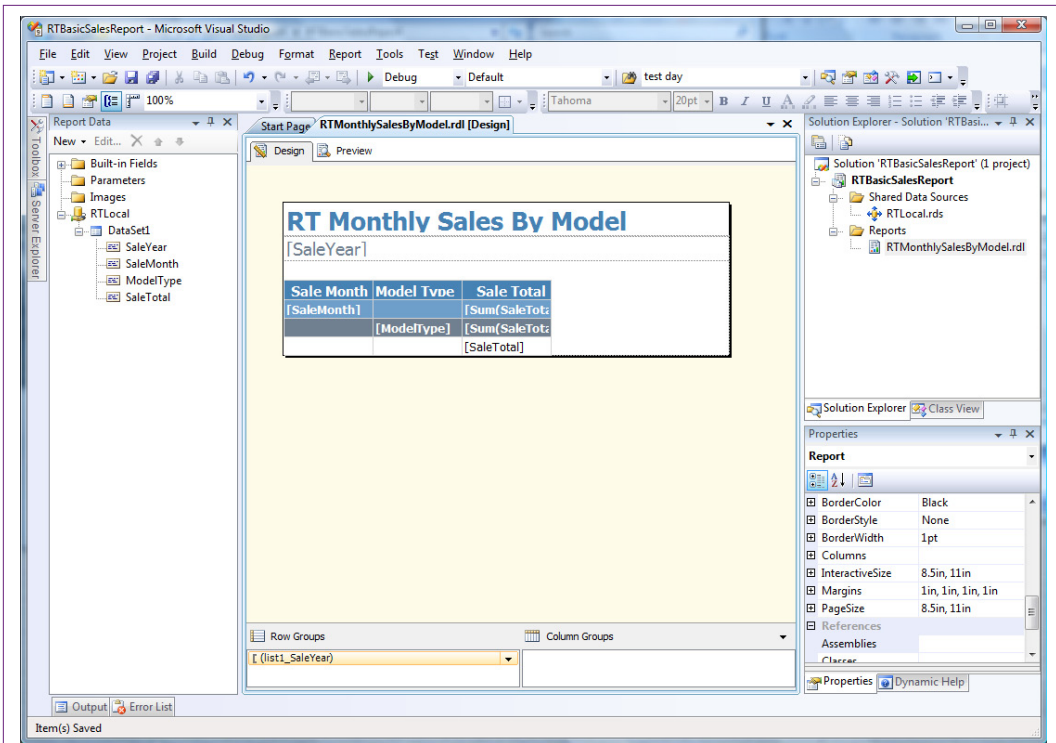
Report layout. Either Stepped or Block will work, but be sure to check both boxes to include subtotals and enable drilldown.

With the data selected, the next step is to define the report structure. Reports can have multiple levels or breaks. Each level can have a header and footer section. Headers are typically used to display titles or labels, and footers are used for subtotals. In the sales by month by model type example, the report will have groups for SaleMonth and ModelType. The page level break will be based on the Year.

Figure 2.43 shows the structure for the sample report using the Tabular layout. Place the SaleYear in the page level break. Place SaleMonth and Model Type—in that order—in the Group section. Move SaleTotal into the detail section.

For more complex cases, it is often best to place only the key or break items into the respective Group boxes. Avoid including additional data. For instance, if a grouping by Customer is desired, place CustomerID into the Group box, but do not include the customer name, address, phone number and so on. The report builder does not associate the additional columns with the key value—it will try to create new groupings based on each column in the group list. Leave related columns in the “available fields” box for now. They can be added to the report by hand after the main structure is defined.

Figure 2.44 shows the next step in the wizard—selecting the report layout. The stepped and block layouts are simply graphics design choices. Sometimes a report has to be tested both ways to see which version displays the data the best—particularly when a large number of columns are involved. The two checkboxes are more important—be sure to select both options to include subtotals and enable drilldown.



**Figure 2.45**

Initial Report design.

After this choice, the wizard provides options to set the overall color and style of the report. In a large project, all reports should follow a similar style. Some companies establish design guidelines that are used for all reports. It is also possible to add custom designs to the list, but those details are not covered in this book. Simply choose one of the existing designs. Follow the steps to finish the wizard and be sure to give the report a meaningful name that describes the report to users. However, the name should not include spaces. If necessary, use the underscore character ( `_` ) as a separator.

Figure 2.45 shows the initial report design and the Visual Studio editor. Because the model type names and sales totals might be wider than the allotted columns, it is useful to expand the column widths by dragging the dividing lines. Also, spaces can be added to the title. The report name itself should not contain spaces, but the display on the report can contain any desired characters. It is also possible to add logos and graphics.

The report can be previewed by clicking the Run button on the main toolbar (small green arrowhead). A useful improvement is to format the display of the totals. In the design mode, select the totals individually. Scroll the properties window near the bottom to find the Format line in the Number section. Enter `#,###0.00` as the format to specify that the values should display with two decimal places and a separator for thousands. The report can be previewed with the Run button. When it is acceptable, right-click the main project name and choose the option to Deploy the project and reports to the reporting server.



The screenshot shows a web browser window displaying a report from SQL Server Reporting Services. The report title is "RTMonthlySalesByModel" and it is for the year 2008. The report is presented in a table with three columns: "Sale Month", "Model Type", and "Sale Total". The data is grouped by month, with some months expanded to show sub-totals for different model types.

Sale Month	Model Type	Sale Total
200801		974,070.00
200802		733,160.00
200803		1,023,300.00
200804		899,460.00
	Mountain	82,250.00
	Mountain full	217,360.00
	Race	358,170.00
	Road	204,470.00
	Tour	37,210.00
200805		962,760.00
200806		842,100.00
200807		934,420.00
200808		812,170.00
200809		879,790.00
200810		909,170.00
200811		905,580.00
200812		814,250.00

**Figure 2.46**

Report displayed in browser. On a local machine, use `http://localhost/reports` to connect to the report server. Then navigate to the specific report.

Figure 2.46 shows the report generated from within a browser. The default location for the reporting services on the local computer is `http://localhost/reports`. After the browser connects to that site, navigate the links to the RT project and open the new report. Notice the scroll buttons at the top of the form to move to a new page, and remember that each page contains data for one year. Initially, the report displays the subtotals for each month. However, because the drilldown option was selected earlier, managers can click the + button in front of a given year and the month will be expanded to show the totals for each model type. This approach is useful for displaying multiple levels of data. Each item added as a Group entry creates a new level that can be expanded or contracted. Still, the approach is limited to the report layout specified in the design. More flexible options are available with the cube browser described in Chapter 3.

It is also possible to add selection boxes to the top of the page to set parameters that can be used in the query to control which data is displayed on the report. Additional calculations can be added to display percentages and other totals. All of these changes are made on the design of the report by adding labels, text boxes, and drop-down lists. Some of these tasks have little tricks (such as using `Parameters.pname.Label` to display a parameter's value) and the details are not covered in this book. Even managers with minimal background in development can learn to build relatively complex reports, but this book focuses more on the exploration of data.

The screenshot shows a 'Sales' form with the following fields and values:

- SaleID: 117
- Customer: Jones
- Phone: (312) 555-1234
- SaleDate: 3/3/2009
- Salesperson: Johnson

The 'ItemsSold' table contains the following data:

ItemID	Category	Description	Price	Quantity	Extended
1154	Shoes	Red Boots	\$100.00	2	\$200.00
3342	Electronics	LCD-40 inch	\$1,000.00	1	\$1,000.00
7653	Shoes	Blue Suede	\$50.00	4	\$200.00
*					

The subtotal is \$1,400.00.

**Figure 2.47**

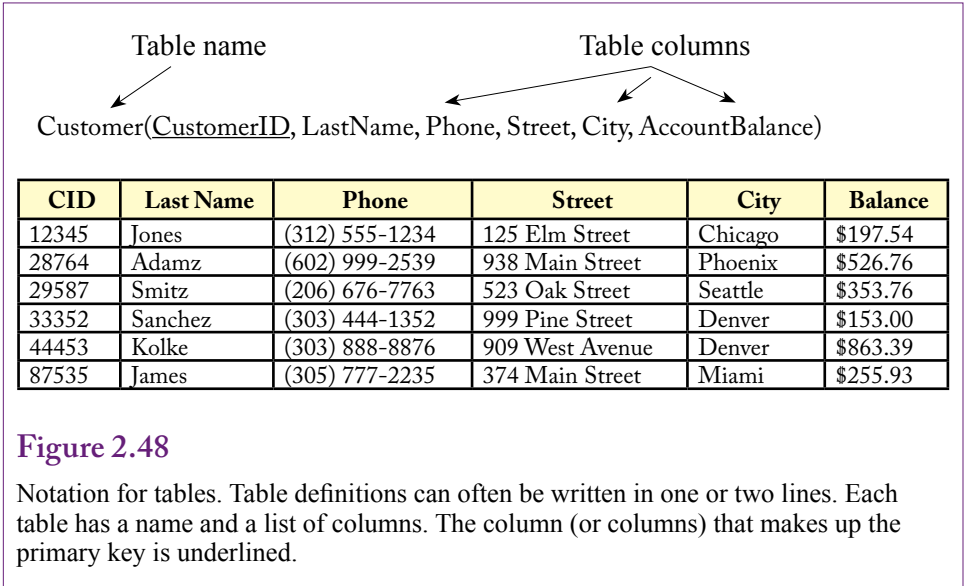
The order form is used in almost any firm. We need to determine the best way to store the data that is collected by this form.

## Database Design Concepts

**How do you create a new database?** This section is optional. It introduces the techniques used to design database tables. Most managers will not be required to create databases, and if they do, a database expert should be consulted to evaluate the design. In a data mining context, most tables already exist; however, it is sometimes useful to understand the foundations of designs to see why some columns are included in specific tables and others are not. Database management systems are powerful tools with the ability to present data in many ways. They are used by managers to answer many different types of questions. However, this flexibility is not automatic. Databases need to be carefully designed; otherwise, managers will not be able to get the information they need. Poor design also leads to unnecessary duplication of data. Duplication wastes space and requires workers to enter the same data several times. **Normalization** is an important technique to design databases.

To understand the process of normalization, consider the example of retail sales. You begin by thinking about who will be using the database and identifying what data they will need. Consider the situation of the salespeople. They first identify the customer then record each item being purchased. The computer should then calculate the amount of money due along with any taxes. Figure 2.47 shows a sample input screen that might be used.

The key design point is that you will need multiple tables to store the data. If you try to cram all of it into a single table, you will end up with unnecessary duplication of data and plenty of problems when you try to delete or insert new data. Each entity or object on the form will be represented by a separate table. For this



example, there are five objects on the form: Customers, Salespeople, Items, Sale, and ItemsSold..

Before explaining how to derive the five tables from the form, you need to understand some basic concepts. First, remember that every table must have a primary key. A primary key is one or more columns that uniquely identify each row. For example, you anticipate problems with identifying customers, so each customer will be assigned a unique ID number. Similarly, each item is given a unique ID number. There is one drawback to assigning numbers to customers: you cannot expect customers to remember their number, so you will need a method to look it up. One possibility is to give everyone an ID card imprinted with the number—perhaps printed with a bar code that can be scanned. However, you still need a method to deal with customers who forget their cards. It is usually better to build a method to look up customers by name.

The second aspect to understand when designing databases is the relationships between various entities. First, observe that there are two sections to the form: (1) the main sale that identifies the transaction, the customer, the salesperson, and the date, and (2) a repeating section that lists the items being purchased. Each customer can buy several different items at one time. There is a **one-to-many** relationship between the Sale and the ItemsSold sections. As you will see, identifying one-to-many relationships is crucial to proper database design.

In some respects, designing databases is straightforward: There are only three basic rules. However, database design is often interrelated with systems analysis. In most cases, you are attempting to understand the business at the same time the database is being designed. One common problem that arises is that it is not always easy to see which relationships are one-to-many and which are one-to-one or many-to-many.

SaleForm(SaleID, SaleDate, CustomerID, Phone, Name, Street, (ItemID, Quantity, Description, Price ) )

Repeating Section  
Causes duplication

SaleID	SaleDate	CID	Name	Phone	Street	ItemID	Qty	Description	Price
117	3/3/2009	12345	Jones	(312) 555-1234	125 Elm Street	1154	2	Red Boots	\$100.00
117	3/3/2009	12345	Jones	(312) 555-1234	125 Elm Street	3342	1	LCD-40 inch	\$1,000.00
117	3/3/2009	12345	Jones	(312) 555-1234	125 Elm Street	7653	4	Blue Suede	\$50.00
125	4/4/2009	87535	James	(305) 777-2235	374 Main Street	1154	4	Red Boots	\$100.00
125	4/4/2009	87535	James	(305) 777-2235	374 Main Street	8763	3	Men's Work Boots	\$45.00
157	4/9/2009	12345	Jones	(312) 555-1234	125 Elm Street	7653	2	Blue Suede	\$50.00
169	5/6/2009	29587	Smitz	(206) 676-7763	523 Oak Street	3342	1	LCD-40 inch	\$1,000.00
169	5/6/2009	29587	Smitz	(206) 676-7763	523 Oak Street	9987	2	Blu-Ray Player	\$400.00
178	5/1/2009	44453	Kolke	(303) 888-8876	909 West Avenue	2254	1	Blue Jeans	\$12.00
188	5/8/2009	29587	Smitz	(206) 676-7763	523 Oak Street	3342	1	LCD-40 inch	\$1,000.00
188	5/8/2009	29587	Smitz	(206) 676-7763	523 Oak Street	8763	4	Men's Work Boots	\$45.00
201	5/23/2009	12345	Jones	(312) 555-1234	125 Elm Street	1154	1	Red Boots	\$100.00

**Figure 2.49**

Converting to notation. The basic rental form can be written in notational form. Notice that repeating sections are indicated by the inner parentheses. If you actually try to store the data this way, notice the problem created by the repeating section: Each time a customer checks out a video we have to reenter the phone and address.

## Notation

It would be cumbersome to draw pictures of every table that you use, so you usually write table definitions in a standard notation. The base customer table is shown in Figure 2.48, both in notational form and with sample data.

Figure 2.48 illustrates another feature of the notation. You denote one-to-many or repeating relationships by placing parentheses around them. Figure 2.49 represents all the data shown in the input screen from Figure 2.47. The description is created by starting at the top of the form and writing down each element that you encounter. If a section contains repeating data, place parentheses around it. Preliminary keys are identified at this step by underlining them. However, you might have to add or change them at later steps. You can already see some problems with trying to store data in this format. Notice that the same customer name, phone, and address would have to be entered several times.

Remember that some repeating sections are difficult to spot and might consist of only one column. For example, how many phone numbers can a customer have? Should the Phone column be repeating? In the case of the retail store, probably not, because you most likely want to keep only one number per customer. In other businesses, you might want to keep several phone numbers for each client. Data normalization is directly related to the business processes. The tables you design depend on the way the business is organized.

SaleID	SaleDate	CID	Name	Phone	Street	ItemID, Quantity, Description, Price
117	3/3/2009	12345	Jones	312-555-1234	125 Elm Street	1154, 2, Red Boots, \$100.00 3342, 1, LCD-40 inch, \$1,000.00 7653, 4, Blue Suede, \$50.00
125	4/4/2009	87535	James	305-777-2235	374 Main Street	1154, 4, Red Boots, \$100.00 8763, 3, Men's Work Boots, \$45.00
157	4/9/2009	12345	Jones	312-555-1235	125 Elm Street	7653, 2, Blue Suede, \$50.00
169	5/6/2009	29587	Smitz	206-676-7763	523 Oak Street	3342, 1, LCD-40 inch, \$1,000.00 9987, 2, Blu-Ray Player, \$400.00
178	5/1/2009	44453	Kolke	303-888-8876	909 West Ave.	2254, 1, Blue Jeans, \$12.00
188	5/8/2009	29587	Smitz	206-676-7763	523 Oak Street	3342, 1, LCD-40 inch, \$1,000.00 8763, 1, Men's Work Boots, \$45.00
201	5/23/2009	12345	Jones	312-555-1234	125 Elm Street	1154, 1, Red Boots, \$100.00

Figure 2.50

A table that contains repeating sections is not in first normal form. Each table cell can contain only basic data. Storing it in repeating form makes it difficult to search, insert, and delete data. This version is not in first normal form.

Figure 2.51

Splitting a table to solve problems. Problems with repeating sections are resolved by moving the repeating section into a new table. Be sure to include the old key in the new table so that you can connect the tables back together.

SaleForm(SaleID, SaleDate, CID, Phone, Name, Street, (ItemID, Quantity, Description, Price) )

SaleForm2(SaleID, SaleDate, CustomerID, Phone, Name, Street)

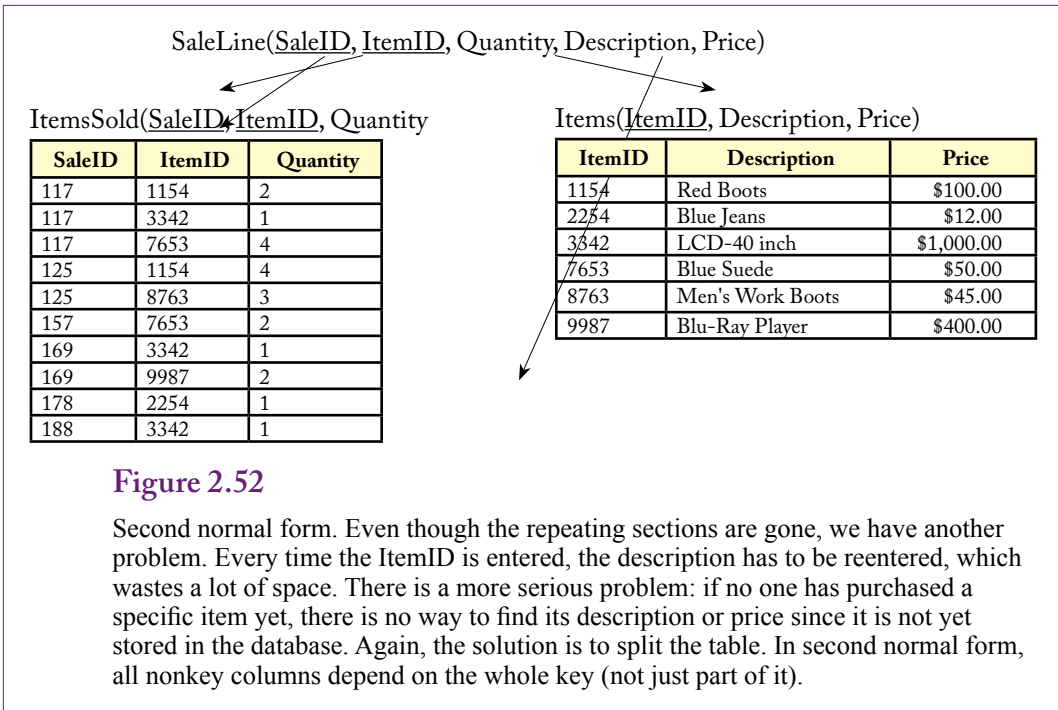
Note replication

SaleID	SaleDate	CID	Name	Phone	Street
117	3/3/2009	12345	Jones	(312) 555-1234	125 Elm Street
117	3/3/2009	12345	Jones	(312) 555-1234	125 Elm Street
117	3/3/2009	12345	Jones	(312) 555-1234	125 Elm Street
125	4/4/2009	87535	James	(305) 777-2235	374 Main Street
125	4/4/2009	87535	James	(305) 777-2235	374 Main Street

SaleLine(SaleID, ItemID, Quantity, Description, Price)

Note replication

SaleID	ItemID	Quantity	Description	Price
117	1154	2	Red Boots	\$100.00
117	3342	1	LCD-40 inch	\$1,000.00
117	7653	4	Blue Suede	\$50.00
125	1154	4	Red Boots	\$100.00
125	8763	3	Men's Work Boots	\$45.00
157	7653	2	Blue Suede	\$50.00
169	3342	1	LCD-40 inch	\$1,000.00
169	9987	2	Blu-Ray Player	\$400.00
178	2254	1	Blue Jeans	\$12.00
188	3342	1	LCD-40 inch	\$1,000.00
188	8763	4	Men's Work Boots	\$45.00
201	1154	1	Red Boots	\$100.00



## First Normal Form

Now that you have a way of writing down the assumptions, it is relatively straightforward to separate the data into tables. The first step is to split out all repeating sections. Think about the problems that might arise if you try to store the repeating data within individual cells. You will have to decide how many rows to set aside for storage, and you will have to write a separate search routine to evaluate data within each cell, and complications will arise when inserting and deleting data. Figure 2.50 illustrates the problem.

The answer to this problem is to pull out the repeating section and form a new table. Then, each item purchased by a customer will fill a new row. Figure 2.51 uses the notation to show how the table will split. Notice that whenever you split a table this way, you have to bring along the key from the prior section. Hence, the new table will include the SaleID key as well as the ItemID key. When a table contains no repeating sections, you say that it is in first normal form.

## Second Normal Form

Even if a table is in first normal form, there can be additional problems. Consider the SaleLine table in Figure 2.51. Notice there are two components to the key: SaleID and ItemID. The nonkey items consist of the Quantity, Description, and Price of the item. If you leave the table in this form, consider the situation of selling a new item. Every time an item is sold it will be necessary to reenter the Description and list Price. It means that you will be storing the description every time an item is sold. Popular items might be sold thousands of times. Do you really want to store the description (and other data) each time?

The reason you have this problem is that when the SaleID changes, the item description stays the same. The description depends only on the ItemID. If the Price

SaleForm2(SaleID, SaleDate, CustomerID, Phone, Name, Address)

Sales(SaleID, SaleDate, CustomerID, SalespersonID)

SaleID	SaleDate	CID	SPID
117	3/3/2009	12345	887
125	4/4/2009	87535	663
157	4/9/2009	12345	554
169	5/6/2009	29587	255
178	5/1/2009	44453	663
188	5/8/2009	29587	554

Customers(CustomerID, Phone, Name, Address, City, State, ZIPCode, AccountBalance)

CID	Name	Phone	Street	City	Balance
12345	Jones	(312) 555-1234	125 Elm Street	Chicago	\$197.54
28764	Adamz	(602) 999-2539	938 Main Street	Phoenix	\$526.76
29587	Smitz	(206) 676-7763	523 Oak Street	Seattle	\$353.76
33352	Sanchez	(303) 444-1352	999 Pine Street	Denver	\$153.00
44453	Kolke	(303) 888-8876	909 West Avenue	Denver	\$863.39
87535	James	(305) 777-2235	374 Main Street	Miami	\$255.93

**Figure 2.53**

Third normal form. There is another problem with this definition. The customer name does not depend on the key (SalesID) at all. Instead, it depends on the CustomerID. Because the name and address do not change for each different SalesID, the customer data must be in a separate table. The Sales table now contains only the CustomerID, which is used to link to the Customers table and collect the rest of the data. The same rule applies to Salespeople.

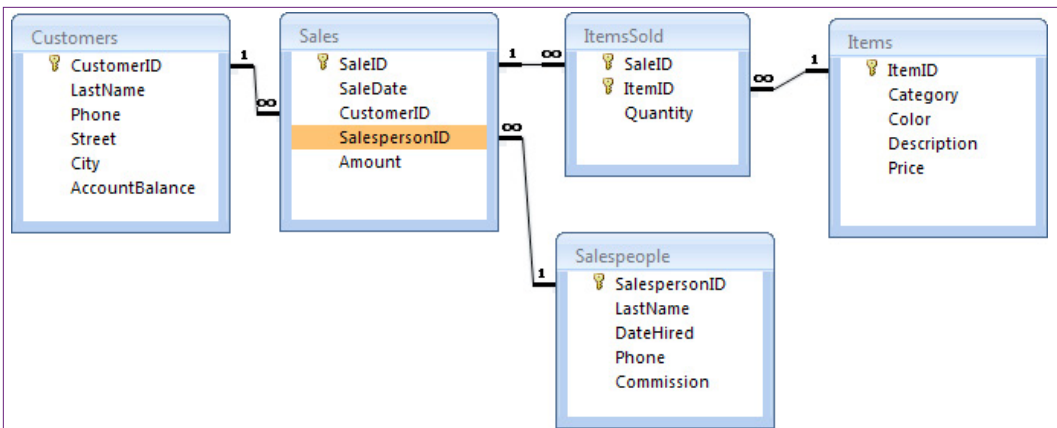
represents the list price of the item, the same dependency holds. However, what if the store offers discounts on certain days or to specific customers? If the price can vary with each transaction, the price would have to be stored with the SaleID. The final choice depends on the business rules and assumptions. Most companies resolve the problem by creating a list price that is stored with the item and a sale price that is stored with the transaction. However, to simplify the problem, stick with just the list price for now.

When the nonkey items depend on only part of the key, you need to split them into their own table. Figure 2.52 shows the new tables. When each nonkey column in a table depends on the entire key, the table is in second normal form.

### Third Normal Form

Examine the SaleForm2 table in Figure 2.51. Notice that because the primary key consists of only one column (SaleID), the table must already be in second normal form. However, a different problem arises here. Again, consider what happens when you begin to collect data. Each time a customer comes to the store and buys something there will be a new transaction. In each case, you would have to record the customer name, address, phone, city, and so on. Each entry in the transaction table for a customer would duplicate this data. In addition to the wasted space, imagine the problems that arise when a customer changes a phone number. You might have to update it in hundreds of rows.





**Figure 2.54**

Third normal form tables. There are no repeating sections and each nonkey column depends on the whole key and nothing but the key. This figure also shows the relationships between the tables that will be enforced by the DBMS. When referential integrity is properly defined, the DBMS will ensure that rentals can be made only to customers who are defined in the Customers table.

The problem in this case is that the customer data does not depend on the primary key (SalesID) at all. Instead, it depends only on the CustomerID column. Again, the solution is to place this data into its own table. Figure 2.53 shows the split. Splitting the table solves the problem. Customer data is now stored only one time for each customer. It is referenced back to the Rentals table through the CustomerID. The same rule applies to Salespeople, resulting in the fifth table.

The five tables you created are listed in Figure 2.54. Each table is now in third normal form. It is easy to remember the conditions required for third normal form. First: There are no repeating groups in the tables. Second and third: Each nonkey column depends on the whole key and nothing but the key.

Note in that if the Customers table contains complete address data, including ZIP Code, you could technically split the Customers table one more time. Because ZIP codes are uniquely assigned by the post office, the city and state could be determined directly from the ZIP code (they do not depend on the CustomerID). In fact, most mail order companies today keep a separate ZipCode table for that very reason. For our small retail firm, it might be more of a nuisance to split the table. Although you can purchase a complete ZIP code directory in computer form, it is a large database table and must be updated annually. For small cases, it is often easier to leave the three items in the Customer table.

## Summary

Databases, particularly relational database systems, hold much of the data used in business. Most business applications, including enterprise systems and Web sites, store data in relational databases. SQL Server Analysis Services is closely tied to the SQL Server database system. For these reasons, you need to know how to cre-

ate queries to retrieve data. Queries can be built with the design editor but they are actually built in SQL. The four main steps in building a query are: (1) Identify the output needed, (2) Specify the constraints, (3) Select the tables holding the data, and (4) Indicate how the tables are joined.

SQL can perform some basic computations. Arithmetic is handled on a row-by-row basis and new columns can be defined using data on a single row at a time. SQL Server support several functions to perform additional computations, such as standard mathematical functions, string manipulation, and date calculations and formatting. Aggregations (Sum, Count, Avg, and so on) are performed across rows of data. Subtotals are computed using the GROUP BY statement. Common business questions involve subtotals for each customer, employee, state or region, and item category. Questions involving complex nested subtotals are better handled with hyper cube browsers covered in the next chapter.

SQL has many powerful features useful for formatting and cleaning data sets. The UPDATE, INSERT, and DELETE commands are useful for transferring data. This chapter touched on the basic capabilities of SQL. People who need to focus on building data warehouse systems should study the advanced SQL options in more detail—in a database textbook.

## Key Words

---

aggregation	identity
BETWEEN	JOIN
Boolean algebra	normalization
columns	NOT
cross join	one-to-many
data definition	ORDER BY
data manipulation	primary key
data type	query system
database	row-by-row calculations
database management system (DBMS)	SELECT
DESC	SQL
enterprise resource planning (ERP)	SQL Server Business Intelligence (BI)
FROM	table
GROUP BY	view
HAVING	WHERE

## Review Questions

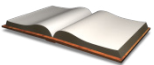
---

1. What is a table and why is it important in a relational database?
2. What four questions must be answered to create a query and what are the corresponding parts of the SQL statement?
3. What is the command word for matching portions of a string in SQL?
4. Why is it critical to store dates using a datetime format?
5. How are aggregation functions such as Sum different from arithmetic calculations (+/-)?
6. What is the purpose of the GROUP BY function?
7. In common problems involving subtotals created in the design editor, such as sales by customer, why is it often necessary to specify a WHERE option when a date condition is included?
8. What is the syntax for joining two tables in SQL (for example Bicycle and Customer)?
9. What is the typical role of database reports?

## Exercises

---

For the SQL exercises, just submit the SQL statements, not the query results.



### Book

1. List the tables in the Rolling Thunder Bicycles database.
2. SQL. Which race bicycles were ordered in 2012 with a framesize greater than 60 cm?
3. SQL. Which mountain bicycles in 2011 had a list price over 7000?
4. SQL. What is the total value of all bikes sold in November 2012?
5. SQL. What is the total number of bicycles sold of each model type in March 2012?
6. SQL. Who was the best sales person by value in 2011?
7. SQL. Which customer from California spent the most on bikes from 2010 through 2012?
8. SQL. Which model types were not sold in December 2012? (Hard: See database book)



## Rolling Thunder Database

9. SQL. Who are the top customers who bought the most bikes?
10. SQL. In which state were the most bicycles sold in 2010?
11. SQL. List the manufacturers and the total value of cranks purchased from them in 2012.
12. SQL. What was the most popular paint color for mountain bicycles (including full suspension) in 2011?
13. SQL. Create a query to list all Customers, Bicycles, and Components installed on bicycles in 2010
14. SQL. Create a query to list bicycle sales by month and model type by count and value, where month is displayed in the form YYYYMM such as 199401.



## Diner

15. What are the columns in the Diners table?
16. SQL. How does the average bill compare for Lunch, Dinner, and Evening?
17. SQL. What are the average sales by day of week?
18. SQL. What are the average sales by gender of the group?
19. SQL. How many times did a Mixed group order dessert? How many times did they not order dessert? Hint: Both can be answered with one query.



## Corner Med

20. List the tables in the Corner Med database.
21. SQL. Create a view to compute the number of patients and total amount billed by day.
22. SQL. What is the average number of patients seen per day and the standard deviation? Hint: Use the view from #21.

23. SQL. What is the average amount of money billed per day and the standard deviation? Hint: Use the view from #21.
24. SQL. Which physician has treated the most patients (based on visits)?
25. SQL. What is the most common diagnosis based on a first three letters of the diagnosis code? Hint: Use the Substring function.



## Basketball

26. List the tables in the basketball database.
27. SQL. Which player scored the most points in the regular season 2010-2011 (not playoff)?
28. SQL. Which player had the most total steals plus blocks plus defensive rebounds in the regular season 2010-2011?
29. SQL. Which player with at least 20 free throw attempts had the worst free throw percentage in 2009-2010? Hint: Cast(FT as real).
30. SQL. List the total number of wins by each team in the regular season 2010-2011 (not playoff), ordered by the number of wins.



## Bakery

31. List the tables in the Bakery database.
32. SQL. What is the total value of sales by year?
33. SQL. What is the total value of sales by category in 2009?
34. SQL. In 2010, what was the best-selling item in the Muffin category (by count)?
35. SQL. What is the average number of items and average value of a Sale (basket) in 2012? Hint: First create a view that computes values by sale.



## Cars

36. What are the columns in the Cars database?
37. SQL. What is the average MPG and average price by category?
38. SQL. What is the average weight and average price by Make (company)?
39. SQL. List the vehicles in descending order of Power/Weight ratio. Hint: Use Cast(HP as real).



## Teamwork

40. Each person in the group should choose one of the databases and write a business question (not in the existing exercises). Share each question with the other team members and write the query to answer all of the questions. Compare the answers by each team member.

## Additional Reading

---

Post, Gerald, 2011, *Database Management Systems*: <http://www.JerryPost.com/Books/DBBook>. [Textbook on standard database management systems, covering design, queries, applications, and management.]

## OLAP Cubes

### Chapter Outline

- Introduction, 96
- Challenges with the Relational Model, 98
  - Indexes, 99*
  - Data Warehouse, 99*
  - Extraction, Transformation, and Loading, 100*
  - MOLAP, ROLAP, and HOLAP, 101*
- OLAP Design, 102
  - Facts and Dimensions, 104*
  - Star Design, 105*
  - Snowflake Design, 107*
  - Hierarchies, 107*
- Creating a Cube with Microsoft Analysis Services, 108
  - Data Sources, 109*
  - Data Source Views, 111*
  - Cubes, 116*
- Dimensions, 120
  - Hierarchies, 123*
  - Time Dimensions, 124*
  - Custom Geographic Hierarchy, 128*
  - Attribute Relationships, 130*
- Fine Tuning the Cube, 133
  - Calculations and Queries, 134*
  - Perspectives, 138*
  - Internationalization and Translations, 140*
  - Performance: Partitions and Aggregations, 142*
- Excel PivotTables, 144
  - Actions, 146
- Key Performance Indicators, 149
  - Definition, 149*
  - Creating KPIs, 150*
  - Browsing a KPI, 153*
- Summary, 154
- Key Words, 155
- Review Questions, 155
- Exercises, 156
- Additional Reading, 158

### What You Will Learn in This Chapter

- Why is OLAP important or even necessary?
- Why are transactions and analysis difficult to combine?
- What is the process for designing and creating a data cube?
- How are OLAP cubes created using Microsoft SQL Server Analysis Services?
- How are dimensions created and modified to improve browsing?
- How can the cube provide more information?
- How can the cube be accessed outside of Analysis Services?
- How can the cube connect to external data such as Web sites and maps?
- How can simple data be provided to managers on a daily basis?



### **Winmetrics Corp.**

In some ways, OLAP cubes are similar to reports—with the added features of interaction. Users can click to drill down and get more detail, or roll up totals to compare by region, product, or any defined category. With a few clicks, a user can quickly filter by any desired conditions. Setting up an OLAP cube often takes time and experience, so companies such as WinMetrics make money by helping companies configure their databases and OLAP cubes. WinMetrics describes the results at one company, which is the leading outsource collection agency for government debts in America. Initially, the company used fixed printed reports, including a 500-page “CARE” report that took more than 24 hours to generate and was difficult to use. It took WinMetrics about eight weeks to create a system to clean and transfer the data to SQL Server and build the OLAP cube. After training some in-house engineers to build additional cubes, the company eventually built cubes for most of the accounting functions as well. Instead of relying on fixed reports for standard items including general ledger, payroll, budget, and forecasting, OLAP cubes were created to enable dynamic access to the data. The underlying data was exported from the third-party accounting system and transferred into OLAP cubes in SQL Server. All of the accounting statements are now accessible through a few clicks of the OLAP cube, including the ability to drill down and see details such as the source of variances in the budget model. [WinMetrics]

Managers and analysts want dynamic access to data. The ability to explore and examine different aspects is useful to understanding the results, spotting problems, and making decisions.

WinMetrics Corp., How a Financial Services Company Developed a Performance Report for Clients, Saved \$200K and Sold \$167,000,000 of Equity in Just 9 Months. [http://www.winmetrics.com/olap\\_casestudies.html](http://www.winmetrics.com/olap_casestudies.html)

## Introduction

---

**Why is OLAP important or even necessary?** Relational database systems were designed to collect and store transaction data efficiently. They have been effective at handling standard business data for dozens of years. Designing the database properly by splitting the data into tables is important. With separate tables, new data can be added without impacting the existing data—enabling the databases to become huge and still maintain performance when adding new data. SQL is a powerful tool that retrieves data from multiple tables, creates calculations and subtotals. So what is the reason for OLAP?

The problem with relational databases lies with retrieving the data. SQL and the visual designers are nice tools to use and they work well on small problems. But putting millions or billions (Jacobs 2009) of rows into multiple tables is going to cause problems. One of the biggest problems is created by joining tables. Joining tables by matching data keys is a slow process. Most relational database systems try to improve performance by building indexes—but the indexes take space and slow down updates, particularly with millions of rows of data.

**Online analytical processing (OLAP)** is designed specifically to improve performance for retrieving and analyzing data. The goal is to provide enough speed so that even with huge databases, analysts and managers can examine data interactively with minimal delays. To accomplish this task, OLAP systems typically store data in a new design format—which often entails pre-building all of the joins. Design issues are an important section of this chapter.

Even if the data is stored so that it can be retrieved efficiently, how are managers and analysts going to interact with the data? The presumption is that managers need to see various subtotals and to compare these values by different categories. For example, most managers need to see sales or production values at different points in time, and often want to compare values by different product or model types. As shown in Figure 3.1, a **cube browser** is a common OLAP tool that enables managers to select categories (such as Model Type, Time, and Location), and examine some critical fact for any desired subtotal. Generally, the cube is interactive and the manager can drag-and-drop items of interest and examine multiple levels of subtotals. Filters, such as Location in the example, can be used to display data for specific situations. All of these actions are accomplished by selecting or deselecting items on the screen—without writing any queries.

OLAP cubes are useful for exploring the data, but sometimes managers need even simpler tools. **Key performance indicators (KPI)** can be defined and used to create simple visual gauges that display the current values and trends of critical factors. For example, a sales manager or CEO might create a gauge that shows daily sales and comparisons to the prior year or month. Financial managers might track movements in stock prices or interest rates. Just as the gauges in an automobile provide information and feedback to drivers, **digital dashboards** are designed to provide critical information at a glance to managers.

Many tools exist to help build OLAP projects and analyze data. Some similarities exist across the tools, but all of them have their own quirks. This chapter focuses on the Microsoft business intelligence or SQL Server Analysis System (SSAS) tools.

	A	B	C	D	E	F	G	H	I	J
1	Sale Price	Column Labels	Mountain	Mountain full	Race	Road	Tour	Track	Grand Total	
2	Row Labels	Hybrid								
3	Calendar 1994	199,006	559,020		928,984	486,774	346,201	35,029	2,555,013	
4	Calendar 1995	124,240	567,940		294,390	1,074,490	709,230	187,920	2,958,210	
5	Calendar 1996	631,760	1,469,870		1,174,850	361,020	213,360		4,050,860	
6	Calendar 1997	2,780	783,820	1,826,320	1,851,160	894,040			5,358,120	
7	Calendar 1998	110,360	1,366,180	2,712,310	683,090	1,562,180	203,270		6,637,390	
8	Calendar 1999		3,138,210	2,792,330	1,998,490	1,816,790	634,260		10,380,080	
9	Calendar 2000	94,010	1,657,070	1,611,920	671,200	600,180			4,634,380	
10	Calendar 2001	180,500	1,221,870	1,399,290	936,430	876,740	72,890		4,687,720	
11	Calendar 2002		1,523,740	2,507,560	1,581,050	1,645,770	225,980		7,484,100	
12	Calendar 2003		2,241,600	3,675,790	1,858,560	2,593,660	526,680		10,623,290	
13	Calendar 2004		896,390	3,372,690	1,858,720	2,183,070	624,800		8,935,670	
14	Calendar 2005		934,540	2,812,930	3,113,150	2,285,410	486,990		9,633,020	
15	Calendar 2006		1,205,720	2,700,080	3,235,570	2,319,750	599,460		10,060,580	
16	Calendar 2007		1,098,330	3,370,890	4,431,690	2,965,450	313,340		12,179,700	
17	Calendar 2008		846,990	2,930,500	3,931,900	2,508,860	471,980		10,690,230	
18	Calendar 2009		829,900	3,452,630	4,244,930	2,951,360	296,470		11,775,290	
19	Calendar 2010		1,018,340	4,277,740	5,171,940	3,417,090	616,580		14,501,690	
20	Calendar 2011		894,730	5,435,980	5,708,890	4,079,390	1,009,970		17,128,960	
21	Calendar 2012	628,450	1,283,150	5,327,330	5,903,620	3,748,000	632,210		17,522,760	
22	Quarter 1, 2012	111,940	164,510	1,063,010	1,154,660	716,840	161,460		3,372,420	
23	Quarter 2, 2012	178,070	367,800	1,265,120	1,430,560	1,022,140	147,300		4,410,990	
24	April 2012	67,780	158,450	478,280	576,620	348,270	64,520		1,693,920	
25	May 2012	42,450	75,840	467,680	459,010	355,720	41,310		1,442,010	
26	June 2012	67,840	133,510	319,160	394,930	318,150	41,470		1,275,060	
27	Quarter 3, 2012	116,990	350,800	1,215,100	1,165,540	702,210	102,010		3,652,650	
28	Quarter 4, 2012	221,450	400,040	1,784,100	2,152,860	1,306,810	221,440		6,086,700	
29	Calendar 2013	736,360	1,286,090	5,437,450	6,093,490	3,687,020	695,840		17,936,250	
30	Calendar 2014	691,900	970,200	5,792,490	6,101,470	4,560,560	588,610		18,705,230	
31	Grand Total	3,399,366	25,793,700	61,436,230	61,700,574	46,617,604	9,268,121	222,949	208,438,543	
32										

Figure 3.1

OLAP cube browser. OLAP browsing consists of examining subtotals of a fact measure for any dimension. Typically, users can drag-and-drop to change dimensions. Hierarchies such as date provide the option to drill-down to see detail or rollup to see the summary level.

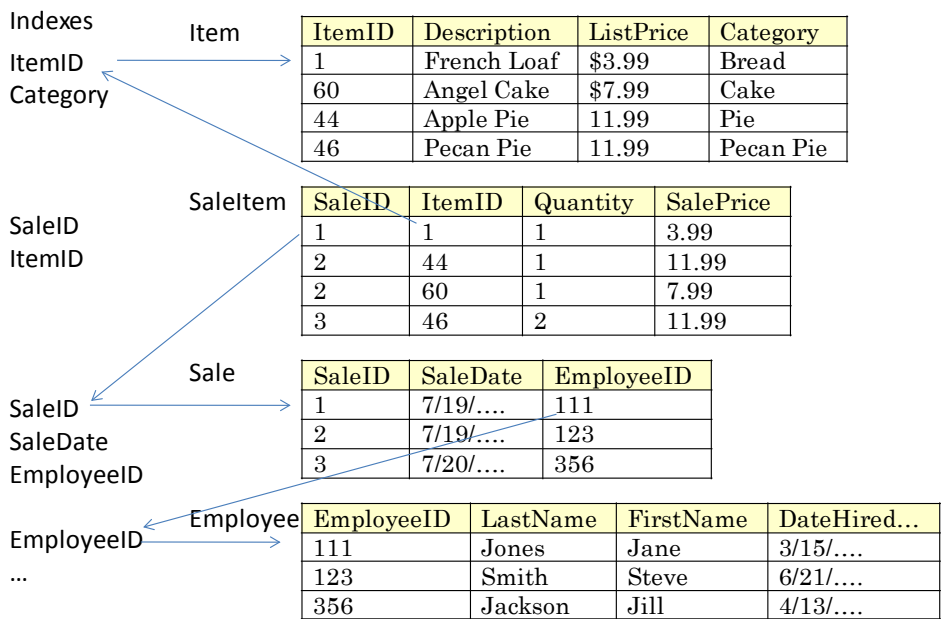
## Challenges with the Relational Model

**Why are transactions and analysis difficult to combine?** Relational databases are designed for **online transaction processing (OLTP)**. The primary task in OLTP is to collect transaction data (such as sales) and store it efficiently. The systems also print basic reports, but these usually consist of simple receipts and totals. Each data object is stored in a separate table—isolating it from the rest of the data. Adding new data is fast and efficient. Data duplication is avoided (except for backups), and data is typically retrieved in small pieces.

However, data retrieval has always been an issue with relational systems (and with hierarchical and network systems before that). Splitting data into multiple tables means that the tables have to be joined to retrieve data. Most relational systems rely on indexes to improve performance for data retrieval. An **index** is a file that contains the key values sorted with a link to the rest of the data row. By sorting the data, items can be located with a binary search (or better). A binary search splits the data in half at each pass. Think about an alphabetical list of names. To find a name (say Smith), start in the middle. If the name there is less than Smith, discard the entire first half of the list. Split the remaining names in half and look at the middle again. Continue the process until the desired name is matched, then use the index pointer to retrieve the rest of the data row. Indexes are a fast way to improve retrieval performance. Searching through a million entries sequentially would take from 1 to 1 million data retrievals (averaging 500,000) without an in-

**Figure 3.2**

Relational tables and indexes. All key columns have indexes to improve performance. To improve queries, other columns often have indexes. More indexes mean faster queries. But inserting one row can require updating dozens of indexes, slowing down transaction performance.



dex. With a binary index, any entry could be found within 20 lookups because  $2^{20}$  is greater than 1 million.

## Indexes

As indicated in Figure 3.2, indexes are commonly used for key columns in relational databases. If managers want to create subtotals by other columns, such as Item Category, Employee name, or Customer City, these columns are typically indexed as well. The result is that each table could have many indexes. Adding a single row of data then requires updating every one of those indexes. The indexes are relatively efficient, but at various points, adding data requires restructuring large portions of the index. The result is that indexes are useful for improving the performance of retrieving data, but indexes slow down data storage and updates needed for transactions. This conflict is what leads to the need for a new storage method.

For large databases, the solution typically involves creating a new database—designed just for OLAP. This **data warehouse** often retrieves data from multiple sources, cleans it up, and places it into a completely new storage structure. Data within the warehouse does not change constantly. Instead, it is extracted from the relational sources on a periodic basis and the warehouse is updated in bulk. At that time, indexes can be rebuilt one time, making it ready for any type of data retrieval and analysis.

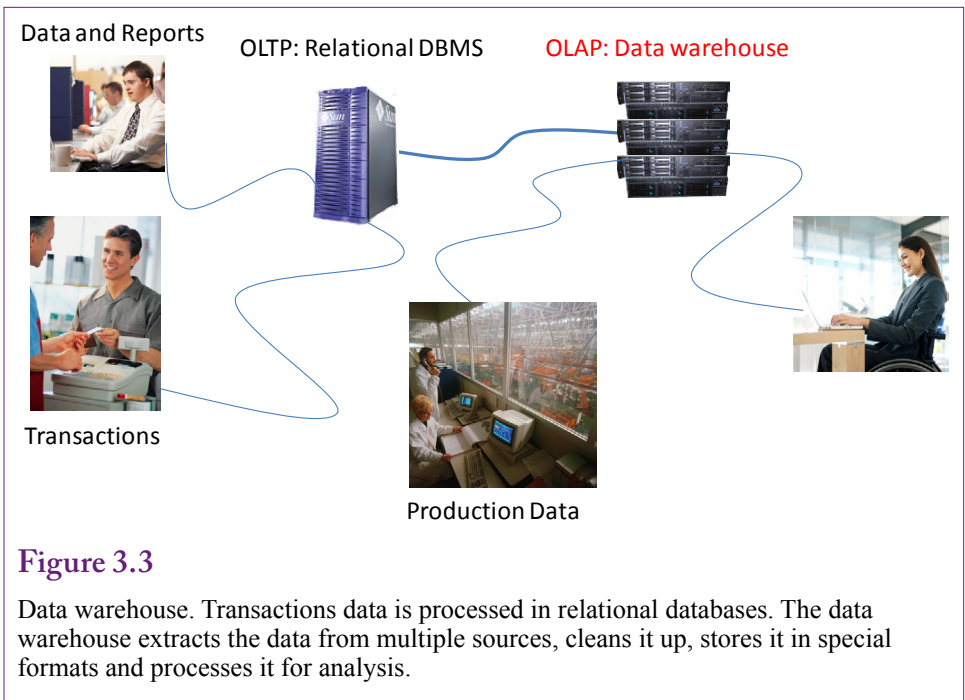
## Data Warehouse

The main issues in OLAP design and construction are examined in the next section—particularly applied to Microsoft’s tools. However, some basic principles apply to all of the tools and some questions have to be answered before any specific tool is selected. A data warehouse is a database specifically designed for data retrieval and analysis. Typically, it has a special design. It usually contains a copy of the data taken from a relational OLTP database. Figure 3.3 illustrates the basic concept of collecting the relational data into a central data warehouse.

Professional arguments still exist over how to handle the tradeoffs between storage and retrieval of data. Some experts and DBMS companies believe that the relational model with efficient indexes can handle most problems. Others believe data needs to be moved to a completely separate data warehouse to be efficient. One drawback to the warehouse approach is that the data being analyzed is not live—it is updated at certain intervals. Updates once a day are common, where the warehouse is reloaded and rebuilt overnight. Some companies update the data more often—perhaps a couple of times a day. Others delay as long as a week. The timing depends on the data and the needs of the managers. The question of whether data has to be live is an important one to answer when starting the process of designing an OLAP system. Microsoft’s Analysis Services has the ability to store data using different methods so it can handle live and warehouse data at the same time—but performance is slower with relational data.

An intermediate solution to a full data warehouse is the use of materialized views. A **view** in a relational database is a saved query. When run, the query retrieves the data defined by the SQL statements from the desired tables and displays the results. A view is dynamic—it rebuilds the joins and queries the raw data each time it is executed. A complex view with many joins can take a long time to run—in an OLAP context, views are often too slow—taking many seconds or minutes (or worse) to retrieve data. And each time a user wants the same data, the

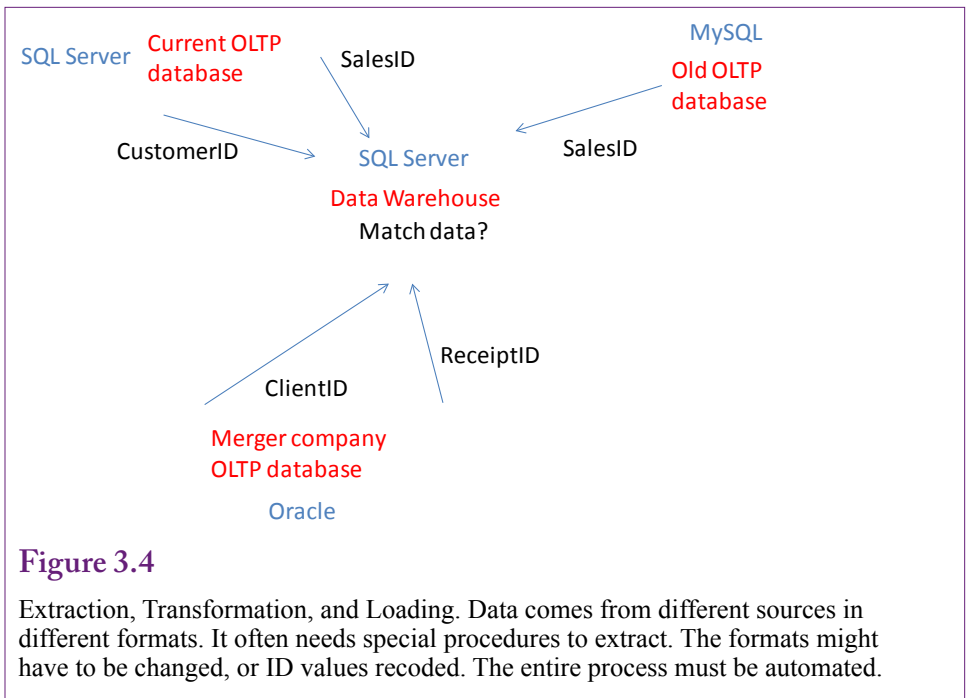




entire query has to be re-executed. Oracle, followed by most other DBMS vendors, introduced the materialized view. A materialized view starts as a query but saves the resulting rows as a separate table. Indexes can be built for the materialized view (but not for a regular view) because it contains the actual data. The data in the view has to be updated periodically to collect changes from the underlying tables. In a sense, a materialized view is a miniature data warehouse within the relational database. It is particularly useful for retrieving data that does not change often, but is heavily used. For example, customer and employee names are relatively stable. Product categories and descriptions might change for a few items, but the majority remains constant for long periods of time. All of these examples are items that might be used in data analysis—particularly for look up values and subtotals. If they are built into a materialized view, the resulting query can be significantly faster.

### Extraction, Transformation, and Loading

One of the biggest challenges in creating a data warehouse is the need to build all of the connections and transfer functions that collect the needed data. These data transfers are executed on a regular basis so they need to be completely automatic. It is not a question of cut-and-paste data from one source to another. The system needs to be automated. Figure 3.4 shows that data can come from many different sources—even different types of databases. Business mergers are particularly challenging in terms of combining data because terminology, data format, and ID values might be radically different. Even older data often becomes a problem. Managers want to use the old data for comparisons, but it often has to be redefined to match the new data. It is easy to underestimate the amount of data an organization contains and the thousands of ways it can be messed up—until you sit down to define how to collect it all and transfer it into the data warehouse.



**Figure 3.4**

Extraction, Transformation, and Loading. Data comes from different sources in different formats. It often needs special procedures to extract. The formats might have to be changed, or ID values recoded. The entire process must be automated.

**Extraction, transformation, and loading (ETL)** is the term used to describe the process of collecting data from various sources and transferring it into the data warehouse. Because the updates have to be made every day or more the process must run automatically and unattended. Consequently, all of the data sources and transformations must be written down and procedures created to handle the various tasks. All of the steps and code have to be tested and debugged to ensure they work perfectly. The process of designing the ETL can take several months of effort. Even if all data is held in a single enterprise resource planning system with consistent definitions, it takes time to locate the exact data and ensure the proper formats and security permissions are established.

SQL queries are useful tools for transforming bulk data. For instance, SQL can be used to alter ID values with a single command. A useful trick if data from two sources has overlapping ID values and one of them needs to be shifted before merging the data. SQL is also good at identifying unmatched data—such as old product ID values in a sales item table that no longer have records for matching products.

The ETL process almost always requires database programmers to write the transformation code, establish the database connections, and validate the security controls. These programming and administrative tasks are beyond the scope of this book. If you are planning a data warehouse project, talk to the programmers early, particularly before any products are purchased.

## MOLAP, ROLAP, and HOLAP

SQL Server Analysis Services can use one of three methods to store data in the data warehouse. **Multidimensional OLAP (MOLAP)** is the preferred storage method for most applications. MOLAP storage extracts the data from its source, pre-builds most joins, and writes the data into a new format with several indexes.



Retrieving data from MOLAP is faster than the other methods. The new physical format closely matches the logical structure of cubes described in the next section. **Relational OLAP (ROLAP)** leaves the data in the source database. The **meta-data** or descriptions of the data are stored in the Analysis Services database. Because the data remains on the original source, the queries always produce up-to-date information from the live source. However, queries can be considerably slower than the MOLAP approach. On the other hand, MOLAP systems require processing time at the start to transfer the data and create the basic structures. Generally, the effect of this processing time can be minimized by performing it at night or when the data and systems are lightly used. The third method is **hybrid (HOLAP)** storage. With this approach, the original data remains in the relational database, but aggregations or subtotals are computed and stored in the warehouse. HOLAP might seem like a useful compromise, but its query performance has been reported to be similar to ROLAP, so it is probably not very useful.

One nice feature of Analysis Services 2008 is that data can be defined in partitions and each partition can have its own storage method. If the application needs live data, leave that portion in ROLAP structures. Put the rest of the data into MOLAP partitions. The logical data is treated the same and the data is easy to combine even from multiple sources on different partitions.

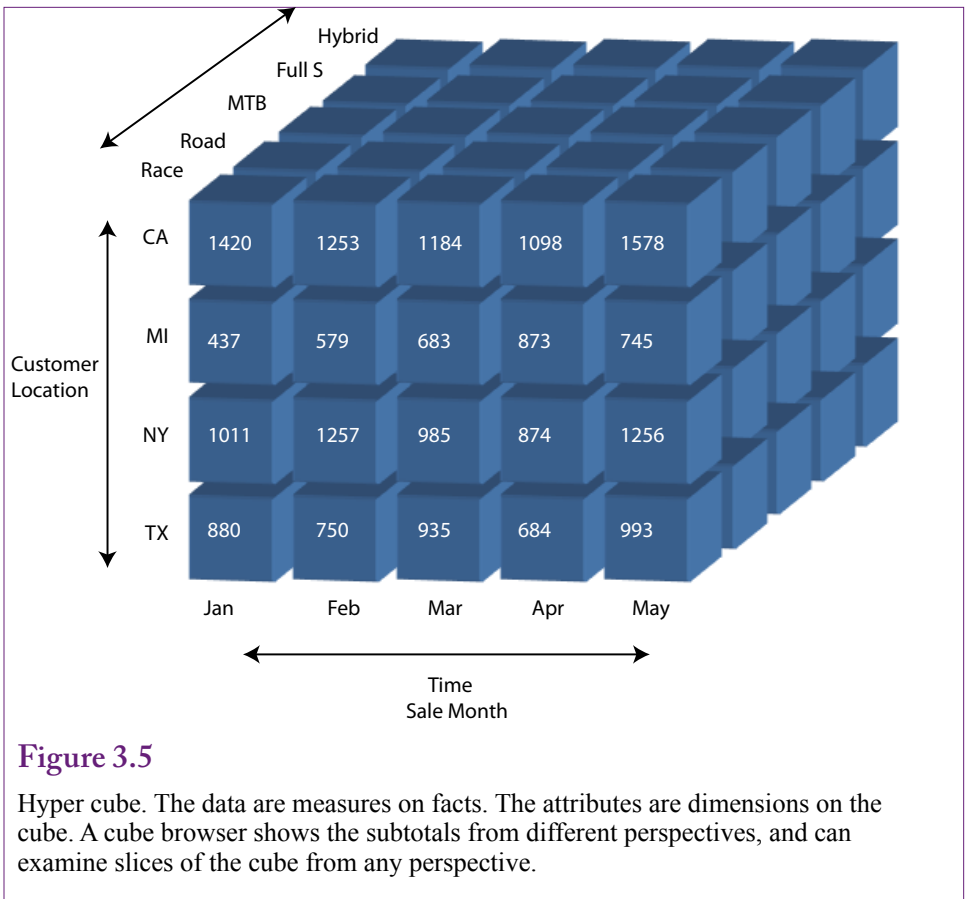
For the major elements of data needed in a project, someone has to decide if live, absolutely-current, data is needed. If so, this data is placed into ROLAP partitions. Other data should be placed into MOLAP partitions. Later, performance specialists can tweak the system to improve overall performance. Realistically, unless a project has at least 20 million rows, it is small enough that the choice of storage method is not going to have much effect on performance. All of the projects used in this book are small enough to easily use any of the storage methods—but MOLAP is the obvious choice because none of the data needs to be live.

## OLAP Design

---

**What is the process for designing and creating a data cube?** Perhaps the first question is why OLAP browsing typically refers to a cube. The two questions are related. Figure 3.5 shows a sample three-dimensional cube. The data displayed on the cube are **measures** on some **fact**. The attributes on the sides are **dimensions**. The dimensions consist of factors that are important to decision makers. The cells in the cube show subtotals for the values of the two intersecting dimensions. A hypercube can have any number of dimensions, but it is hard to draw anything above three dimensions. In fact, most people can read only two dimensions at a time in a basic table. So, most OLAP tools provide a cube browser that is essentially a dynamic table. Analysts can choose the dimensions by dragging from a list. If the values in the dimension are small enough, they can be stacked in the table. Filters can be used to effectively slice the cube—changing the table values to show a single slice of the cube. For example, setting a filter to Month=Jan could be used to show a table with Customer location against Category for just that month. Dynamic changes and quick responses are key features of cube browsers. No one wants to wait minutes for a cube to update when a dimension is changed.

Figure 3.6 shows the Analysis Server cube browser with data from Rolling Thunder Bicycles. It currently displays totals for the dimensions of Model Type and Location of the customer. The date of the purchase is set as a filter dimension, but it currently displays data for all years. The dimensions can be changed simply by dragging them from the list in the left panel onto one of the three spots (row,

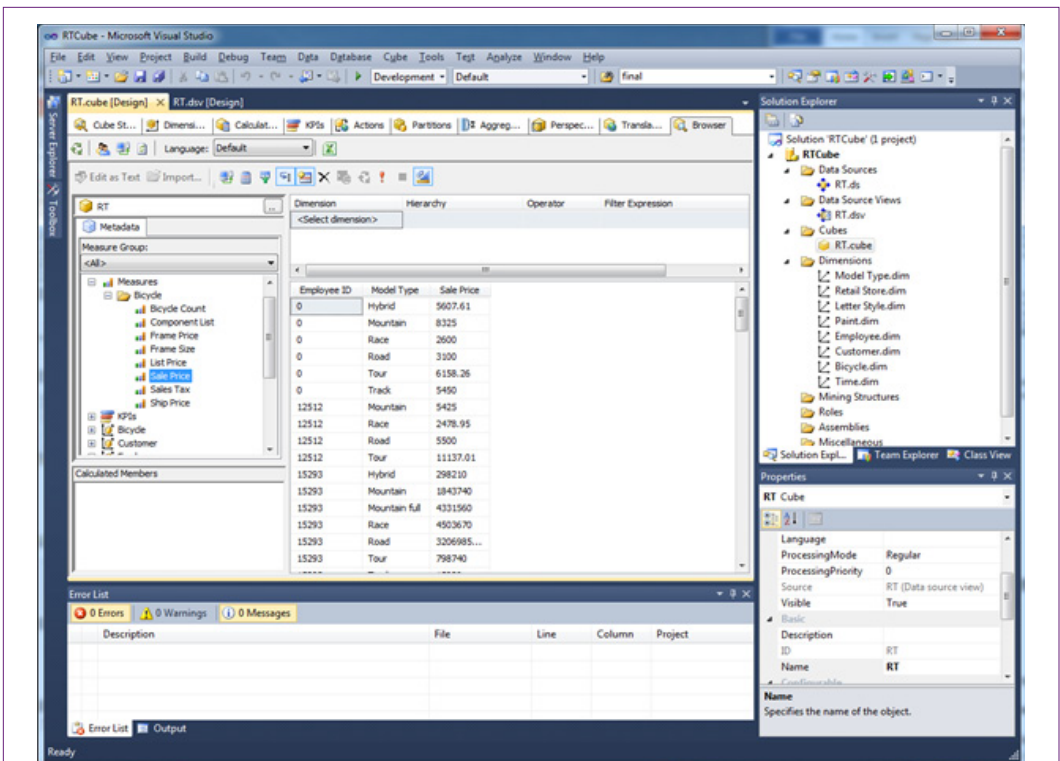


column, or filter). Dimensions are removed by dragging the off the table. It is possible to put multiple dimensions on the rows, but the table size quickly gets out of control and hard to read with too many values.

A dimension can be organized into a **hierarchy** or levels. In the example, location is a hierarchy from country to state, city, and ZIP code. Similarly, the date is a hierarchy from year, quarter, month, and date. These hierarchies have to be created within the dimensions, but once defined the cube browser handles the hierarchy automatically.

Notice that the left panel also contains a list of possible measures that can be used within the table. The cube currently displays subtotals for Sale Price. Other measures include List Price and the count of the number of bicycles. The measures can be dragged on or off the center of the cube. It is possible to display multiple measures at the same time side-by-side. However, the table can quickly get out of control when too much data is placed on it.

The default computation for values is to sum them. This computation can be changed to one of several options—but it cannot be changed from within the browser without some additional configuration. This restriction makes sense when you understand how the OLAP tool works. To provide near instantaneous response, the totals are computed only when the cube is processed. Browsing simply retrieves the values that match the displayed dimensions. So the next step is to learn how to setup and create the basic cube.



**Figure 3.6**

OLAP cube browser. Dimensions can be dragged from the left onto rows, columns, or the filter at the top. The data totals in the table update automatically as the dimensions are changed. Individual attribute values can be deselected with the drop-down arrow by the dimension name.

## Facts and Dimensions

The cube image provides the foundation for the structure of how OLAP works. The key to understanding OLAP is to focus on the measures and dimensions. In fact, the multidimensional storage systems use this structure to organize the data for fast retrieval. The first step in designing an OLAP system is to identify the facts that need to be measured. In business applications, the facts typically involve sales, which are generally computed as price times quantity. In the bicycle case, all of the bicycles are unique, so the quantity is always one and Sale Price is the relevant measure. In other cases, it will be necessary to create a calculated column that handles the multiplication. Those details are explained in the next section. Some problems have multiple facts, but business cases typically focus on the value of sales and perhaps the number of items sold or number of sales. At the moment, it is not important where the fact measures are stored. Even if the data columns are stored across multiple tables, the OLAP system will combine everything into one set of measures.

Once the facts have been identified, the attributes that might affect those facts need to be identified. These attributes become the dimensions of the cube. At this stage, it is important to identify all of the potential dimensions that manag-

Product	Customer
Category Color Size Manufacturer	Location: City Gender Age Experience
Store	Sale
Country Region City Department Salesperson	Date Time of day Discount

**Figure 3.7**

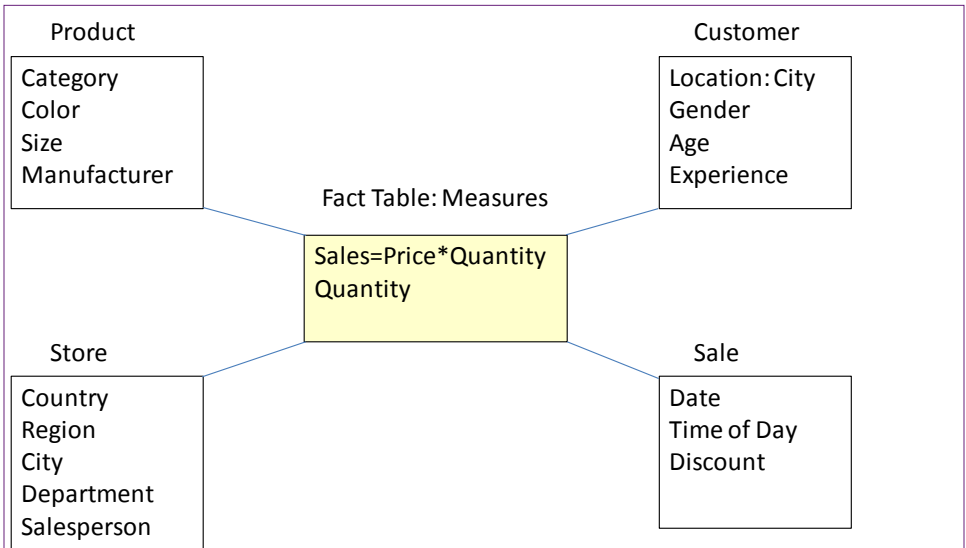
Common business dimensions. Look at tables and categories for ideas about which dimensions to include. Ultimately, all of the dimensions are combined and treated equally.

ers and analysts might want to consider. Avoid looking at the data in relational terms. Simply look at the existing data and decide which attributes would be useful dimensions.

Figure 3.7 shows that common business dimensions include product attributes, customer demographics, salesperson, region or store, and sale attributes. Product attributes often include items such as category, color, size, and manufacturer. Customer demographics might include location (City), gender, age, and experience. The availability of data will depend on the type of business and the level of contact with the customers. Large organizations will want to track sales geographically by country, region, and city. Many will want to track sales by department and by salesperson. The Sale itself typically includes a date dimension which sometimes is tracked down to time of day. Some organizations provide different types of discounts and need to track those. The point is to go through all of the data tables and identify anything that might be used to define a grouping or subtotal.

### Star Design

In the end, it is not important where facts and dimensions come from. The basic design in an OLAP system is the **star design**. Figure 3.8 shows an example of a star for the common set of business dimensions. The star name arises because the fact measures reside in the center and every dimension is connected directly to the fact table through single links—like rays emanating from a star. The star design is efficient for retrieving data because every dimension is directly tied to the fact measures. Pulling data from a relational database, the system often accomplishes this structure by duplicating key data. The data in a star structure does not have to be stored in relational normal form. Many systems improve performance even more by pre-computing all of the subtotals. The star configuration represents the MOALP or multidimensional storage. Data is stored according to the dimensions.

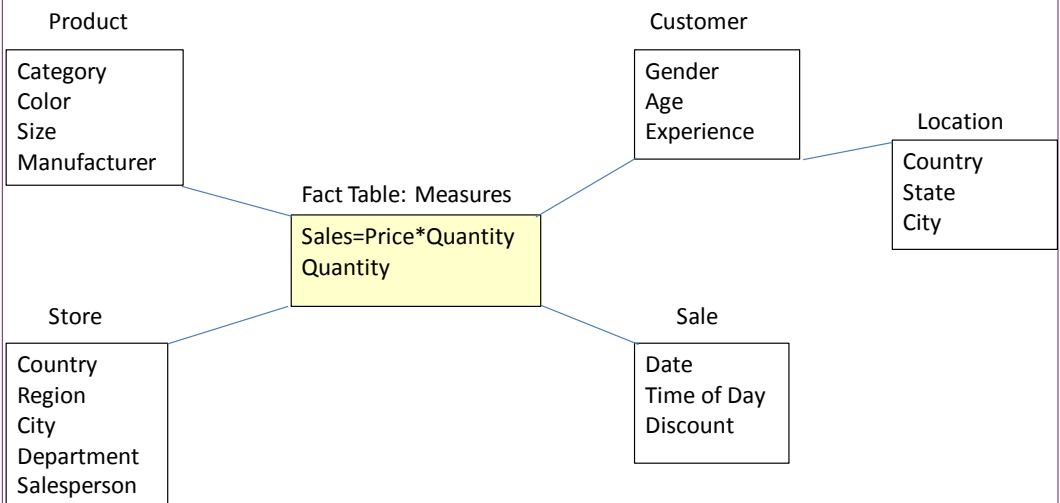


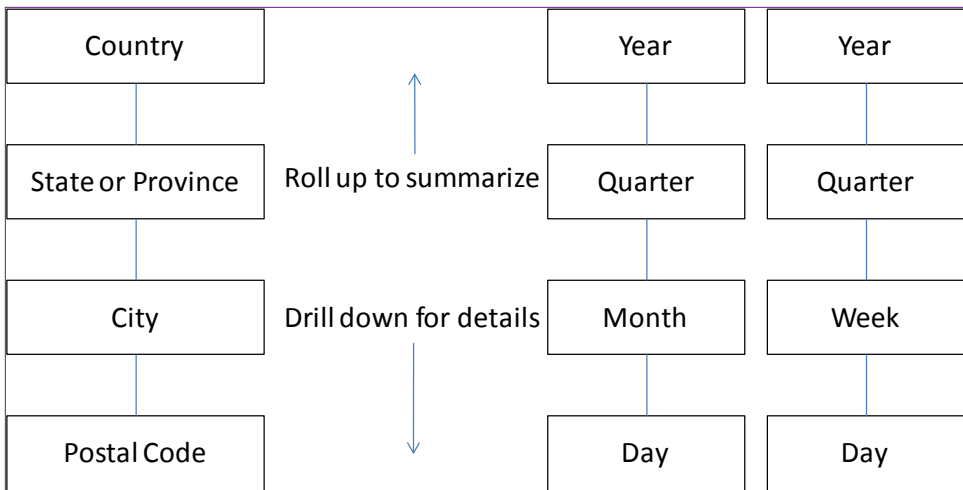
**Figure 3.8**

Star design. The fact table in the middle holds the measures. All other dimensions are one link away from the measures. The star design is often created by duplicating data. Keeping dimensions down to a single link improves retrieval performance.

**Figure 3.9**

Snowflake design. Dimension tables can have links to other tables, creating tables that are multiple links away from the fact measures. Complex snowflake designs require the OLAP engine to perform joins to obtain data and slow down the performance. Try to avoid them except for lookup tables.





**Figure 3.10**

Hierarchies. Common natural hierarchies are based on geography and time. Sometimes, as with dates, multiple hierarchies can be defined from the same data.

## Snowflake Design

Some OLAP systems support only the star design. Others provide more flexibility by allowing links to the tables holding the dimensions. Microsoft Analysis Services allows the creation of these links. Ultimately, these secondary links make the design appear more like a **snowflake** than a star. Figure 3.9 shows a simple example that links the City table to the Customer table. Throw enough secondary links into the picture and it begins to look like an idealized snowflake. The problem with these secondary links is that the OLAP engine has to build joins to retrieve the associated data. Although the query engine can build indexes, the links generally decrease performance—particularly with huge tables. The links often arise because of the way the data is stored in the relational OLTP database. In most cases, the OLAP tool supports methods to extract the data from the outlying tables and build it into a MOLAP structure. Even though the data might appear to be stored across multiple links, internally, it is denormalized and stored in a star structure. Still, it is best to avoid secondary links when possible. The example here uses it only as a lookup table, and that connection is handled later as a hierarchy.

## Hierarchies

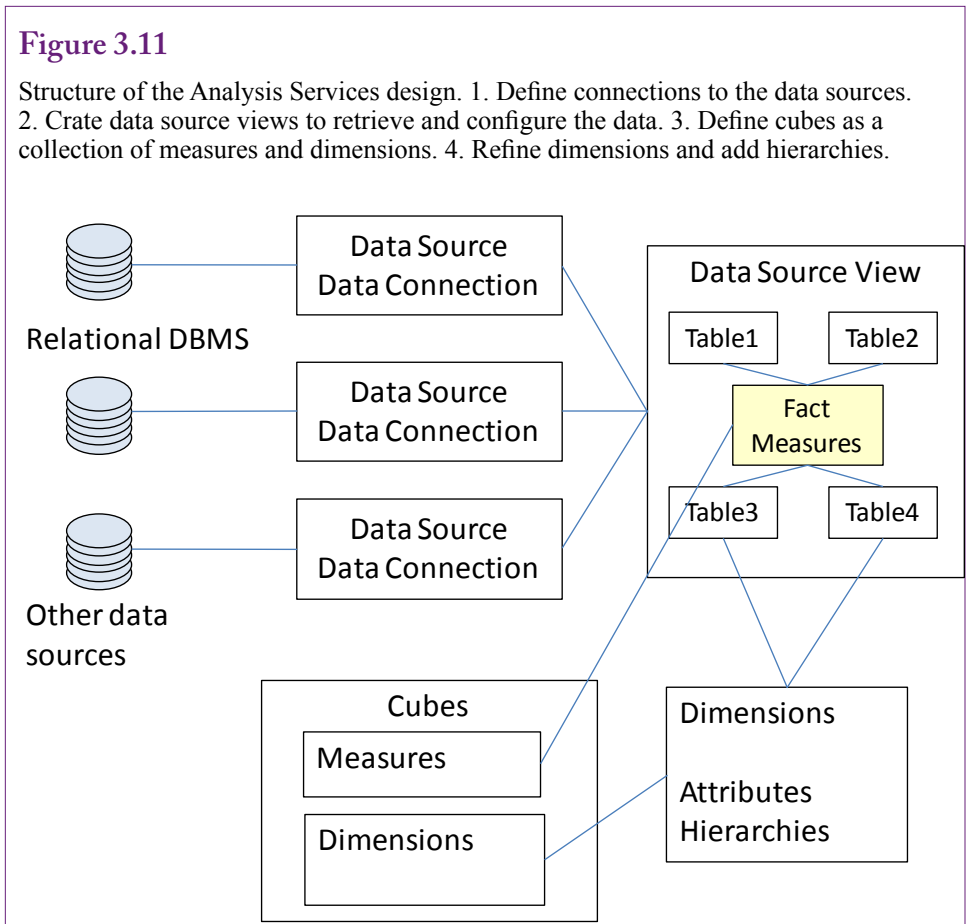
Hierarchies are dimensions that ultimately are perceived as a single dimension with multiple levels. People often use hierarchies to simplify problems. Looking at data by levels makes it easier to see the big picture and still provide the details needed to understand a system. Figure 3.10 shows an example of two common natural hierarchies: location and time. Notice that multiple hierarchies can be defined for the same set of data. Some retail companies use weeks to define sales instead of months, but analysts and managers might want both hierarchies for different purposes.

When hierarchies are applied to a cube, the browser displays buttons (often + signs) that enable the analyst to **drill down** to see the more detailed level. Similarly, detailed levels can be **rolled up** to see the totals for the higher level.

Microsoft Analysis Services has tools to help build common natural hierarchies, particularly for time and geography. It is also possible to build custom hierarchies, such as groupings of products, or employee teams. These hierarchies can only be created from existing data. If you know that certain hierarchies will be needed, be sure to include all of the related columns needed to define the hierarchy. For instance, an Employee table might include a ManagerID column to report a hierarchical reporting relationship, so ManagerID will have to be included in the dimension list. For workgroup teams, a separate table defining members of the teams will likely be needed.

## Creating a Cube with Microsoft Analysis Services

**How are OLAP cubes created using Microsoft SQL Server Analysis Services?** The basic concepts of OLAP cubes are relatively common. The process of creating one is quite different depending on the specific product. The steps with Microsoft's Analysis Services are straightforward, largely because of the use of wizards. The basic process is to build connections to the data sources,





define views and calculations, create an initial cube, and refine the dimensions and create hierarchies. Figure 3.11 shows the basic structure of the objects within the Development Studio for building OLAP cubes.

Although other tools exist, the most powerful tool to define and control the OLAP structure is provided with the **Business Intelligence Development Studio**. The BI studio is based on Microsoft Visual Studio, which is a design environment for many types of programming. The BI projects are built using a specialized set of templates. The studio should be installed on your workstation using the Client disk from the SQL Server installation package. Even if a workstation already has Visual Studio for other uses, the BI templates need to be installed from the SQL Server setup.

The data defined for OLAP cubes is also used later for data mining. The statistical analysis tools are embedded in the Development Studio and the data can be retrieved through OLAP views or directly from OLAP cubes.

## Data Sources

Almost any type of database or standard file can be used as a data source with SSAS. Microsoft has been building database connectivity systems for years, and most of them can be used to read data into the OLAP structure. One catch is that if multiple connections are going to be used, the first connection should be to a SQL Server database. Apparently, Analysis Services routes the connections to the other systems through the SQL Server database.

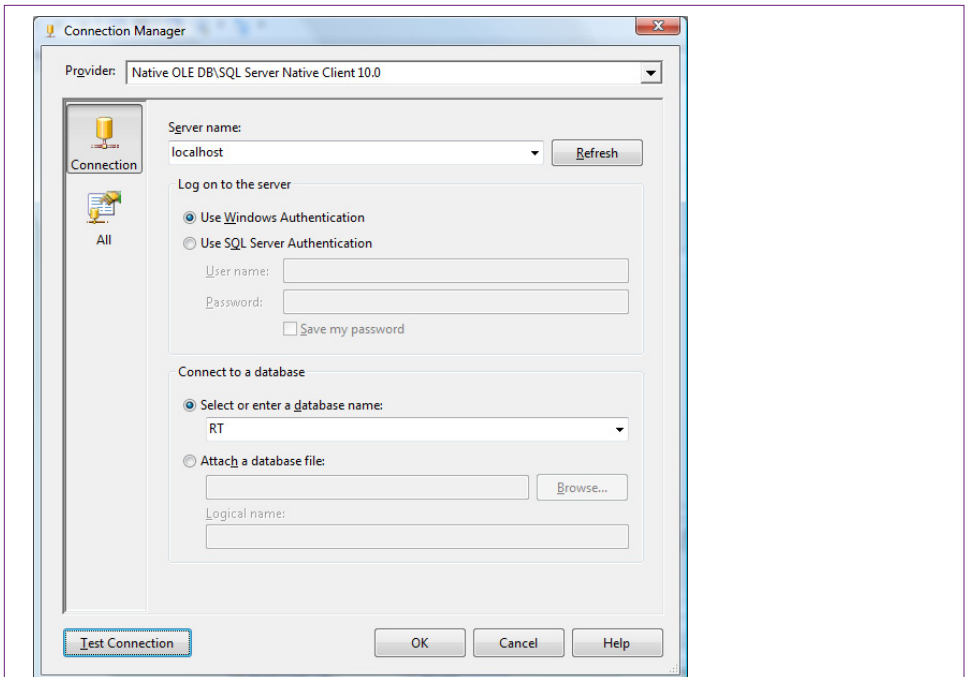
Start the BI tool and create a new project. Choose the Business Intelligence project type and select the Analysis Services Project template. All projects are created as a set of files in a folder on the local computer. Most of the files are XML text that describes the specific objects created in the project. A solution (.sln) file holds links to all of the files. When the cube is processed, the information is compiled into a specialized format and sent to the specified Analysis Server. For now, you simply need to choose a name and location for the project. If possible, stick with the default location. Create a name that describes the project that can be recognized later.

Initially projects are empty and Visual Studio displays a list of tasks in the Solution Explorer. The first task is to define a connection to a data source. A data source is typically a relational database. It might be SQL Server running on a central server or even on your local computer. The sample files for this book have scripts to build them on SQL Server, but the data also could be loaded into an Oracle or MySQL database. In a real-world project, the database should be running on a separate server.

### *Relational Database*

Most OLAP projects retrieve data from a relational database. SQL Server is certainly easy to connect to with SSAS, but almost any common DBMS has drivers for establishing connections to Microsoft tools. To connect to an Oracle DBMS, you should obtain the OLE DB client from either Microsoft or Oracle. OLE DB is a Microsoft standard for establishing connections with database systems and the common DBMSs have providers—but you might have to install them separately. The providers will also have to be installed on the server running the Analysis Services engine.

To create a data source, right-click the Data Sources entry in the Solution Explorer and choose the New Data Source option. Click through the startup forms



**Figure 3.12**

SQL data source configuration. Choose the correct server. If possible, use Windows authentication. Select the correct database and always test the connection.

and click the New button. Choose the provider from a drop down list—it defaults to the latest OLE DB driver for SQL Server. The configuration form shown in Figure 3.12 is standard for SQL Server connections. The form changes slightly depending on the provider selected. For SQL Server, choose the name of the server. If possible, use Windows authentication. Eventually, for production systems, it is better to create a separate SQL Server login. For now, select the appropriate database and always be sure to test the connection. Follow the steps to accept the choices and finish the wizard. When finished, an entry will appear under the Data Sources in the Solution Explorer.

At a minimum, the logon connection specified (either Windows or SQL Server) must have read permission on the tables or queries needed for the project. These permissions are set within SQL Server (or whatever DBMS is holding the data). Later, to create hierarchies within dimensions, the logon user will also need permissions to create new tables. Hierarchies are stored within separate tables. Time dimensions are the only exception—SSAS provides a mechanism to store time dimensions on the OLAP server for cases where the logon user cannot be given the permission to create a table.

### *Multiple sources*

Multiple data sources can be added to a project. Simply follow the same steps to add new connections. Connections can be made to different databases, even those stored on different servers and different DBMSs. Always remember to test the connections as they are built—it will be difficult to identify the cause of problems later if the connection fails.

Think about what it means to use multiple data sources. The OLAP engine provides a level of abstraction. As long as Analysis Services can connect to a database and retrieve data, the data can be included in the project. In essence, this trick enables you to integrate data and even define relationships across data tables from different databases. Once the database connection is defined, any accessible data can be used in the project. From this point forward, you no longer care where the data is stored. The OLAP engine handles the details of retrieving the appropriate data. The process of connecting tables is handled in the Data Source Views, but all tables are equal at that point and the source does not matter.

## Data Source Views

A data source view in OLAP has some similarities to queries or views in a database engine. The purpose of a view is to provide a perspective on a subset of the data. It is also used to join tables and define new computed columns. One big difference is that the data source view is required for OLAP. The data source view serves as a buffer and it is not possible to retrieve data directly from a table. To build a cube or to retrieve data for data mining, all of the data needed must be defined in a data source view.

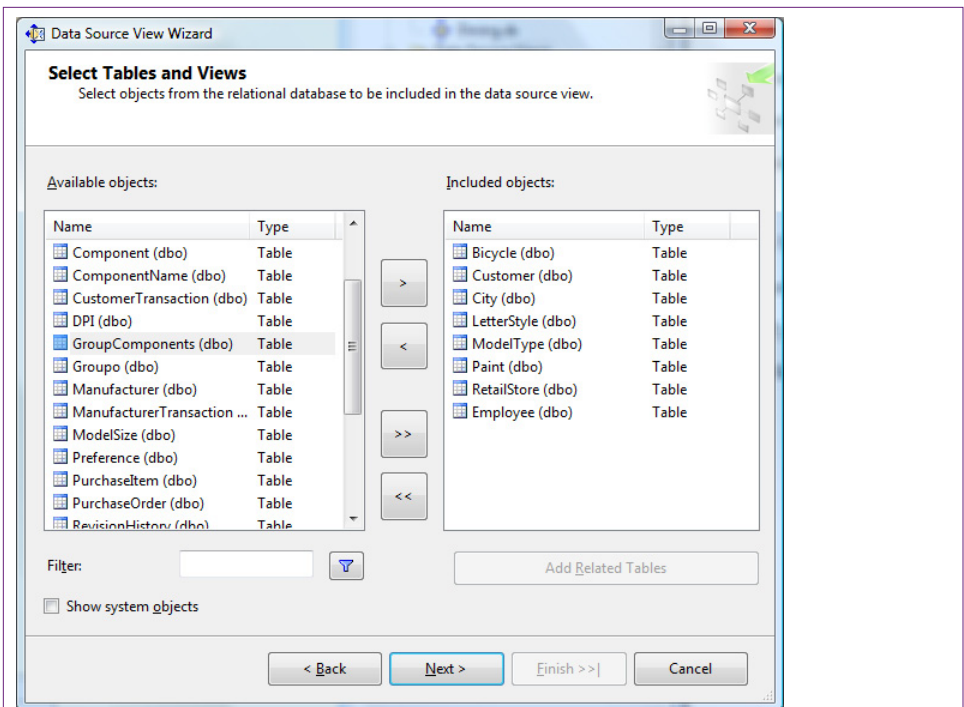
A project can contain multiple views. Some projects are built by cramming all data tables into one view. With even a few dozen tables, this approach quickly gets messy. Sometimes there is no choice and all of the tables are needed together. However, in many cases, a better approach is to create separate data source views for each particular problem. Separate data views also make it easier to assign security. For example, a different view can be created for each specific group of users or analysts. Access can be controlled by assigning permissions to the views.

If it really is necessary to include dozens of tables in one data source view, the display can be cleaned up by creating separate diagrams. A diagram shows the relationships among tables and it is easy to create as many diagrams as necessary within one data view. The diagram does not affect the use of the data—which is good and bad. The good part is that the displays can be simplified and it is easier to work with a few tables at a time. All of the data is still visible later when building cubes and data mining models. The bad part is that the division only applies to the diagrams so every table and dimension shows up in huge lists when building cubes and selecting data. A diagram only applies to the display of the data source view on the diagramming window.

To reduce the overall number of tables and dimensions that groups work with, it is necessary to create separate data source views. Whenever possible, separate views should be used. Just be careful to give them descriptive names that everyone will recognize.

### *Creating a Data Source View*

Creating a new data source view is straightforward. Right-click the Data Source Views entry in the Solution Explorer and choose the option to add a new one. The wizard will ask you to select the data source, although it has an option to create a new one if you skipped that first step. The second step is to select the tables and views (queries) from the relational database. Figure 3.13 shows the basic process. For complex problems, it will be helpful to study the relationship diagram from the underlying database. If the tables in the underlying database are linked indirectly through other tables, all of the linking tables must be included in the view to be able to reconstruct the relationships. If you are uncertain about which tables



**Figure 3.13**

Selecting tables for a data source view. This view focuses on the bicycle and the sale. Most of the tables are lookup tables used for the dimensions.

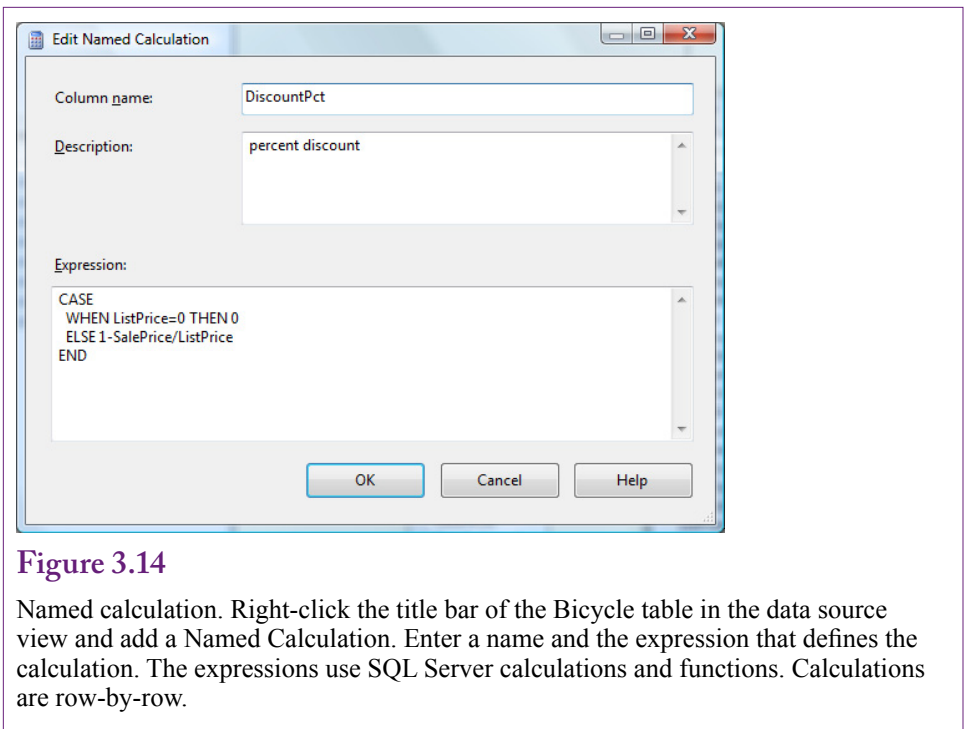
are required, you could start with the table that holds the facts then click the button to “Add Related Table.” The drawback to this approach is that it is likely to include many tables that you do not necessarily need in one view. But sometimes it is easier to add everything and remove the ones you do not need.

The last step is to name the data source view. Be explicit, use extra words. It is important to use a name that everyone will recognize later. A short name with abbreviations might seem cute now, but a year from now, it might be unrecognizable. This particular view might be called RT Bicycle Sales and Customers. But do not worry, views can be renamed later.

### *Creating a Named Calculation*

Sometimes existing columns are not in the correct form or do not include the exact data needed for a cube or for data mining. Two of the most common situations are: (1) Common business items such as  $\text{Value} = \text{Price} * \text{Quantity}$  or  $\text{Profit} = \text{Revenue} - \text{Cost}$ , and (2) Creating concatenated columns to form a single primary key. The data mining routines all require a single column as key and it is often necessary to build one within the data source view. For now, it is useful to learn how to build a common business calculation.

In the Bicycle case, each bicycle is unique so there is no need to multiply price and quantity. However, the Bicycle table contains a ListPrice and SalePrice. The difference between the two is the discount amount. Some managers might want to explore the cube with respect to the discount values.



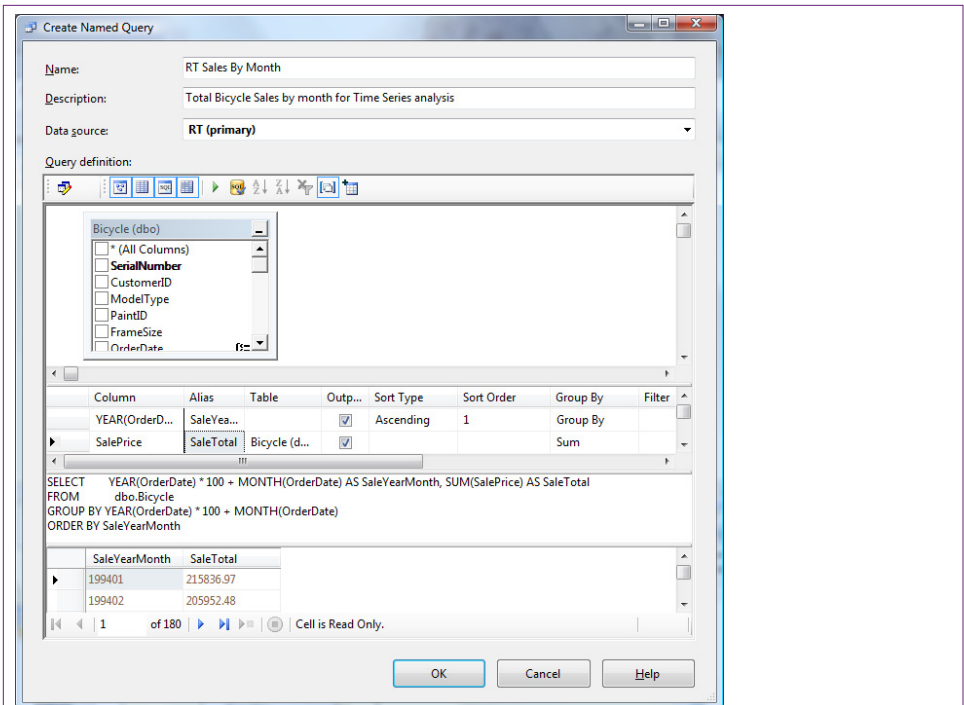
**Figure 3.14**

Named calculation. Right-click the title bar of the Bicycle table in the data source view and add a Named Calculation. Enter a name and the expression that defines the calculation. The expressions use SQL Server calculations and functions. Calculations are row-by-row.

A row-by-row calculation is created within a table in the data source view and is called a **named calculation**. It is equivalent to creating a calculated column within an SQL query. In fact, it uses the same functions and computations that are used in SQL queries. To create the calculation for the discount, right-click the top of the Bicycle table in the view and select New Named Calculation. It is important to select the title bar of the table; the option will not appear if you select the middle of the table.

As shown in Figure 3.14, enter a descriptive name. Enclose it in brackets if it contains a space or reserved character. A longer description can explain its purpose. The expression is the calculation. Discounts are typically evaluated as a percentage  $(ListPrice - SalePrice) / ListPrice$ . Expressions can be any SQL Server calculation or function. To obtain help building expressions, it is often easier to create the expression in a SQL Server query window, test it there, and copy it back to this OLAP expression box. Either way, once the expression is created, it is important to test it within the data source view. Right-click the main body of the table and choose the Explore Data option. Verify the computed column.

One problem with dividing is that it is important to test for zero values of the divisor. The SQL CASE statement is useful to create the two conditions needed (zero and not zero). If ListPrice is zero (most likely missing), set the discount to 0, otherwise define the percentage as  $1 - SalePrice / ListPrice$ , which is a slightly simpler expression of the percentage. As an advance warning—do not attempt to use this calculated column in a cube just yet. A section below on calculations explains an important issue.



**Figure 3.15**

Create a named query. The edit window is similar to creating a query in SQL Server, but the named query is stored in the data source view. Use the icons or right-click to add tables and add Groups to create subtotals. The ORDER BY clause is used only to test the query. It must be removed to save it.

### *Creating a Named Query*

It is also possible to create entire queries within a data source view. A **named query** is similar to queries created within SQL Server but they can use all of the tables within the data source view. If all of the tables come from a single SQL Server database, it does not matter if the query is created and saved in the underlying database or in the data source view. However, the strength of creating a named query is that it can use data from any table from any data source—so it can easily combine data from diverse locations. This feature will not be used for the example in this section, but remember that it is available.

Data mining problems often require data to be in specific formats. For instance, to perform a time series analysis on monthly data, the data totals by month need to be computed before starting the time series analysis. The time series tool does not have an option to compute subtotals—they have to be set up ahead of time in the data source view. The data mining tools also require a single column as the primary key, so it is often necessary to use a query to concatenate separate columns into a single key.

Creating a named query is similar to creating a query in SQL, but it is handled in the data source view. Right-click an open location in the data source view and choose the option to add a new named query to open the editor window. Figure 3.15 shows the edit window, which is similar to editing queries in SQL Server.

You can work with SQL directly or use the graphical interface to help build the query. Use the icons or right-click the table window to show the list of tables and to add the Group By options. Always test the query before saving it. In the example, the ORDER BY clause is used to help evaluate the results; but it must be removed before the query can be saved. The SQL Query is:

```
SELECT          YEAR(OrderDate) * 100 + MONTH(OrderDate) AS
SaleYearMonth, SUM(SalePrice) AS SaleTotal
FROM            dbo.Bicycle
GROUP BY YEAR(OrderDate) * 100 + MONTH(OrderDate)
```

Notice the way the YearMonth column is created in this query. Because a single column will be needed as the primary key for time series analysis, the Year and Month functions are used to generate dates of the form: YYYYMM. Multiplying the Year by 100 shifts it to the left by two places to create space for the month number. Also note that views cannot contain the ORDER BY clause. It is sometimes useful for testing, but needs to be removed before saving the named query.

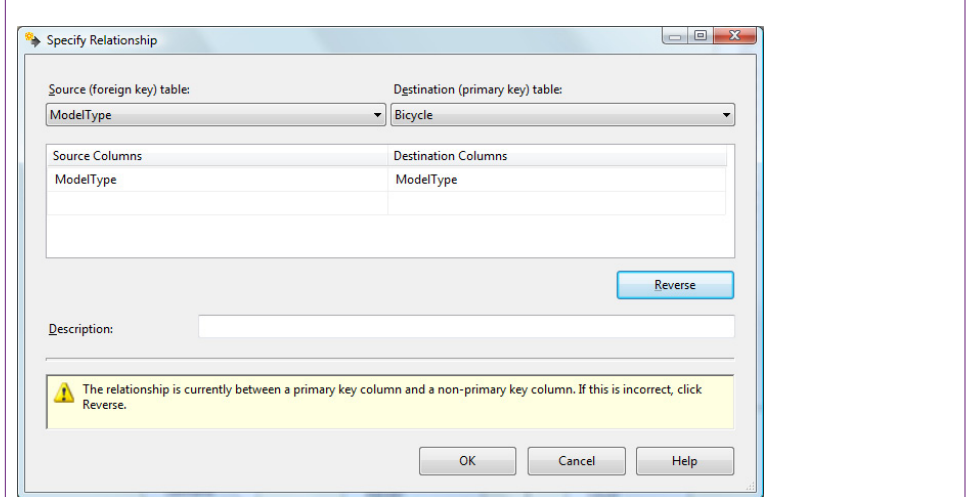
Once the named query is created, it is important to set the logical primary key for the new query. On the data source view, right-click the column name (SaleYearMonth) and choose the option to Set LogicalPrimaryKey. Defining the primary key within the named query sets the default values for any tool that uses that query.

### *Creating a Relationship*

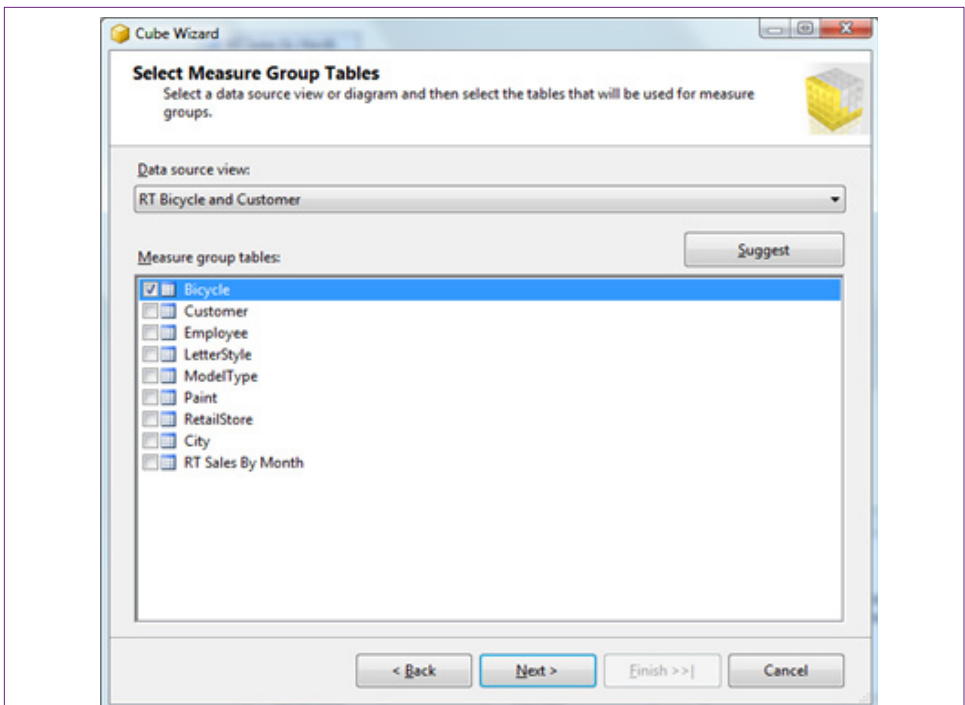
Most of the time, when tables are imported from data sources the relationships between the tables are also imported. However, when creating named queries or importing tables from multiple data sources the relationships will not be built automatically. In those cases, it is straightforward to create the relationships manually.

**Figure 3.16**

Creating relationships. When using drag-and-drop to create a relationship, drag from the foreign key (many) table and drop onto the primary key in the related table. Relationships created the other way need to be corrected by pushing the Reverse button.







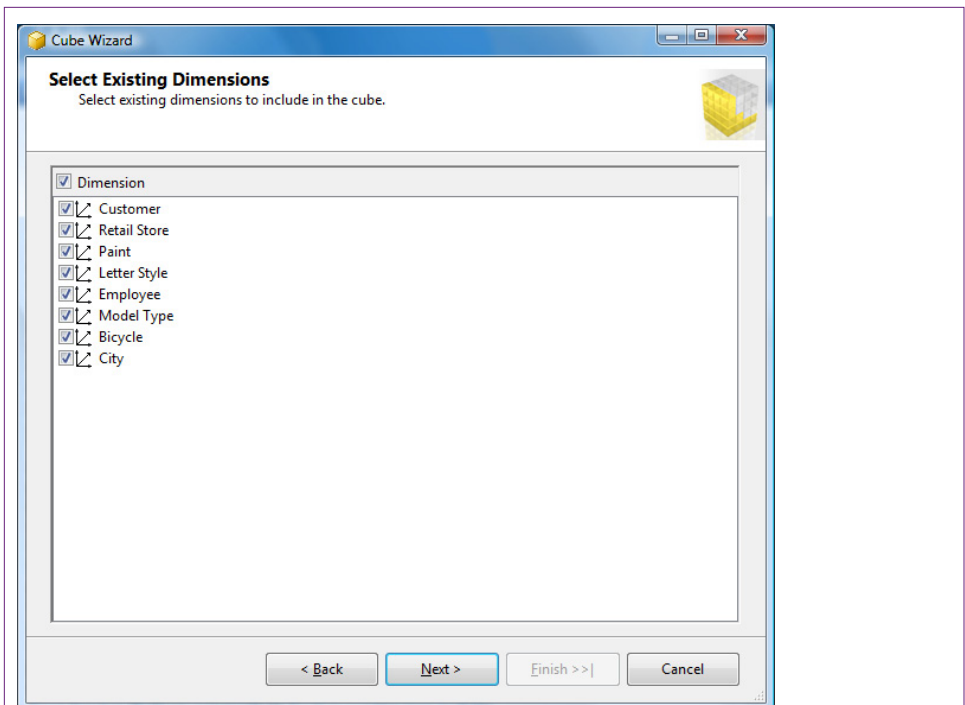
**Figure 3.17**

Select tables that hold measures. First select the appropriate data source view in the drop-down list. Then choose the Bicycle table which holds the sales information for the RT case.

Relationships can be created by right-clicking the data source view window and choosing the option to create a new relationship. However, it is easier to drag a column from one table and drop it on the matching column in the related table. As shown in Figure 3.16, the big catch is that you need to drag from the foreign key (many) side of the relationship and drop onto the primary key of the related table. The relationship editor is not smart enough to identify the relationship automatically. If you try to create the relationship the other way, the warning message explains the problem. Click the Reverse button to switch the direction. If that does not solve the problem, either the keys on the tables are wrong, or the wrong columns have been selected for the relationship.

## Cubes

When the data source and data source view have been created, a hypercube can be defined with the selected data. The main steps in creating a cube are to choose the columns for the fact measures and select the columns that can be used as the display dimensions. If the data source view is built correctly, these steps are straightforward with the help of the cube wizard. When the cube has been built, it can be explored with the browser. Next, more complex dimensions can be created by building hierarchies.



**Figure 3.18**

Dimension tables. Tables in the data source view that are linked to the measure tables are automatically included as dimensions. The City table can be removed later when the location hierarchy is created.

### *Wizards*

Start the wizard by right-clicking the Cube entry in the Solution Explorer and choosing the New Cube option. The first screen has options for selecting the data. The default choice of using existing tables is generally the best way to start. As shown in Figure 3.17, the next step is to choose the data source view from the drop down list. The view should have been created to hold both the measures and dimensions. The main purpose of the form is to select the tables that hold the facts or measures that will be used in the cube. The RT case uses the Bicycle table to hold standard sales data, so select that table.

On the next screen, the wizard automatically selects all of the columns in the chosen table to use as measures. Because only a couple of the columns are actually measures, click the checkbox at the top to deselect all of the columns. Then select the three useful measures: List Price, Sale Price, and Bicycle Count. The last entry (Count) is not an actual column, it is a measure added automatically by the wizard. It essentially counts the number of rows. It is useful in cases such as the Bicycle table where no quantity exists. Similar situations exist for table on people, such as customers or employees. For more traditional problems containing a Quantity column, including the row count could lead to confusion, so it should be left out.

The next step is to choose the dimensions that can be used in the cube. The wizard automatically selects the tables linked to the fact table. In the RT custom-

er data source view, the dimension tables include: Customer, Model Type, Retail Store, Paint, Letter Style, Employee, City, and Bicycle. Essentially, those tables are lookup tables that provide a description for the ID value stored in the Bicycle table. Recall that the City table was linked through the Customer table. Later, it can be removed when a location hierarchy is created to handle the lookups. Also, notice in Figure 3.18 that the Bicycle table is included by default because it is the measure table. This table needs to be included with the dimensions because it contains the sale date information needed to explore sales over time. If the list lacks some tables that you feel should be included, you should cancel the wizard and return to the data source view to ensure that all of the needed dimensional tables are linked to the measures table. At this point, do not worry about hierarchies because they will be configured after the basic cube is built. The final screen summarizes the cube and enables you to enter a descriptive name.

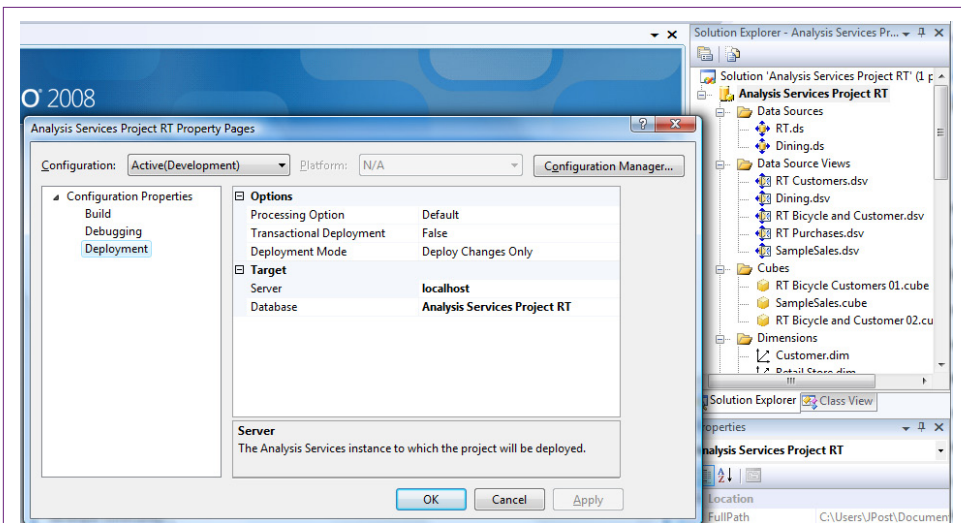
That is the entire process needed to create a cube. The cube now needs to be transferred to the analysis server and processed. Then it can be browsed. It might not be perfect yet, but it is a good practice to test it to ensure it processes correctly and that it has the fundamental data needed.

### *Deployment and Processing*

Most of the work to this point has taken place on the client workstation. The data connections, data source views, and even the cube definition are currently stored as definitions in XML files. The cube with data does not yet exist. To use the cube, the definition has to be transferred to the Analysis Services server and processed. Processing consists of building the dimensions and pre-computing most of the subtotals for each of the dimensions. Executing this step is straightforward: right-click the new cube name in the Solution Explorer and choose the Process option. The cube will be deployed to the server and a job schedule created to process the data. In most cases, for testing purposes, just click the Run button on the processing form to start the computations. In a production environment, the cube could be huge and take large amounts of resources and time to process. The processing form has options to estimate the impact on resources and limit the number of processors used. But these options are more useful when a cube is being reprocessed to load new data.

After the processing runs, the results will be presented on a progress form. It is critical that the processing complete successfully. Any errors will be displayed on the form and they need to be corrected. Some error messages are confusing, but two common errors are (1) The server is not running or not accepting your connection, and (2) The cube definition has errors—often due to problems with dimensions and hierarchies. Server connection errors

By default, all projects attempt to deploy to the Analysis Services running on the local computer (localhost). In many cases, a central server will be used to run Analysis Services and the deployment location needs to be changed. For instance, students might be asked to write data to a central server, and projects set for final release need to be deployed to a central server. The process for changing the deployment server is not obvious, so Figure 3.19 shows the configuration form. Right-click the project name in the Solution Explorer and choose the Properties option to open the configuration form. Click the Deployment link. Change the name of the Server from localhost to the network name of the server running the Microsoft SQL Server Analysis Services. Your Windows account will need appropriate developer permissions on the Analysis Server. In most cases, it is simpler



**Figure 3.19**

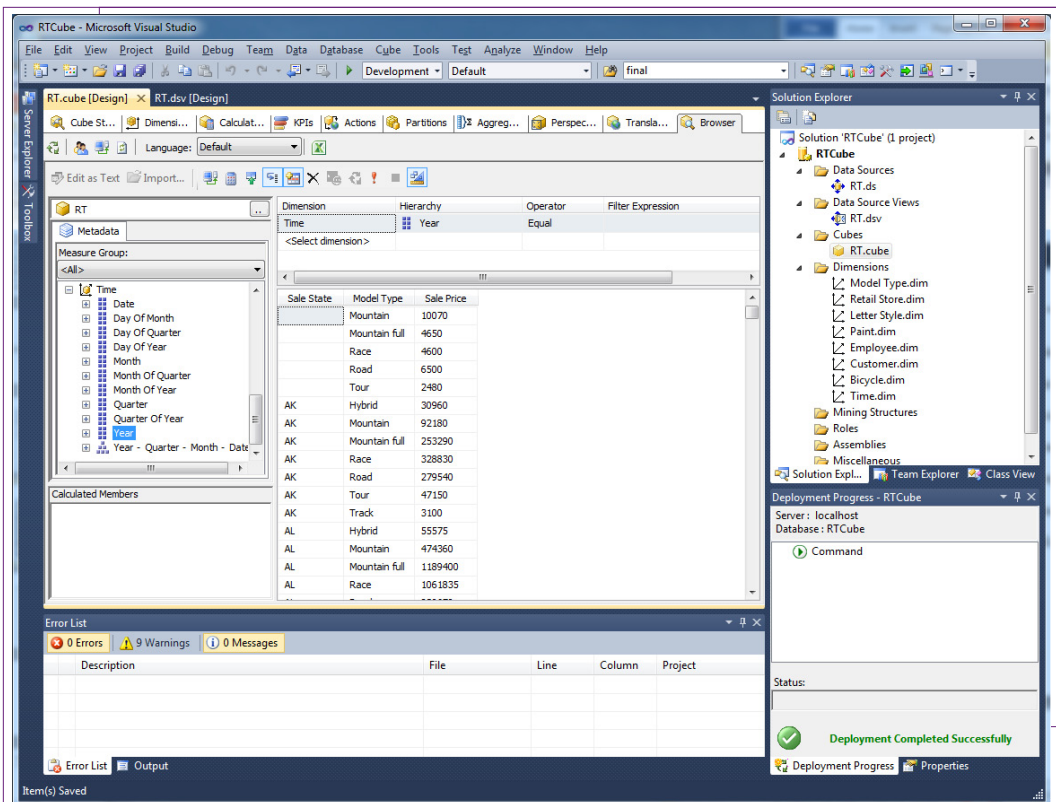
Change the deployment server. Right-click the project name in the Solution Explorer and choose Properties. Click the Deployment link. Change the name of the server from localhost to the network name of the central server.

to build and test all projects on your local computer, because security controls are easier to set.

### *Browsing the Cube*

Finally, you are ready to browse the cube and explore the data. The Visual Studio OLAP project has a cube browser, but the 2012 version is quite limited compared to the earlier 2008 version. In particular, the dimensions can be displayed only as rows, not columns. For browsing, analysts will probably want to use an Excel PivotTable instead. The PivotTable can connect to the SQL Server cube, so SQL Server performs most of the processing work. The PivotTable browser allows placing attributes on columns as well as rows so some types of data are easier to read. The Cube Browser in Visual Studio 2012 has an Excel button that will open Excel and automatically add the current cube. (Details for creating a PivotTable independently are covered in another section.) The PivotTable is initially empty with markers for where to drop fields: Row dimensions on the left, Column dimensions near the top, Totals in the middle, and Filter dimensions at the top of the cube. It is easy to drag dimensions to different locations at any time, so the initial placement is not critical. Typically, the main factor in deciding whether to put a dimension on a column or row is the size of the dimension. More rows than columns can be displayed on the table, so larger dimensions are easier to read as columns.

Figure 3.20 shows one variation of the initial cube, formed by placing the Sale Price in the middle, Model Type as columns, and Employee ID as rows. The figure shows the tree structure of the measures and dimensions in the right panels. Any of these items can be expanded and the specific dimensions dragged onto the cube. The cube computes subtotals for each dimension. In the example, a single cell shows the total sales value of a specific Model Type made by the Employee



**Figure 3.20**

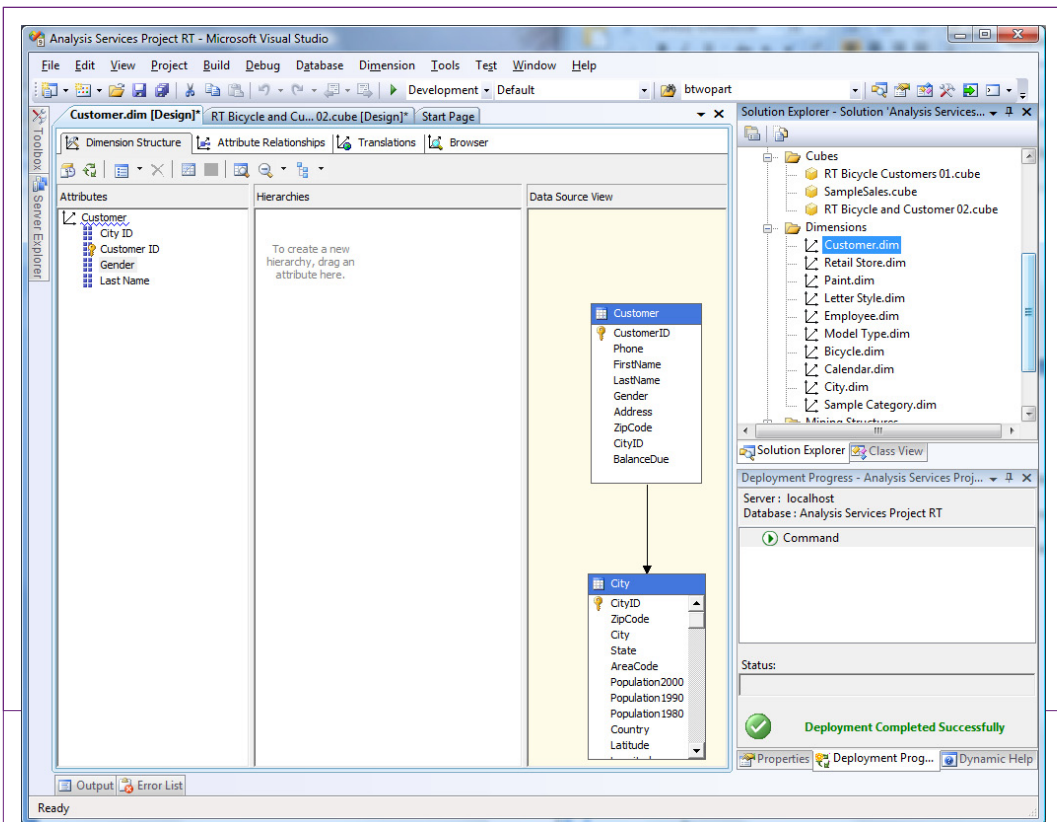
Browse the initial cube. Drag the Sale Price measure to the middle/values. Drag the Model Type dimension to the top/columns. Drag the Employee ID dimension to the left/rows.

shown on the row. Notice the Order Date is not included yet, so the values represent totals for the entire time the company has been in business. Note the difference with the SSAS cube browser, where the PivotTable can display dimensions as columns.

## Dimensions

**How are dimensions created and modified to improve browsing?** The initial cube is somewhat disappointing. Check out the dimension choices: Employee ID, City ID, Letter Style (ID), Paint (ID), Retail Store ID, and so on. Where are the actual descriptive names? What about hierarchies? Sale Date needs to be configured into at least Year-Quarter-Month-Day, and it would be nice to track customer location by state and city name. And it would be nice to format the totals to remove the decimals and add commas to make the values easier to read. All of these improvements need to be made to the cube's dimensions.

The last one, formatting is probably the easiest to fix, so consider that one first. Switch to the Cube Structure tab (the first one on the left). In the Measures panel, expand the Bicycle entry to see the three items that were selected for this cube. Select the Sale Price entry and open the Properties panel. Under the Basic section,



**Figure 3.21**

Edit the Customer dimension. Double-click the Customer dimension in the main list to open it. Drag the Gender and LastName columns from the Customer table into the Attribute panel on the left.

change the FormatString to: #,###0;-#,##0 which adds commas and removes the decimal places. Repeat the process for List Price. Bicycle Count is probably fine without formatting because the numbers are not too large. To test your changes, Process the cube and switch back to the Browser tab. Click the Reconnect icon to refresh the data.

It is more important to fix the dimensions so that managers can see the names of the dimensional values instead of just the ID numbers. No one wants to memorize lists of ID numbers. All of these require editing the underlying dimensions. First, understand that all of the dimensions are defined independently from the cube. Dimensions exist outside of the cube and then are used within any cube. This process might seem strange at first, but it makes the dimensions reusable, so they can be defined one time and applied to many cubes.

Figure 3.21 shows the basic process for editing the Customer dimension. Open the list of dimensions in the Solution Explorer and double-click the Customer dimension to edit it. Drag the Gender and Last Name columns from the Customer table into the Attribute panel on the left. After the cube is reprocessed, these col-



		Mountain	Mountain full	Race	Road	Tour	Track	Grand Total
Calendar 1994	199,006	559,020	928,984	486,774	346,201	35,029	2,555,013	
Calendar 1995	124,240	567,940	294,390	1,074,490	709,230	187,920	2,958,210	
Calendar 1996	631,760	1,469,870	1,374,850	361,020	213,360		4,050,860	
Calendar 1997	2,780	783,820	1,826,320	1,851,160	894,040		5,358,120	
Calendar 1998	110,360	1,366,180	2,712,310	683,090	1,562,180	203,270	6,637,390	
Calendar 1999		3,138,210	2,792,330	1,998,490	1,816,790	634,260	10,380,080	
Calendar 2000	94,010	1,657,070	1,611,920	671,200	600,180		4,634,380	
Calendar 2001	180,500	1,221,870	1,399,290	936,430	876,740	72,890	4,687,720	
Calendar 2002		1,523,740	2,507,560	1,581,050	1,645,770	225,980	7,484,100	
Calendar 2003		2,241,600	3,675,790	1,585,560	2,593,660	526,680	10,623,290	
Calendar 2004		896,390	3,372,690	1,858,720	2,183,070	624,800	8,935,670	
Calendar 2005		934,540	2,812,930	3,113,150	2,285,410	486,990	9,633,020	
Calendar 2006		1,205,720	2,700,080	3,235,570	2,319,750	599,460	10,960,580	
Calendar 2007		1,096,330	3,370,890	4,431,690	2,965,450	313,340	12,179,700	
Calendar 2008		846,990	2,930,500	3,931,900	2,508,860	471,980	10,640,230	
Calendar 2009		829,900	3,452,630	4,244,930	2,951,360	296,470	11,775,290	
Calendar 2010		1,018,340	4,277,740	5,171,940	3,417,090	616,580	14,501,690	
Calendar 2011		894,730	5,435,980	5,708,890	4,079,390	1,009,970	17,128,960	
Calendar 2012	628,450	1,283,150	5,327,330	5,903,620	3,748,000	632,210	17,522,760	
Quarter 1, 2012	111,940	164,510	1,063,010	1,154,660	716,840	161,460	3,372,420	
Quarter 2, 2012	178,070	367,800	1,265,120	1,430,560	1,022,140	147,300	4,410,990	
April 2012	67,780	158,450	478,280	576,620	348,270	64,520	1,693,920	
May 2012	42,450	75,840	467,680	459,010	355,720	41,310	1,442,010	
June 2012	67,840	133,510	319,160	394,930	318,150	41,470	1,275,060	
Quarter 3, 2012	116,990	350,800	1,215,100	1,165,540	702,210	102,010	3,652,650	
Quarter 4, 2012	221,450	400,040	1,784,100	2,152,860	1,306,810	221,440	6,086,700	
Calendar 2013	736,360	1,286,090	5,437,450	6,093,490	3,687,020	695,840	17,936,250	
Calendar 2014	691,900	970,200	5,792,490	6,101,470	4,560,560	588,610	18,705,230	
Grand Total	3,399,366	25,793,700	61,436,230	61,700,574	46,617,604	9,268,121	222,949	208,438,543

Figure 3.22

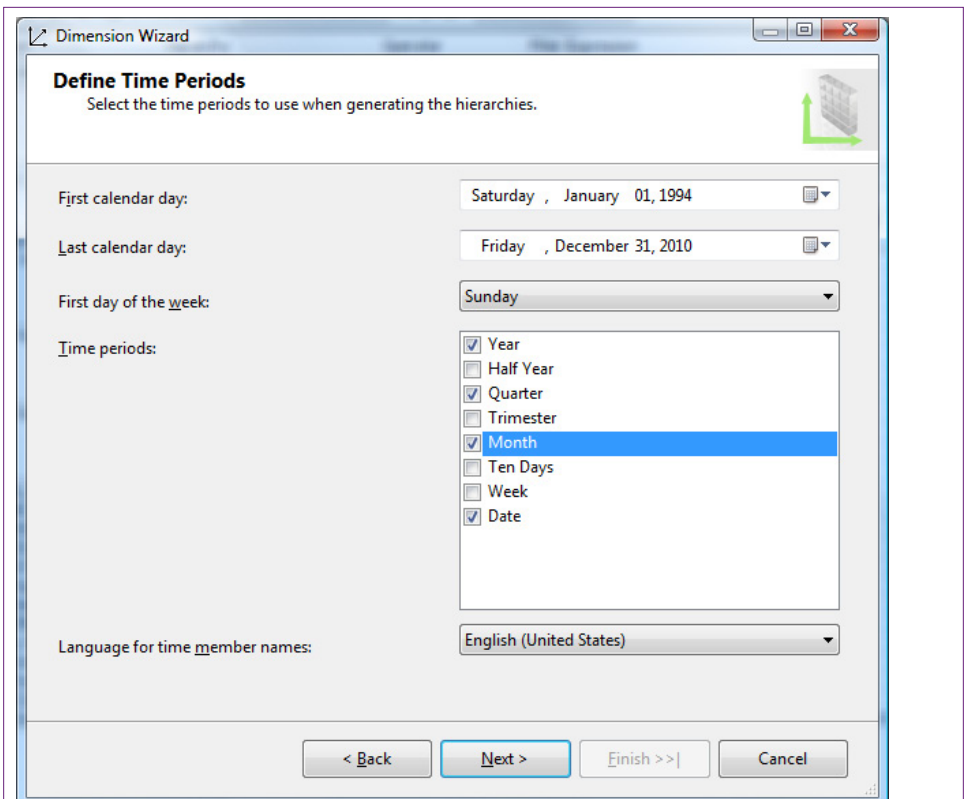
A better cube using Excel PivotTable. The formatted values are easier to read and the dimensions use names instead of ID numbers.

columns will be available to be displayed on the cube. Test it by reprocessing and reconnecting the cube. Then replace the CustomerID dimension with Gender.

Use a similar process to edit the other dimensions and add descriptive columns for: Employee, Letter Style, Paint, and Retail Store. Save the changes and close the dimensions after making the changes to reduce the clutter in the editor. Reprocess the cube after making all of the changes. One quick note about names. The attributes for names are usually listed separately (Last Name and First Name). The dimension editor will treat these columns separately. If they need to be combined into a full name (such as Smith, John), it has to be done with a Named Calculation in the data source view. Then the new Full Name column can be used as a single attribute in the dimension. Rolling Thunder has only a few employees, so they can be identified by last name. Most organizations will want to use a longer identifier, and will probably need to include the ID, phone number, or date hired to distinguish employees who have the same first name and last name.

Figure 3.22 shows the results of the changes. The cube is better because the formatted values are easier to read and the dimensions have recognizable names instead of numbers. The sample cube shows how dimensions (Paint Color and Customer Gender) can be combined to examine multiple levels of detail. This version is better, but the cube still needs hierarchies for time and location.





**Figure 3.23**

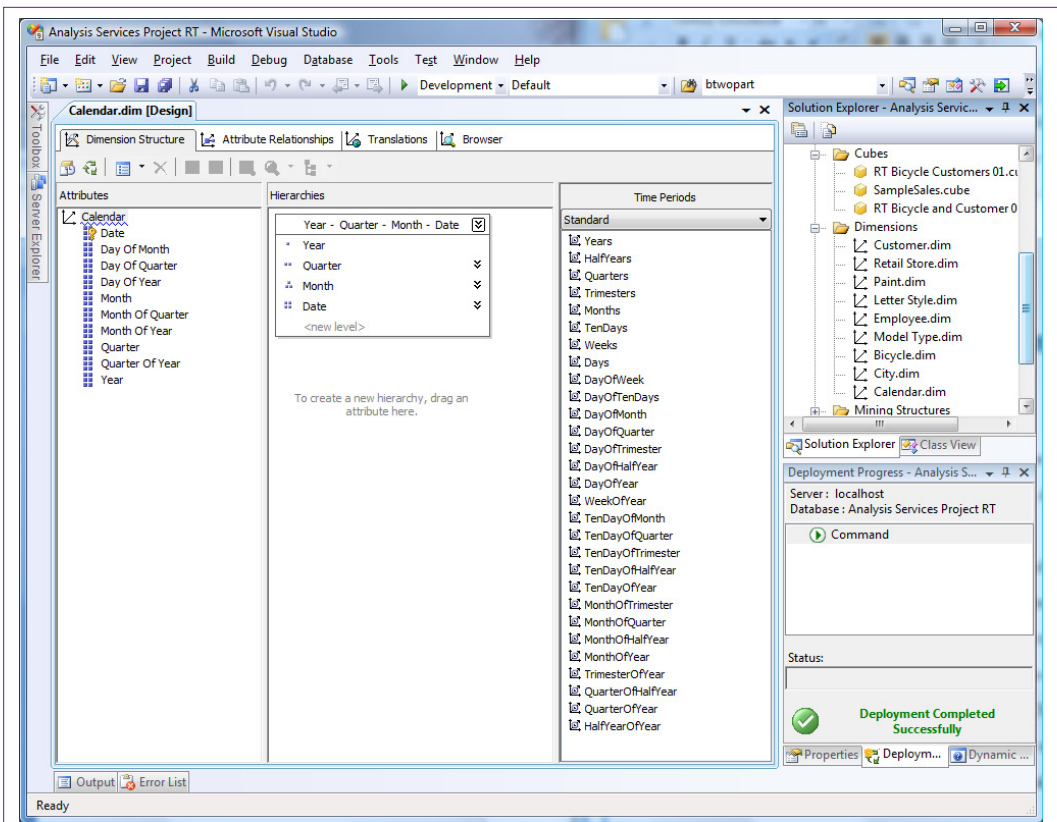
Creating a time hierarchy. Hierarchies on the analysis server require a starting and ending date, but it is okay to add extra dates. Choose the levels that will be used in the hierarchy.

## Hierarchies

Many dimension attributes need to be created as hierarchies to show different levels. Three common hierarchies in business are: (1) Dates, (2) Location, and (3) Managers or divisions. Time and Geography are so popular that Analysis Services has wizards to help create them. Internal hierarchies by managers (or perhaps products) can be created but usually require custom definitions.

Hierarchies are built within dimensions and an important aspect of dimensions is that they are created independently from the cubes. They can be applied to multiple cubes. The same concept is important to hierarchies. A hierarchy is created separately and can be applied to multiple cubes. For example, the common Year-Quarter-Month-Day hierarchy applies to many dimensions. This process is handled by defining the hierarchy and all of its data levels as a separate dimension and then building a relationship between the lowest level (Day) and the underlying dimension. With this process the date hierarchy can even be applied to multiple dates within the same table, such as Order Date and Ship Date.

Think of hierarchies as lookup tables with a specific structure. The structure and the data have to be created and stored someplace, which is independent of the cube. Then the levels of the structure can be linked to a specific data cube.



**Figure 3.24**

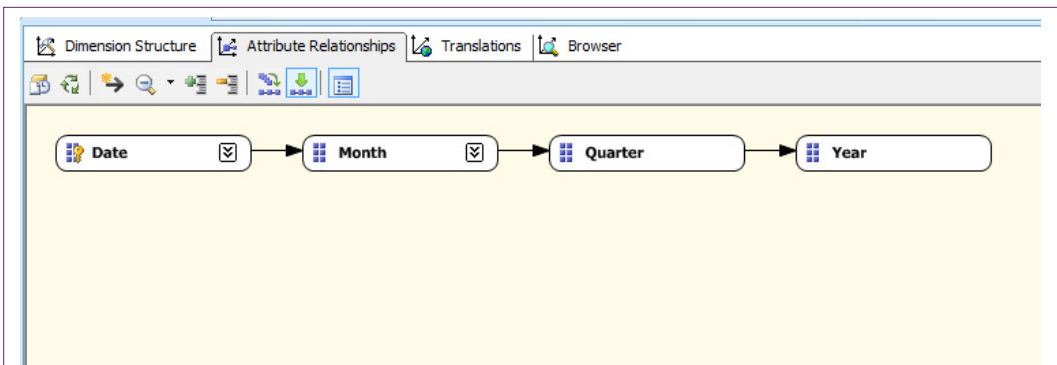
Calendar hierarchy. This dimension design form is the standard way to create and edit dimensions and hierarchies. Notice the hierarchy in the middle pane. All of the attributes in the left pane will be visible to the cube browser. Additional attributes in the table in the third pane can be dragged to the attributes pane if they are needed.

## Time Dimensions

Time dimensions are so important to most organizations that Analysis Services has several methods to create time hierarchies. Remember that hierarchies are dimensions and all data displayed in the dimensions must be stored somewhere. For time dimensions that means every year, quarter, month, and even day have to be predefined. However, because definitions of time levels are relatively standard, the dimension wizard can automatically create and populate tables with the data. The main decision you have to make is where you want to store the table: (1) Create a new table in the data source, or (2) Create a time table on the analysis server.

Because time hierarchies are the only ones that can be created on the server, that is the approach used in this chapter, simply to illustrate the process. Also, obtaining CREATE TABLE permission on the data source can be harder than it appears. To use the other option of creating a time table on the server, it is probably necessary to use the Windows login connection.

The one drawback to storing time hierarchies on the server is that the starting and ending dates have to be specified when the hierarchy is built. At a minimum,



**Figure 3.25**

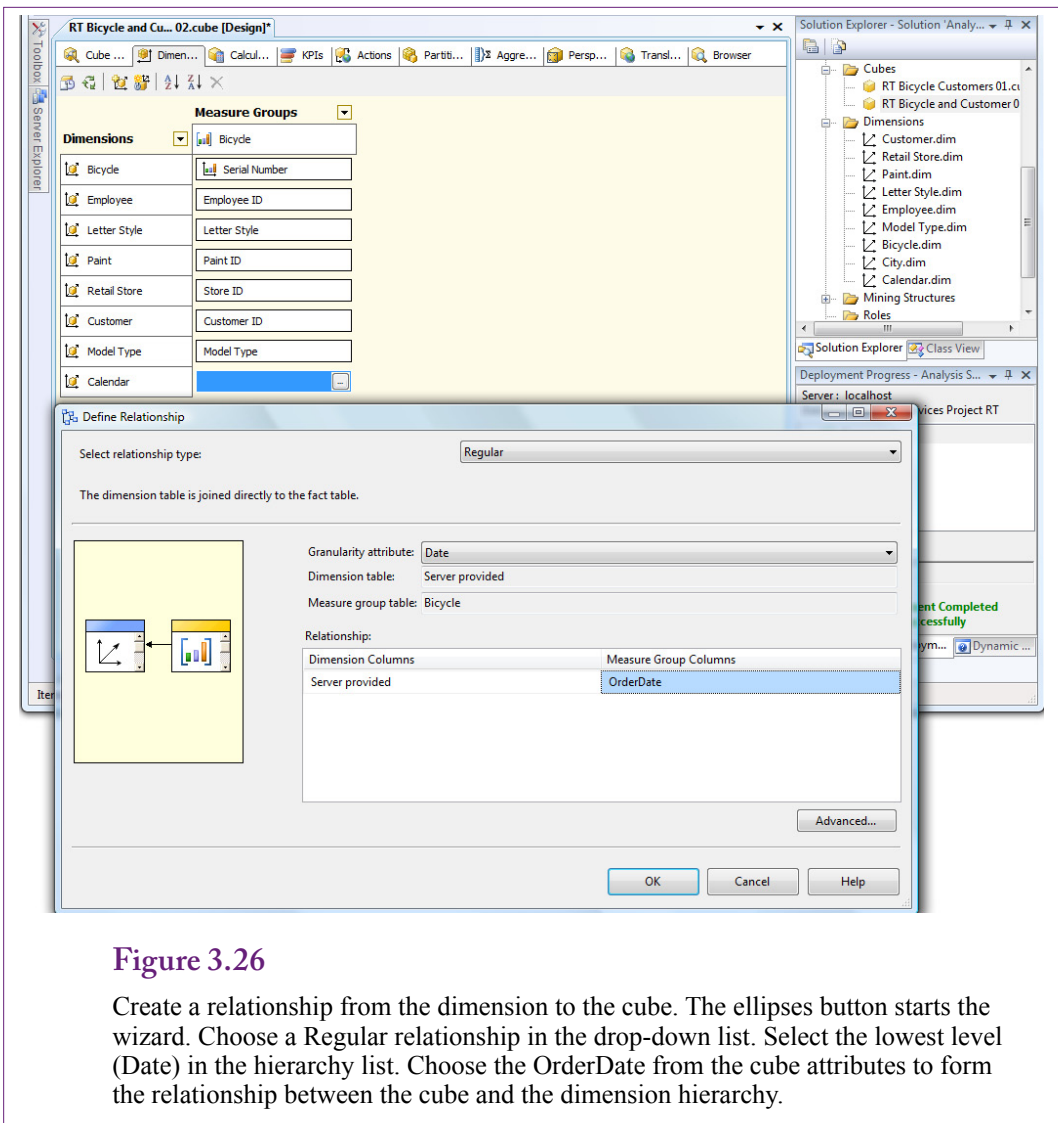
Attribute relationships. To improve performance, all levels in a hierarchy should be related in a chain from lower to highest level. The relationship property should be fixed—signaled with a solid arrowhead.

the values need to include all of the dates in the database, so you need to know those values ahead of time. Fortunately, additional dates can be stored in the hierarchy. If dates in the hierarchy do not match the data, they are not displayed in the cube browser.

To create a time hierarchy, right-click the Dimensions heading in the Solution Explorer and choose the New Dimension option. The type of dimension is the first question that needs to be answered. Choose the option to generate the time table on the server. Figure 3.23 shows the form used to specify the details of the hierarchy. Because the table will be stored on the server, the dates are independent of any existing data, so you have to set the values for the starting and ending date. The Rolling Thunder database runs from 1994-01-01 through 2008-12-31 but like a real company, new data gets added over time. So set the ending date to the end of 2010. Later, when the hierarchy is used in the cube browser, dates that do not exist in the database will be hidden.

The time wizard supports several types of calendars. Some of the basic options can be configured on the main hierarchy screen—such as the starting day of the week and the language. Others are specified on the next screen. The two main choices are a Regular calendar and a Fiscal calendar, although several others exist, including the ISO 8601 calendar that displays dates in a standardized format. Note that it is possible to generate multiple calendars for the same hierarchy, so different users in the organization can choose a calendar to meet their individual needs. For now, stick with the Regular calendar.

Figure 3.24 shows the dimension hierarchy created by the wizard. This screen and the hierarchy are worth examining because the same form is used to create other hierarchies and it is nice to have a correct example to work with. The left pane shows the attributes that will be displayed in the cube browser. The terminology needs some explanation, although it is clear once the data is displayed. Specifically, what is the difference between Quarter and Quarter of Year? In Microsoft's terminology, Quarter is a specific quarter in the time period, such as 1994 Q1 or 2007 Q3. Quarter of Year is a generic quarter without the year, such as Q1—which represents the first quarter for all years. Similar definitions apply to the Month and Day variations.



**Figure 3.26**

Create a relationship from the dimension to the cube. The ellipses button starts the wizard. Choose a Regular relationship in the drop-down list. Select the lowest level (Date) in the hierarchy list. Choose the OrderDate from the cube attributes to form the relationship between the cube and the dimension hierarchy.

The middle pane shows the actual hierarchy in top-down order. This hierarchy can be created or modified by dragging attributes from the left pane and dropping them at the appropriate level. Generally, it is best to work from the top level down to the lowest. The third pane shows all of the columns available in the data source table. Any of these can be added to the attribute list by dragging them over. However, because the wizard has finished, you will have to manually create levels with the new columns.

Technically, the hierarchy will be usable as long as it is defined correctly in the middle pane. However, to improve performance, Microsoft strongly recommends defining **attribute relationships** among the items in the hierarchy. Figure 3.25 shows the most efficient relationships are a simple chain from the lowest level to the highest. The wizard automatically created the attribute relationships, but for other problems you will have to build your own relationships. Relationships are displayed as the arrows and are created by selecting the lower-level node and

Sale Price	Mountain	Mountain full	Race	Road	Tour	Track	Grand Total	
Calendar 1994	199,006	559,020	928,984	486,774	346,201	35,029	2,555,013	
Calendar 1995	124,240	567,940	294,390	1,074,490	709,230	187,920	2,958,210	
Calendar 1996	631,760	1,469,870	1,174,850	363,020	233,900		4,050,860	
Calendar 1997	2,780	781,820	1,826,320	1,851,160	894,040		5,358,120	
Calendar 1998	110,360	1,166,180	2,712,110	683,090	1,562,180	203,270	6,117,930	
Calendar 1999	1,118,210	2,792,130	1,998,490	1,816,790	634,260		10,180,080	
Calendar 2000	94,010	1,657,070	1,611,920	671,200	600,180		4,614,180	
Calendar 2001	180,500	1,221,870	1,399,290	936,430	876,740	72,890	4,687,720	
Calendar 2002	1,523,740	2,507,560	1,581,050	1,645,770	225,080		7,484,100	
Calendar 2003	2,241,600	3,675,790	1,585,560	2,593,660	526,680		10,623,290	
Calendar 2004	896,390	3,372,690	1,858,720	2,183,670	624,800		8,935,670	
Calendar 2005	934,540	2,812,930	3,113,150	2,285,430	486,990		9,633,020	
Calendar 2006	1,205,720	2,700,080	3,235,570	2,319,750	599,660		10,060,580	
Calendar 2007	1,098,130	3,170,890	4,431,690	2,965,450	313,380		12,179,700	
Calendar 2008	846,990	2,930,500	3,931,900	2,508,860	471,080		10,690,730	
Calendar 2009	829,900	3,452,630	4,244,930	2,951,360	296,470		11,775,290	
Calendar 2010	1,018,340	4,277,740	5,171,940	3,417,090	616,580		14,501,690	
Calendar 2011	894,730	5,435,980	5,708,890	4,079,390	1,009,970		17,128,960	
Calendar 2012	628,450	1,283,150	5,327,330	5,903,620	1,748,000	632,210	17,522,760	
Quarter 1, 2012	133,940	164,510	1,065,050	1,154,660	716,840	161,460	3,172,420	
Quarter 2, 2012	178,070	367,800	1,265,120	1,430,560	1,022,840	347,300	4,410,990	
April 2012	67,780	158,450	478,280	576,620	348,270	64,520	1,693,520	
May 2012	42,450	75,840	467,680	459,010	355,720	41,310	1,442,010	
June 2012	67,840	133,510	319,160	394,930	318,150	41,470	1,275,060	
Quarter 1, 2012	116,990	350,800	1,215,100	1,165,580	702,210	102,010	3,652,650	
Quarter 4, 2012	221,450	400,040	1,784,100	2,152,860	1,306,810	221,490	6,086,700	
Calendar 2013	736,360	1,286,090	5,437,450	6,093,890	1,687,620	695,840	17,936,750	
Calendar 2014	691,900	970,200	5,792,490	6,101,470	4,546,560	588,610	18,205,230	
Grand Total	3,399,366	25,793,700	61,436,230	61,700,574	46,617,604	9,268,121	222,949	208,418,543

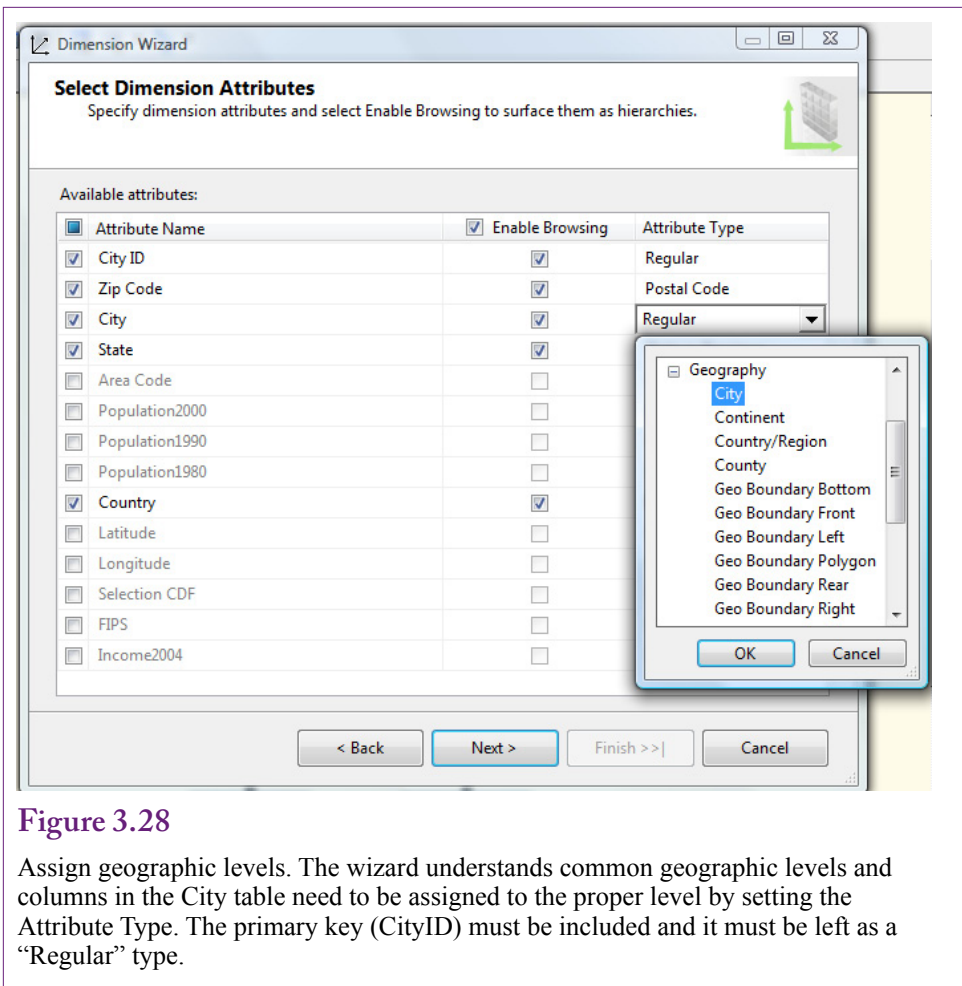
Figure 3.27

Sample cube with Calendar hierarchy. Click a + button to drill down to a detailed level. The cube is now easy to navigate for different time periods.

dragging it onto the next level. To create the first relationship in the example, you would drag the Date node and drop it onto the Month node. Notice that the arrow is solid—that signals that the relationship is fixed—the days in a specific month will never change. If necessary, this property is assigned by right-clicking the relationship arrow and setting its properties. No changes are necessary here.

Save the new Calendar dimension. Remember that it is currently a standalone dimension and is not linked to the cube. In fact, because it is a server-based dimension, it cannot be used until it is created. Save it, then right-click its name in the Dimension list of the Solution Explorer and choose the option to Process it. This method processes only the new dimension and generates the values for all of the dates. If any errors appear in processing, delete the new dimension and start over.

The generic Calendar dimension can now be assigned to the RT cube. Open the cube and switch to the Dimension (second) tab. Right-click the main window and choose Add Cube Dimension. Select the new Calendar dimension from the list of available dimensions. The calendar dates will now be available to the cube, but it is critical to link the calendar to a specific date in the data source view. As shown in Figure 3.26, select the gray box next to the new Calendar dimension and click the ellipses button. Choose Regular relationship from the drop-down list. A relationship links the hierarchy values to the data in the cube, so it is necessary to specify the attribute in the hierarchy that matches an attribute in the cube. For



**Figure 3.28**

Assign geographic levels. The wizard understands common geographic levels and columns in the City table need to be assigned to the proper level by setting the Attribute Type. The primary key (CityID) must be included and it must be left as a “Regular” type.

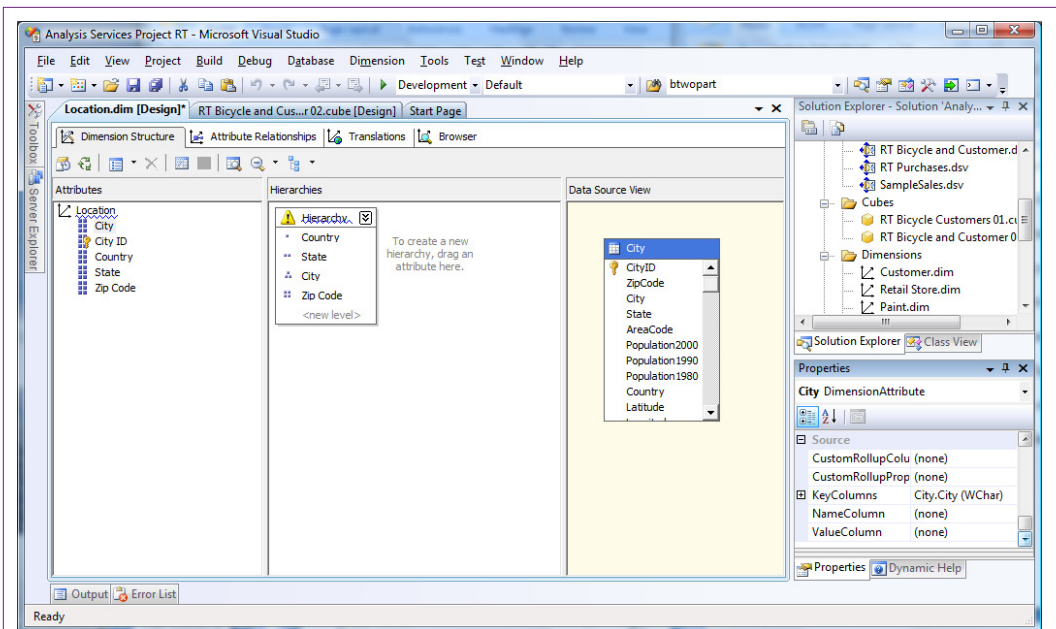
dates, choose the lowest level in the hierarchy: Date. Then choose the Order Date in the cube to build the link. It is possible to repeat the process and add a calendar for any other dates in the cube (such as Start Date and Ship Date), but keep the cube simple for now and ignore those two dates. Save everything and process the cube. Browse the cube, reconnect if necessary, remove the existing dimensions and rebuild it with Model Type in the columns and the Calendar in the rows. Figure 3.27 shows a version of the cube using the PivotTable. It is now possible to drill down and examine sales at different time periods.

### Custom Geographic Hierarchy

Geographic or location hierarchies are also common in many business problems. However, the geographic divisions are less standardized than the time divisions. The basics (Country, State, City, Postal Code) are relatively common, but the data that falls within those categories is more variable. Plus, companies often define regions or sales territories to the list and these are always defined differently. Even something as commonly used in the U.S. as “the Midwest” has many variations.

Analysis Services has a wizard to help define a geographic hierarchy, but Rolling Thunder Bicycles already has a City table that includes basic geographic data.





**Figure 3.29**

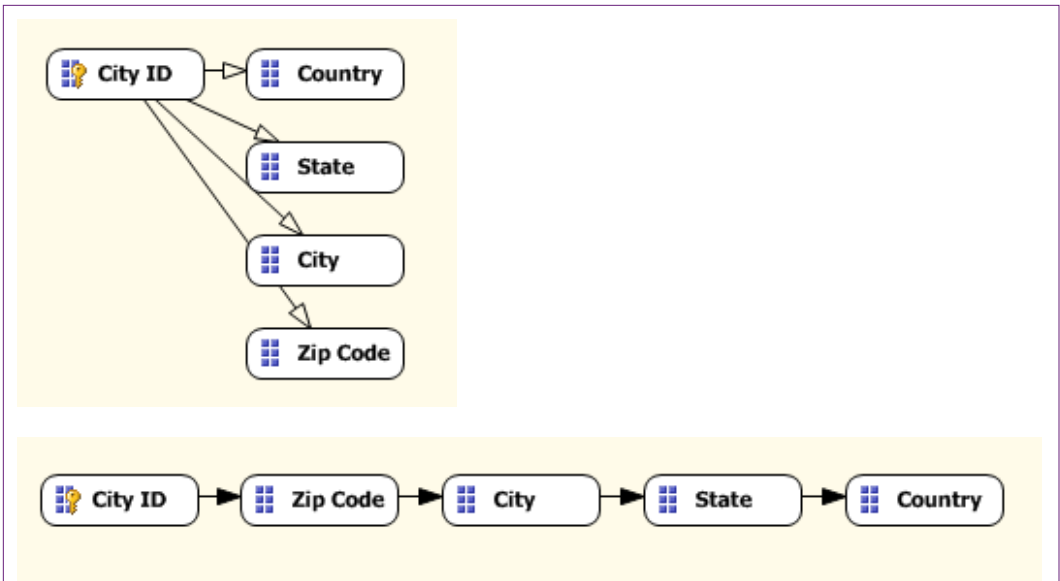
Building a hierarchy of attributes. The hierarchy begins as an empty panel. Drag the top level (Country) onto the middle panel. Drag State and drop it just below Country. Repeat the process for City and Zip Code.

Fortunately, tables of cities, states, and countries can usually be created from existing data. If a company has been in operation for years and collected customer data, the internal databases should already contain lists of addresses that can be extracted to form a City table complete with ZIP (Postal) codes. City databases can also be found online, some government agencies provide standardized databases. The Rolling Thunder Bicycle case was built with a City database that includes relatively detailed information. But, if you look closely at the data, you might spot one issue. The ZIP code aspects of the RT data are okay, but not perfectly realistic. Cities can have multiple ZIP codes, and in a few cases, a ZIP code can apply to multiple cities. This many-to-many relationship is difficult to handle in a database that relies on generating data.

Creating a geographic hierarchy is straightforward. Begin by right-clicking the Dimension entry in the Solution Explorer and choosing New Dimension. Choose the option to “Use an existing table.” On the second screen, select the data source view and choose City as the main table. The wizard automatically picks up the CityID column as the primary key.

Figure 3.28 shows the screen where columns from the City table are assigned to specific geographic levels. The common values of City, State, Country, and ZIP code are assigned to City, State or Province, Country/Region, and Postal Code. Any hierarchy must also include the primary key (CityID) and its attribute type remains as Regular. This key value will be used to link the hierarchy dimension to data stored in other tables, such as Customer, Employee, and Manufacturer. The final screen shows the attributes and has a box for changing the name. Keep the name simple, such as Location, because the hierarchy is generic and can be used for different tables.





**Figure 3.30**

Build attribute relationships to improve performance. In the initial diagram at the top, work from the bottom up. Drag Zip Code and drop it onto City, then City onto State, and State onto Country. Double-click each arrow and select the Fixed relationship until the diagram matches the one at the bottom.

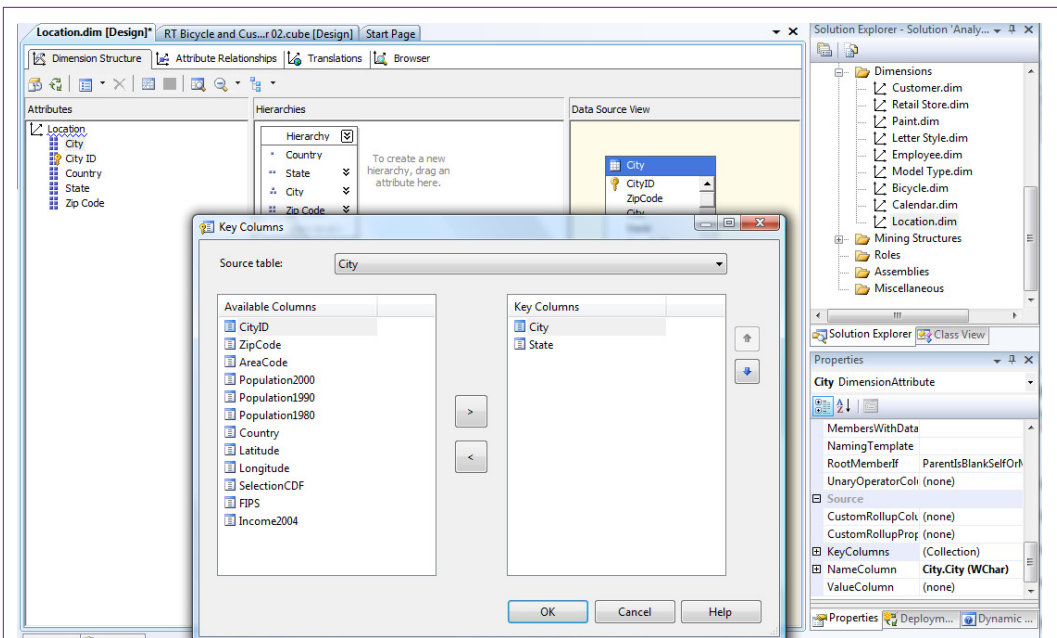
Building a dimension from an existing table does not automatically create the hierarchy. The attributes still need to be organized. Figure 3.29 shows the basic edit screen. Drag attributes from the left panel onto the middle pane (hierarchies). Begin with the top level (Country), and drag State onto a new level below Country. Repeat the process down to the ZIP Code attribute.

### Attribute Relationships

Building hierarchies from scratch requires another critical step. As shown in Figure 3.30, the attributes need to be linked together with relationships to improve performance. Switch to the Attribute Relationships tab. Initially, all of the attributes are tied to only the CityID. The hierarchy needs to be specified in terms of relationships. In terms of the arrows, it is built from the bottom up: CityID à Zip Code à City à State à Country. The relationships are created by dragging the lower-level and dropping it onto the next level (Zip Code onto City and so on).

Also, to improve performance, the relationships should be defined as rigid instead of flexible (the default). Double-click a relationship arrow and change the setting in the drop-down list. The arrowhead will be filled in to indicate the strong relationship. If you make a mistake when building relationships, a relationship can be deleted by selecting an arrow and pressing the Delete key.

Now comes the tricky part. Try to process the new dimension. An error message will be generated and the processing halted. Attribute relationships follow a very precise rule: There must be a one-to-many relationship between each level. For example, a State can have many Cities, but a City can exist in exactly one State. This relationship holds if “City” is defined as CityID, but City in the relationships

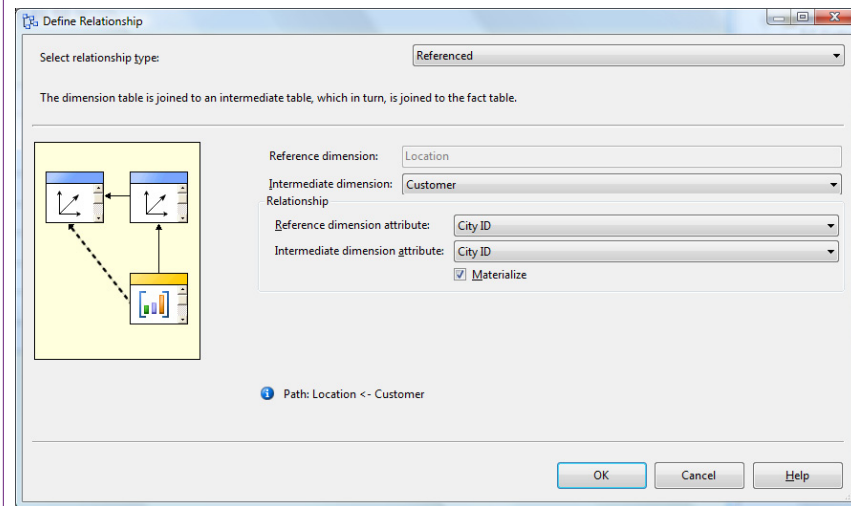


**Figure 3.31**

Adding key columns to create a one-to-many relationship. Because a City name can exist in many states, use the KeyColumns property to add State to the key so that State has a one-to-many relationship with City+State. With multiple columns in the Key, the Name column also must be set (to City).

**Figure 3.32**

Hierarchy relationship to data is referenced through Customer. Because of the snowflake design, a referenced relationship is needed to specify the intermediate Customer table.



Bicycle Count	Column List						
Row Labels	Mountain	Mountain full	Race	Road	Tour	Grand Total	
+	2			2	1	5	
- USA	7515	13065	11103	9717	2397	43797	
+	1	1	1			3	
+	AK	22	54	47	55	10	188
+	AL	136	240	148	180	36	740
+	AR	94	172	110	134	26	536
+	AZ	84	150	128	110	30	502
- CA	508	981	1216	709	180		3594
+	Acton	3	1		1		5
+	Adelanto		1	1			2
+	Agoura Hills			1	1		2
+	Alameda	6	10	5	7		28
+	Alamo				1		1
+	Albany	1	1				2
+	Alhambra	1	1	3	4		9
+	Aliso Viejo	2	3	8	1		14
+	Allendale CDP				1		1
+	Alondra Park	1				1	2
+	Alpine	1		1			2
+	Alta Sierra			1			1
+	Altadena		2	2			4
+	Alturas	1		1	1		3
+	Anaheim	2	3	9	2		16
+	Anderson	1					1

Figure 3.33

Sample cube with location hierarchy. The heading drop-down lists can be used to remove little-used data such as Italy, Unknown, and Blank for Country.

is the name of the city, and clearly the same city name can exist in many different states or countries (for instance, Paris is in Texas, Tennessee, and France).

Figure 3.31 shows how to solve this problem by adding a second column to the key for City (and any other attribute with the same problem). In the dimension editor, switch to the Dimension Structure tab. Select the City attribute in the left pane. Open the Properties window in the bottom right pane. Scroll down and select the KeyColumns entry. Click the ellipses button. If you are unsure of which values are many-to-many, try to process the dimension and watch for error messages, then modify the specified column. In this case, State needs Country added, and Zip Code needs the City.

The City table has now been converted into a dimension hierarchy. The final step is to attach that dimension to the cube. Close the dimension editor and open the cube editor. Switch to the Dimension tab, right-click the main window and choose to add a cube dimension. Select the new Location dimension which adds it to the display. Click the ellipses button in the gray box to the right of the Location dimension to open the relationship editor. The process for creating this relationship is slightly different than it was for the time dimension. Remember that the CityID is in the Customer table, not the Bicycle measure table. This link from City to Customer to Bicycle is a feature of the snowflake diagram.

As shown in Figure 3.32, change the relationship type from “No relationship” to “Referenced.” Set Customer as the intermediate dimension. The reference dimension attribute is CityID and the intermediate dimension attribute is also CityID.

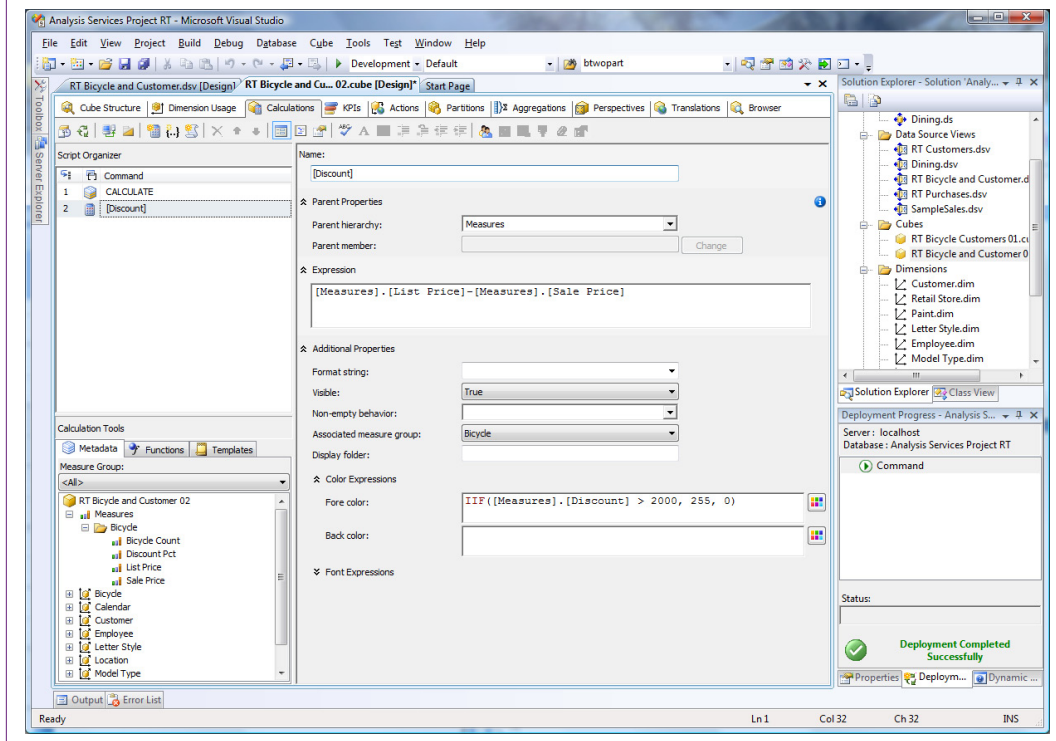
Finally, save everything, process the cube, build a PivotTable, and browse it. Use Location as the row dimension and ModelType as the column dimension. Use Bicycle Count as the value. The heading Country has a drop-down arrow that can be used to remove small entries for Unknown, Blank, and Italy. Expand a state and check the cities and ZIP codes. Almost all of the cities have only a single Zip code in the database—a consequence of the data generator relying on a single ZIP code when generating city data. Figure 3.33 shows the final cube.

## Fine Tuning the Cube

**How can the cube provide more information?** Think of an OLAP cube as an easy way for managers to explore the data by examining various sub-totals or groupings of the data. Of course, the filters are also useful for showing slices of the cube to examine specific details or categories. But, the cube browser is restricted to the data measures and dimensions that have been predefined. Consequently, as the designer, you need to think about all of the possible things that

**Figure 3.34**

Calculated measure. Right-click the Script Organizer window and create a new calculation. Enter a name (Discount) and assign it to the Measure hierarchy. Drag attributes from the list on the lower left and drop them in the Expression window. Add the minus sign. Assign it to the Bicycle group. Use the IIF function to set the color to Red for discount totals over 2000.



	A	B	C	D	E	F	G
1	Discount	Column Labels					
2	Row Labels	Mountain	Mountain full	Race	Road	Tour	Grand Total
3	Calendar 1994	3638.39		3361.51	2993.84	2108.4	12102.14
4	Calendar 1995	1349.04		523.2	1933.02	1577.82	5383.08
5	Calendar 1996	4167.95		2886.1	743.04	471	8268.09
6	Calendar 1997	1576.8	3561.99	3380.12	1625.12		10144.03
7	Calendar 1998	2948.79	6138.03	1416.24	3111.83	450.79	14065.68
8	Calendar 1999	4551.26	4015.33	2779.61	1898.54	741.26	13986
9	Calendar 2000	2787.18	2757.42	1107.14	1092.88		7744.62
10	Calendar 2001	1860.35	2136.92	1668.99	1411.26	143.19	7220.71
11	Calendar 2002	2463.05	3937.2	2591.46	2659.83	472.82	12124.36
12	Calendar 2003	2988.88	5053.02	1891.96	3450.06	941.28	14325.2
13	Calendar 2004	1486.24	4509.17	2131.16	2750.25	938.08	11814.9
14	Calendar 2005	1479.42	3837.19	3460.91	3143.24	778.35	12699.11
15	Calendar 2006	1432.01	2941.79	2932.47	2472.16	813.92	10592.35
16	Calendar 2007	1319.33	3674.88	3721.76	2920.11	309.79	11945.87
17	Calendar 2008	955.8	3109.3	3551.64	2487.34	447.79	10551.87
18	Calendar 2009	758.84	2773.95	2589.18	2315.95	210.88	8648.8
19	Calendar 2010	864.03	3278.91	3076.01	2430.78	454.31	10104.04
20	Calendar 2011	727.57	3652.42	3405.22	2821.64	630.33	11237.18
21	Calendar 2012	773.07	3236.72	3516.8	2817.57	454.54	10798.7
22	Quarter 1, 2012	112.12	633.33	680.17	567.28	99.86	2092.76
23	Quarter 2, 2012	217.28	781.19	811.33	750.22	117.52	2677.54
24	Quarter 3, 2012	214.3	667.67	705.53	485.53	70.52	2143.55
25	Quarter 4, 2012	229.37	1154.53	1319.77	1014.54	166.64	3884.85
26	Calendar 2013	698.57	3091.21	4027.3	2757.39	561.43	11135.9

Figure 3.35

Cube results for Discount. Notice that the color conditions and the calculations apply to the total values as they are displayed. With VS 2012, these colors transfer to Excel but do not display in the VS cube browser.

managers will want to see. Note that it is important for the designers to actually talk with the managers while defining cubes.

But, what happens if managers need values that are not in the underlying database? Specifically, managers and analysts often need calculated values such as Profit = Revenue – Cost or Discount = ListPrice – SalePrice. **Calculations** and queries

Also, what happens when a cube gets too big with dozens of measures and attributes? **Perspectives** are used to create multiple views of a cube so that each perspective shows a smaller set of the dimensions and attributes—specifically for one group or type of problem.

What about companies that operate in multiple countries and languages? Internationalization is important, and much of the work has to be carried through the entire database. However, cubes have some functions that provide support for translations. Note that some functions described in this section are supported only on the Enterprise and Developer versions of SQL Server. The Standard version provides limited support for some types of calculations and most optimizations. If some examples do not work on your configuration, verify the version you are using.

## Calculations and Queries

Many problems require computing new values based on existing data. The cube designer provides the ability to specify calculations. Because the cube is based on

	X	Y	X*Y
	100	2	200
	5	10	50
Sum	105	12	1,260 or 250

**Figure 3.36**

The order of operations. It makes a big difference if numbers are added first and then multiplied or multiplied first and then summed.

a data source view which behaves as a query, it is also possible to compute new columns using SQL statements as named calculations within the data source view. Always remember that the cube is designed to display aggregate values—usually totals. Combining calculations with totals can cause serious errors. It is critical to understand how calculations work—and the lack of documentation makes it a challenge.

### *Cube Calculations*

Consider an easy example first, where managers want to explore the value of the discount given on each bicycle.  $\text{Discount} = \text{ListPrice} - \text{SalePrice}$ . The reason this example is easy is because it is not the percentage discount. The calculation involves simple subtraction. As shown in Figure 3.34, select the Calculations tab and right-click the Script Organizer window to add a New Calculated Member. Name it Discount and assign it to the Measures hierarchy. Drag the ListPrice and SalePrice columns from the list in the lower-left window into the Expression window and add the minus sign. Assign the new measure to the Bicycle group. Just for fun, open the Color Expressions properties. For the foreground color, enter the **immediate if function (IIF)** to set the color to red if the total discount exceeds 2000:

```
IIF([Measures].[Discount] > 2000, 255, 0)
```

The IIF function takes three parameters and is similar to the IF function in Excel. The first is a conditional test. The second is the value returned if the condition is true. The third is the value returned if the condition is false. The color values 255 and 0 represent Red and Black. Use the color picker next to the box to obtain more complex color numbers.

Save everything and process the cube. Transfer the cube to Excel and create a new display cube with the Discount as the value totals, ModelType as the column dimension, and Calendar as the row dimensions. Figure 3.35 shows the results. Notice that the color conditions apply to the totals as they are shown. This result makes sense, but it does complicate the choice of the conditional value (2000). As the user drills down by quarter and month, the values will likely fall below the critical number. Conditional color adds some emphasis to the cube display, but it will only work at a certain level.

A more important problem arises but is invisible with this example. It turns out that the defined calculation (Discount) is also applied to the totals or visible data. The Discount value is computed at each displayed level as:  $\text{Sum}(\text{ListPrice}) - \text{Sum}(\text{SalePrice})$  for that level. Because the calculation involves only subtraction (or addition), this method of calculation is fine. From basic arithmetic:



$$\text{Sum}(\text{ListPrice} - \text{SalePrice}) = \text{Sum}(\text{ListPrice}) - \text{Sum}(\text{SalePrice})$$

But this method of calculation can be seriously wrong if the calculation uses multiplication or division. Figure 3.36 shows a simple example with two columns and two rows of data. Because of the **order of operations**, it makes a big difference if the numbers are added first and then multiplied or multiplied first and then added. In the example, the two calculations are 1,260 versus 250. It is absolutely critical to remember that calculations defined within the cube are based on first computing the total and then performing the calculation. In the example, if the cube calculation is  $X*Y$ , the result would be 1,260 which is  $\text{Sum}(X)*\text{Sum}(Y)$ . Essentially, the calculations are performed on the values as they are shown in the cube—not on the detailed rows. Once in a while, this order of calculation makes sense—for example, to compare one subtotal to another. Much of the time, this level of calculation leads to huge errors in interpretation. So, remember a simple rule: Only use addition and subtraction in calculations defined on the cube. Never use multiplication or division. The discount calculation shown in this section is fine because it used only subtraction.

But some problems require the use of multiplication and division! The answer is to define those at the row level of the data—in the data source view or at the database level in a query. For example, a cube-level calculation computes the sum first and then divides, while a query-level calculation computes the division first and then sums (or averages)

$$\frac{\sum \text{Sale Price}}{\sum \text{List Price}} \neq \sum \frac{\text{Sale Price}}{\text{List Price}}$$

**Figure 3.37**

Standard cube using the Discount Percent. But the totals are greater than one. By default, the cube totals all data at lower levels. Totals do not make sense for percentages.

			Model Type ▾					
			Mountain	Mountain full	Race	Road	Tour	Grand Total
Year ▾	Quarter	Month	Discount Pc	Discount Pc	Discount Pc	Discount Pc	Discount Pc	Discount Pc
Calendar 2000			0.97759995	0.98159995	0.4197	0.42900000		2.8079
Calendar 2001			0.6066	0.69659995	0.5936	0.5054	0.0564	2.4586
Calendar 2002			0.86849995	1.3564	0.8655	0.9756	0.217	4.28299995
Calendar 2003			0.84939995	1.46229995	0.5007	1.0559	0.3738	4.24209995
Calendar 2004			0.5544	1.3801	0.523	0.74439995	0.3069	3.5088
Calendar 2005			0.5292	1.1938	0.85019995	0.88199995	0.2532	3.70839995
Calendar 2006			0.3842	0.711	0.568	0.5522	0.2176	2.433
Calendar 2007			0.348	0.88139995	0.69139995	0.59449995	0.0813	2.5966
Calendar 2008	Quarter 1, 2008		0.0682	0.1872	0.1968	0.1234	0.0236	0.5992
	Quarter 2, 2008		0.0573	0.1934	0.1885	0.1568	0.0204	0.6164
	Quarter 3, 2008		0.053	0.1779	0.1556	0.105	0.0261	0.5176
	Quarter 4, 2008		0.0627	0.1833	0.161	0.1562	0.0332	0.5964
	Total		0.2412	0.74179995	0.7019	0.54140000	0.1033	2.3296
Grand Total			13.9648	14.9075	11.2047	11.1553	4.0707	55.3029995



9	⊗ Calendar 2000	0.35%	0.36%	0.25%	0.25%		0.78%
10	⊗ Calendar 2001	0.26%	0.29%	0.28%	0.26%	0.26%	0.70%
11	⊗ Calendar 2002	0.36%	0.46%	0.36%	0.37%	0.28%	1.19%
12	⊗ Calendar 2003	0.32%	0.47%	0.23%	0.41%	0.25%	1.18%
13	⊗ Calendar 2004	0.28%	0.45%	0.22%	0.27%	0.23%	0.96%
14	⊗ Calendar 2005	0.26%	0.38%	0.28%	0.30%	0.21%	1.02%
15	⊗ Calendar 2006	0.18%	0.24%	0.19%	0.21%	0.19%	0.67%
16	⊗ Calendar 2007	0.18%	0.28%	0.23%	0.21%	0.12%	0.72%
17	⊗ Calendar 2008	0.15%	0.25%	0.22%	0.20%	0.11%	0.64%
18	⊗ Calendar 2009	0.12%	0.16%	0.13%	0.15%	0.08%	0.38%
19	⊗ Calendar 2010	0.11%	0.17%	0.14%	0.14%	0.09%	0.43%
20	⊗ Calendar 2011	0.11%	0.17%	0.14%	0.15%	0.08%	0.44%
21	⊗ Calendar 2012	0.09%	0.14%	0.16%	0.16%	0.09%	0.43%
22	⊗ Quarter 1, 2012	0.10%	0.11%	0.14%	0.15%	0.07%	0.34%
23	⊗ Quarter 2, 2012	0.08%	0.14%	0.14%	0.16%	0.09%	0.43%
24	⊗ Quarter 3, 2012	0.09%	0.13%	0.14%	0.12%	0.08%	0.35%
25	⊗ Quarter 4, 2012	0.09%	0.19%	0.22%	0.21%	0.10%	0.62%
26	⊗ Calendar 2013	0.07%	0.13%	0.18%	0.16%	0.10%	0.44%

**Figure 3.38**

Discount Percent as an average. The discount percentage is computed at row-level in the data source view and the cube averages the values from the lower levels.

### Query-Level Calculations

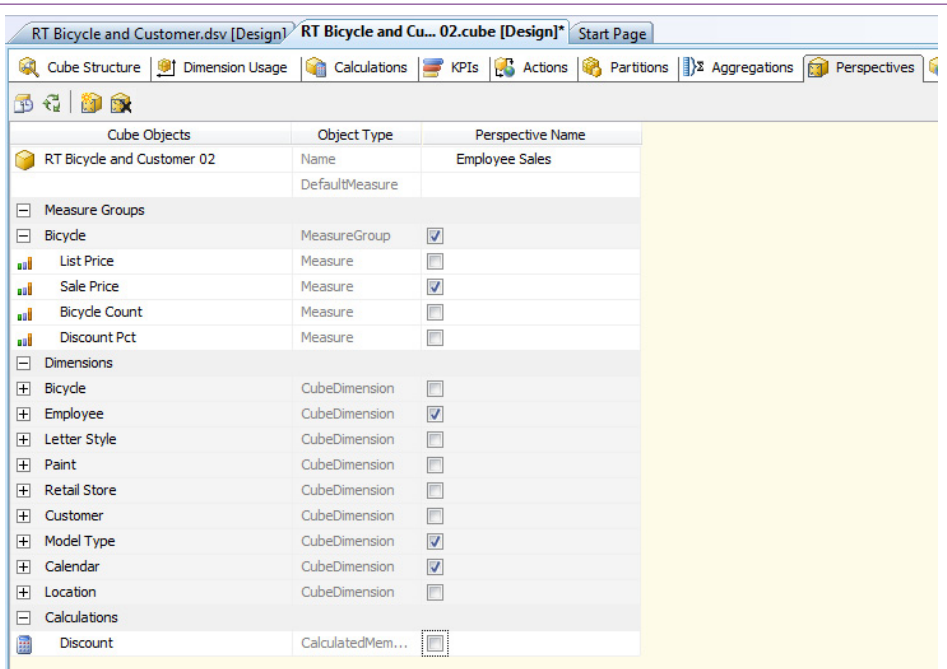
Most business managers looking at discounts prefer to examine percentages. But the calculation for discount percent uses division:  $1 - \text{SalePrice} / \text{ListPrice}$ . Which is the reason this computation was introduced in the earlier section about data source views. You should have already defined this Discount Percent as a named calculation within the Bicycle table. If not, open the data source view and add it to that table. Calculations performed within a query are handled at the row level, performed for each row of data before any aggregations take place.

To see the effect, return to the cube browser. Drag the Discount calculation out of the cube and replace it with the Discount Percent. Figure 3.37 shows the result. Look at the grand totals and notice the problem. How can percentages be greater than one? The answer is that by default the cube browser computes the sum of all lower-level values.

The sum of percentages does not make much sense to a business analyst. It would be better to use averages. Fortunately, the cube browser knows how to compute averages. Unfortunately, because it is an aggregation, it means the cube has to be reprocessed. Switch to the Cube Structure tab and select the Discount Pct measure. Open the Properties window and change the Aggregate Function from Sum to AverageOfChildren. Also, set the FormatString to Percent to make the results easier to read.

Figure 3.38 shows the final cube. It might be nice to add conditional color to highlight larger discounts. But, conditional color is available only for cube calculations. It can be added by calculating a new cube measure that is exactly equal to the Discount Pct column, and apply color to the new value.

Calculations are useful, but it is crucial that you understand the difference between computing values row-by-row in a query and performing cube calculations



**Figure 3.39**

Creating a perspective. Right-click the main screen to create a new perspective. Name it (Employee Sales) and check only the values that should be available.

at the level of the totals. And it is important to document your calculations. Managers and analysts using the cube need to know which method was used to perform the calculations.

## Perspectives

Data and cubes can quickly become huge when multiple measures and many dimensions are available. Putting every possible attribute into one cube is going to make the cube hard to understand. A common solution to handling complex problems is to break them into smaller pieces. Analysis services provides several mechanisms for segmenting a problem. One is to create separate data source views for each specific problem. Related to that approach, separate OLAP cubes can be defined. Each cube can be built for a specific problem or group of managers. It is straightforward to assign security permissions to each cube, and when data changes, each cube can be rebuilt individually. Hence, cubes can be built for specific tasks to be managed by different groups. On the flip side, if the cubes predominantly use the same data, rebuilding each cube wastes processor time by rebuilding the same data multiple times.

Consider the case where basic data needs to be shared among several different groups, but some dimensions should be seen only by a few users. For instance, perhaps HRM and some top managers should be the only ones to see employee sales data. The answer is to define perspectives on the cube. A perspective is a view of the cube that contains a subset of the available measures and attributes. Figure 3.39 shows the basic steps to create a perspective. Open the Perspectives

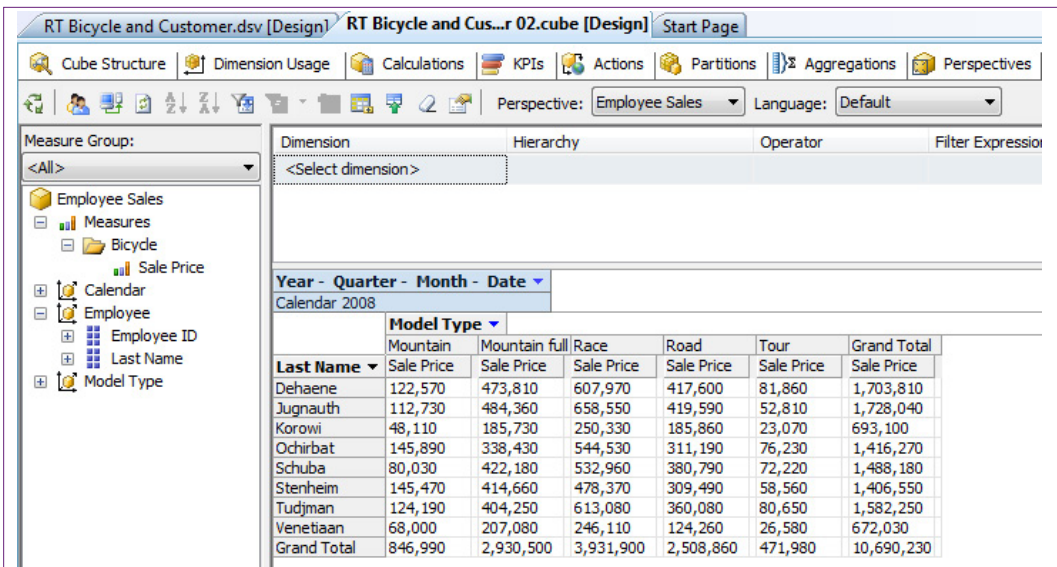


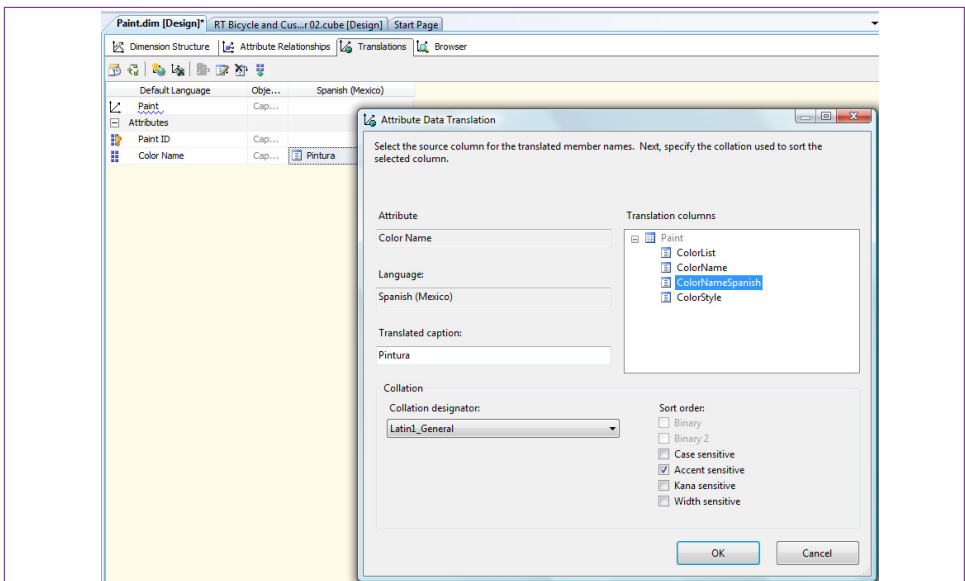
Figure 3.40

A perspective reduces the measures and dimensions visible to the user. Use the drop-down box to select the desired perspective.

Figure 3.41

Translating dimensions. Attributes stored in tables is translated and stored in new columns in the same table.

PaintID	ColorName	SpanishName
1	Neon Blue	Neón Azul
2	Arctic White	Blanco Ártico
3	Sea Green Fade	Verde Mar
4	Black Speckle	Salpicaduras Negras
5	Candy Stripe	Caramelo Banda
6	Fire and Smoke	Fuego y el Humo
7	Mountain Green	Montañas Verdes
8	Purple Accent	Morado Acento
9	Hazard Flame	Llama Peligro
10	Morning Sun	Sol por la Mañana
11	Grey Granite	Granito Gris
12	Copper Haze	Neblina Cobre
13	Sky Fire	Cielo de Bomberos
14	Wine Country	Vino País
15	Black Hole	Agujero Negro



**Figure 3.42**

Assigning dimension translations. Open the dimension and click the translation tab. Right-click the screen to add a new translation and pick the language. Click the ellipses button in the gray box next to the Color Name attribute. Select the matching column name. Translate the title.

tab and right-click the main screen area to add a new perspective. Enter a name for the perspective that all users will recognize. Set the check boxes to select only the measures and attributes that should be included in the new perspective. By default, all available items are checked, so it becomes a process of removing the items not needed.

Save everything, process the cube, switch to the browser and reconnect. Figure 3.40 shows the new cube. The drop-down list in the center of the toolbar is used to choose a perspective. When Employee Sales is chosen, note the limited number of options available for use in the cube.

## Internationalization and Translations

Many organizations today extend across national boundaries and need to provide data in multiple languages. Analysis Services has several tools to facilitate handling multiple languages. Keep in mind there is no magic bullet—someone still has to translate all of the terms. The purpose of the tools is to display the data and metadata in a specified language.

Two types of information need to be translated and displayed correctly: (1) Dimension data stored in the tables and (2) Metadata such as the titles of dimensions. The difference might seem subtle, but the two translations are handled with different techniques.

Dimension data requiring translation would include the different model types, color names, and so on. The original data exist in rows in the table (Paint), so the translation is handled by adding new columns to that table—one column for each language. Figure 3.41 shows sample data that might be used to translate the color names into Spanish. A marketing manager would likely want to improve the

names to be more appealing in the local language. The process is to add a column to the Paint table with a column name that indicates the language and then enter the translations row-by-row. The same process would be followed for the other dimensional attributes.

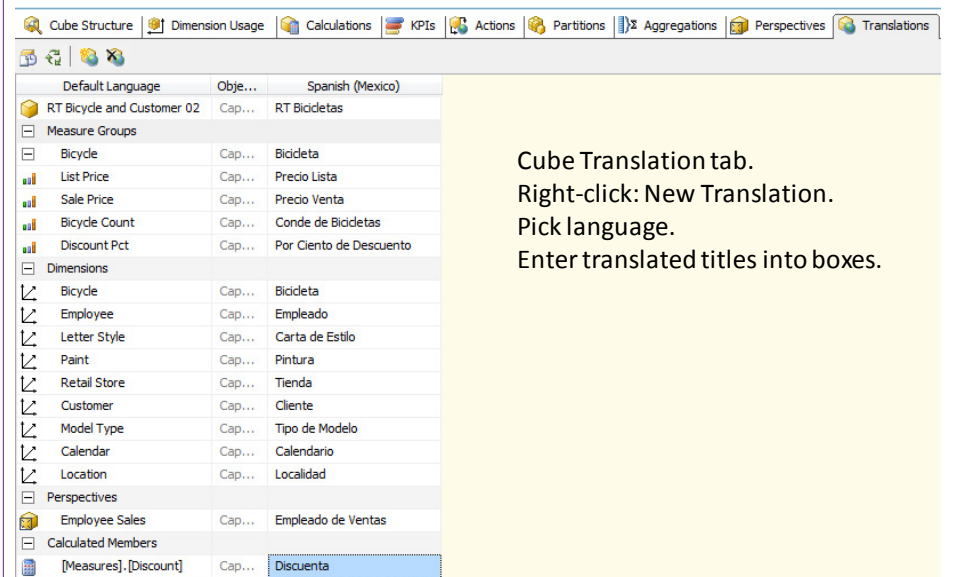
Once the translations exist, they can be assigned within the dimension. Open the dimension from the Solution Explorer and select the Translations tab. Right-click the main screen and choose the option to create a new translation. Choose the desired language. Click the ellipses button in the gray box next to the Color Name attribute to open the edit window. Select the new column name for the correct language. Enter the translation for the title. Note that all of the dimensions will need to be translated. It will take quite a bit of time just to enter the data and assign all of the dimension attributes—not including the time it takes to translate the actual words. When planning a schedule, remember to multiply the time by the number of languages needed.

A second step remains. The metadata for the cube needs to be translated—all of the titles that were entered must also be translated. Fortunately, the cube keeps track of them and organizes them in one location on the Translation tab. As shown in Figure 3.43, the editing process is simpler for metadata. Switch to the Translation tab in the cube editor and right-click to add a new language. Choose the language to create a column next to the original data. Enter the translated titles.

Figure 3.44 shows the cube in the new language. Remember to process the cube first then open the browser. Select the new language from the drop-down list. Notice the dimension names are mostly in Spanish in the left window. The paint names are correct, but the model types have not yet been translated. If a translation does not exist, the cube falls back to the default (initial) values. The point of

**Figure 3.43**

Translating cube metadata. Select the Translation tab on the cube editor. Right-click to add a new language. Enter the translated names for all of the titles.





The screenshot shows the 'RT Bicycle and Customer 02.cube' design interface. The top menu includes 'Cube Structure', 'Dimension Usage', 'Calculations', 'KPIs', 'Actions', 'Partitions', 'Aggregations', and 'Perspectives'. The 'Perspective' is set to 'RT Bicletas' and the 'Language' is 'Spanish (Mexico)'. The 'Measure Group' is 'RT Bicletas'. The dimension hierarchy is 'Year - Quarter - Month - Date' with 'Calendar 2008' selected. The data table shows sales for 'Pintura' models across different regions.

Pintura	Model Type					
	Mountain	Mountain full	Race	Road	Tour	Grand Total
Blanco Ártico	82,300	286,890	392,250	194,080	31,050	986,570
Agujero Negro	30,880	137,370	191,320	107,800	12,560	479,930
Neblina Cobre	73,590	226,560	328,790	234,480	46,900	910,320
Fuego y el Humo	87,040	230,670	319,030	204,960	23,000	864,700
Granito Gris	92,250	294,260	317,860	185,760	43,830	933,960
Llama Peligro	61,760	251,930	275,110	251,020	44,010	883,830
Sol por la Mañana	66,750	266,990	301,150	169,280	21,850	826,020
Montañas Verdes	87,160	203,490	332,720	150,970	57,680	832,020
Neón Azul	24,310	116,570	157,690	100,900	14,150	413,620
Morado Acento	54,800	199,810	387,590	243,450	39,920	925,570
Verde Mar	49,740	215,750	342,750	170,640	62,000	840,880
Cielo de Bomberos	51,710	215,530	293,230	234,820	12,780	808,070
Vino País	84,700	284,680	292,410	260,700	62,250	984,740
Grand Total	846,990	2,930,500	3,931,900	2,508,860	471,980	10,690,230

Figure 3.44

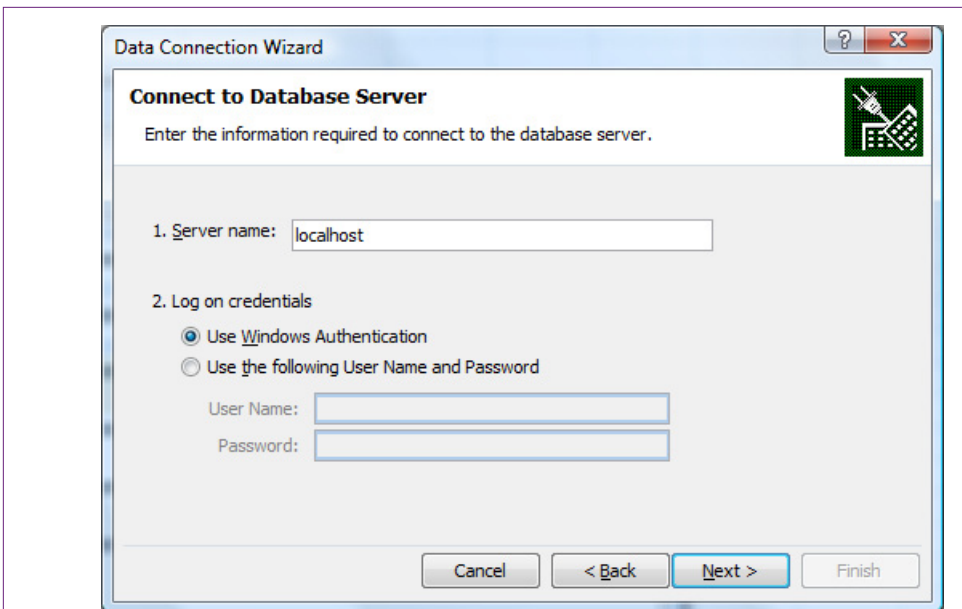
The cube partially in Spanish. Browsing the cube in a different language is accomplished by electing the desired language from the drop-down list.

the demonstration is that it takes quite a bit of time to identify all of the data and metadata and to enter the translated values. However, once the translation and data-entry is done, the users simply select a language and the cube picks up the appropriate values automatically.

Of course, countries generally have different currencies. The cube browser has a limited ability to convert monetary values to different currencies. Currency conversions are handled by creating a giant table of exchange rates. The Microsoft Adventure Works demonstration database has a sample table. It lists daily exchange rates for several currencies. Tables of historical exchange rates can be found online (for example, [www.oanda.com](http://www.oanda.com)). Setting up the table correctly and obtaining the data is the most complicated step. It is probably best to import the table from the Adventure Works database and create an ExchangeRate measure by copying Microsoft's example. Once the exchange rates are specified, a BI wizard helps apply them to the cube. Right-click the cube name in the Solution Explorer and choose the option to Add Business Intelligence. One of the options is "Define currency conversion." This wizard steps through the process of defining the exchange rate conversions. However, the details are not covered in this book. See (Harinath et al. 2009) for the specific steps using the Adventure Works example.

### Performance: Partitions and Aggregations

Even with advanced hardware, huge problems can lead to performance problems. Analysis Services provides several options that can be added when performance begins to decline. One of the important tools is partitions. A partition is a physical



**Figure 3.45**

Connecting to an OLAP data cube. The first time a PivotTable is created, follow the wizard steps to connect to an external data source and create a new connection. Enter the server name and login information on this screen.

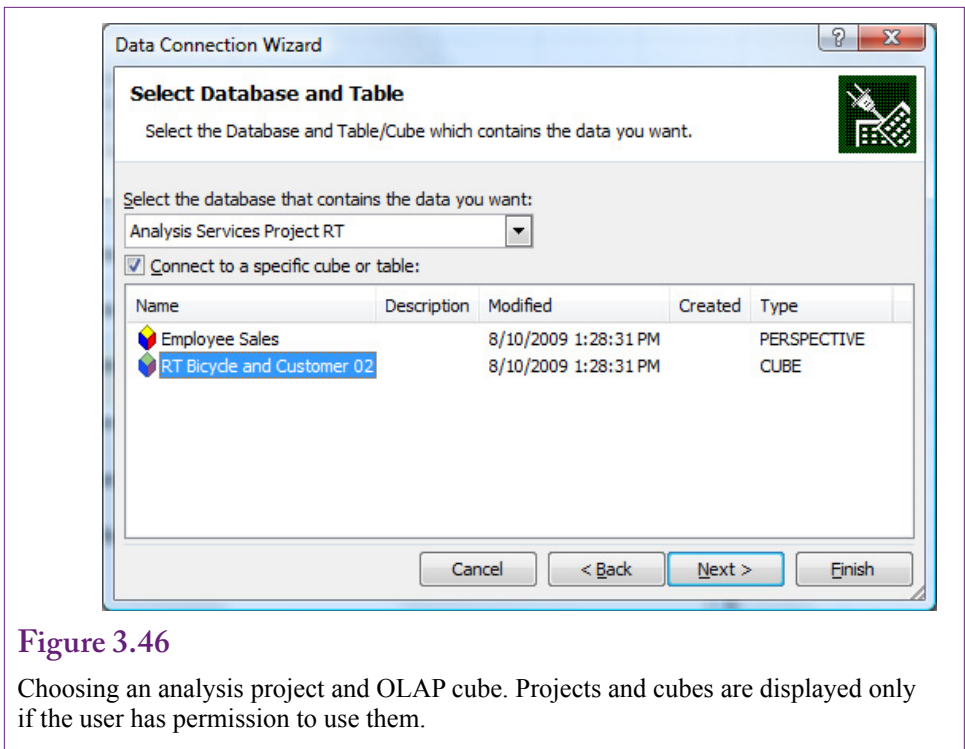
separation of data in the cube. By default, all data is stored in a single partition. Microsoft recommends that a partition should contain no more than 20 million rows. For problems with substantially more rows, new partitions should be added that split the data into these smaller sets. Each partition can be stored in different locations, can be assigned different storage structures (MOLAP, ROLAP, or hybrid), and can be assigned different security permissions.

The partitions are useful because they can be indexed and searched separately. They can also be processed on remote servers—spreading the processing load across multiple servers reduces the overall load and process time. Also, if a query retrieval requires only part of the data, the system automatically pulls data only from the needed partitions.

**Aggregations** are the totals and averages computed on the measures to form the subtotals of the cube. Complex cubes can end up with dozens or hundreds of aggregations. Each aggregation requires processing, storage, and indexing. One way to improve performance is to cut back on the number of aggregations needed. The remaining aggregations can be spread across different partitions, again taking advantage of parallel processing.

Partitions and aggregations can be examined and modified using the respective tabs in the cube editor. However, detailed performance analysis is beyond the scope of this book. Analysis Services does provide a wizard to help design aggregations and optimize the storage structure based on usage patterns. But, the process of optimization requires knowledge of high performance computing and experience with analyzers and the cube processing. You should know that these tools exist and that experts can be found to help tune the cube browser to handle extremely large problems.





**Figure 3.46**

Choosing an analysis project and OLAP cube. Projects and cubes are displayed only if the user has permission to use them.

## Excel PivotTables

### How can the cube be accessed outside of Analysis Services?

Analysis Services and the Developer Studio are good for designing and processing OLAP cubes. However, the user interface for exploring the cubes is a little clunky. It is not something that should be given to managers. Fortunately, the Analysis Services is just that—a service. It processes and provides cube data to any front-end tool that knows how to use Microsoft's data connection methods. At the top of that list is Microsoft Excel and SQL Server Reporting Services, which means that cubes can be distributed easily through SharePoint servers as well using Power Pivot. To illustrate the process and the benefits, this section builds an Excel spreadsheet to connect to the cube, explore the data, and create a chart to highlight trends.

For many years, Excel has supported the PivotTable and PivotChart objects. These tools are a cube browser and a graphical display tool for cube data. The tools accept many types of data for input, including spreadsheets, queries to Access and SQL Server, and direct connections to SSAS cubes. Once managers understand the purpose and flexibility of OLAP cubes, the tools are relatively easy to use. The only challenge lies in connecting to the data—simply because of the number of steps and pop-up windows. Even this process can be simplified for managers. Once the connection is defined, it can be distributed to managers as a file that opens the connection immediately. Microsoft also has a PowerPivot add-in for Excel that can be downloaded ([www.powerpivot.com](http://www.powerpivot.com)). The main strength of PowerPivot is the ability to handle millions of rows of data and publish results to SharePoint servers. The tool can pull data from Analysis Services, but there is not much gain over a simple PivotTable.

Calendar	Mountain	Mountain full	Race	Road	Tour	Track	Grand Total
Calendar 1994	199,006	559,020		928,984	486,774	346,201	2,555,013
Calendar 1995	124,240	567,940		294,390	1,074,490	709,230	2,956,210
Calendar 1996	631,760	1,469,870		1,374,850	361,020	213,360	4,050,860
Calendar 1997	2,780	783,820	1,826,320	1,851,160	894,040		5,358,120
Calendar 1998	110,360	1,366,180	2,712,310	683,090	1,562,180	201,270	6,637,390
Calendar 1999		3,138,210	2,792,330	1,998,490	1,816,790	634,260	10,380,080
Calendar 2000	94,010	1,657,070	1,611,920	671,200	600,180		4,634,380
Calendar 2001	180,500	1,221,870	1,399,290	936,430	876,740	72,890	4,687,720
Calendar 2002		1,523,740	2,507,560	1,581,050	1,645,770	225,980	7,484,100
Calendar 2003		2,241,600	3,675,790	1,585,560	2,593,660	526,680	10,623,290
Calendar 2004		896,390	3,372,690	1,858,720	2,183,070	624,800	8,935,670
Calendar 2005		934,540	2,812,930	3,113,150	2,285,410	486,990	9,633,020
Calendar 2006		1,205,720	2,700,080	3,235,570	2,819,750	599,460	10,060,580
Calendar 2007		1,098,330	3,370,890	4,431,690	2,965,450	313,340	12,179,700
Calendar 2008		846,990	2,930,500	3,931,900	2,508,860	471,980	10,690,230
Calendar 2009		829,900	3,452,630	4,244,930	2,951,360	296,470	11,775,290
Calendar 2010		1,018,340	4,277,740	5,171,940	3,417,090	616,580	14,501,690
Calendar 2011		894,730	5,435,980	5,708,890	4,079,390	1,009,970	17,128,960
Calendar 2012	628,450	1,283,150	5,327,330	5,903,620	3,748,000	632,210	17,522,760

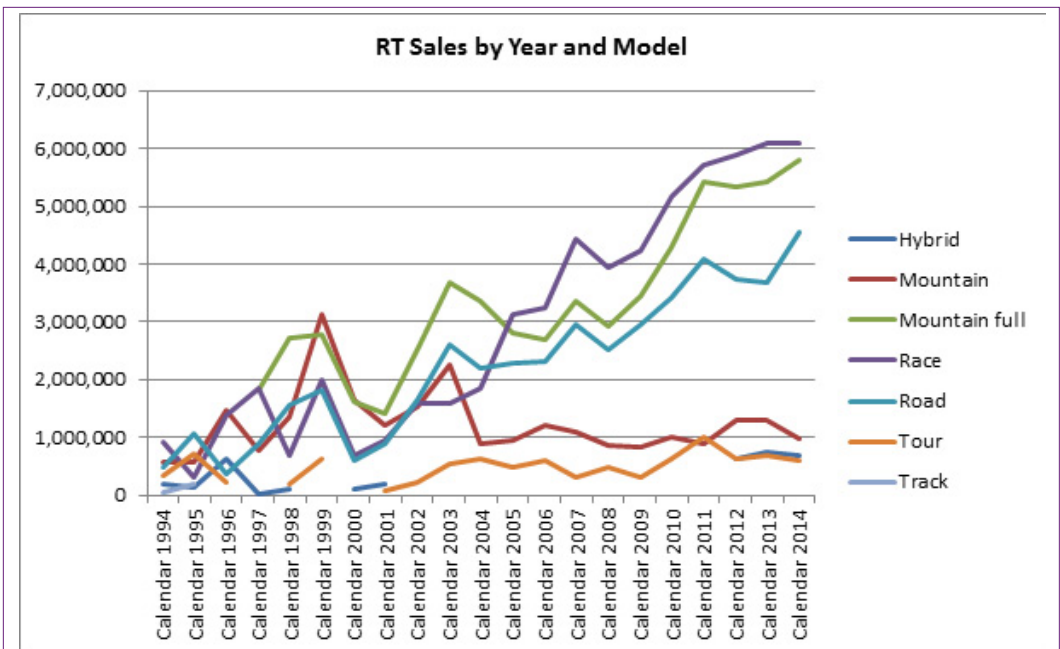
Figure 3.47

Excel PivotTable. Drag dimensions to the four boxes on the lower right. The PivotTable functions are similar to those for the cube browser in SSAS.

To create a PivotTable from scratch, open Excel and use the Insert/Tables ribbon entry to start the PivotTable wizard. Select the option to “Use an external data source,” and click the Choose Connection button. Once the connection file exists, it can be found in the list displayed on the next pop-up window. For now, click the Browse for More button. On the selection window, click the New Source button.

Figure 3.45 shows the first step in the data connection wizard. You need to enter the server name and login information to find the OLAP cube hosted by Analysis Services. If the Analysis Services is running on the client computer, the default server is simply localhost. Windows login is the easiest connection method. However, once the cube is created and built for managers, and often for students, Analysis Services will be running on a standalone server and login information will be controlled by the system administrator (or instructor).

When the connection has been established and security verified, you will be asked to choose the project and the desired cube or perspective from the list within that project. Figure 3.46 shows the basic selection screen. Large projects can have dozens of cubes. Ultimately, projects and cubes are displayed only if the user has permission to use them. Security and other administrative issues are not covered in this chapter, but the process of applying user permissions is relatively standardized. Clicking the Next button leads to the final connection screen. That page enables you to name the file that will hold the connection information as well as provide a description of the connection. Once created, this file can be used to open the cube in the future with a few clicks. Finishing the data connection wizard returns you to the initial PivotTable setup wizard. Simply click the OK button to start the PivotTable.



**Figure 3.48**

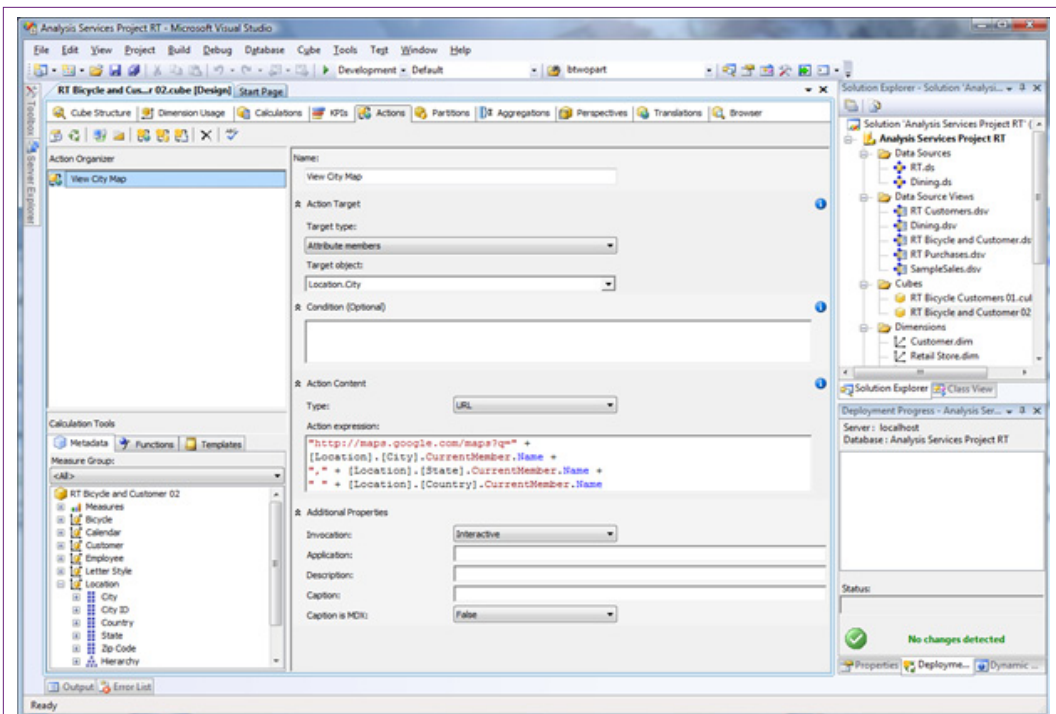
Excel PivotChart. Simply click the Chart button on the PivotTable menu. The chart automatically adjusts as the data in the table is filtered or changed.

The base PivotTable form in Excel is similar to the blank OLAP cube browser. It contains four basic locations: column dimensions, row dimensions, filter dimensions, and the main value form. These four areas are shown on the initial screen and as four display boxes in the PivotTable popup toolbar. Figure 3.47 shows one version of the cube. It is straightforward to drag dimensions to different locations and to drill down to details in the hierarchy. Standard Excel formatting options are available.

One of the nicer features of using Excel to explore the data is that a PivotChart can be created simply by clicking the Chart button in the PivotTable Options ribbon. Figure 3.48 shows one version of the chart. The chart itself is connected to the table. Changing the filters for the page, such as selecting one state, updates the table and the chart. Removing an attribute or changing a dimension automatically updates the chart display. The chart provides a visual presentation of the data that updates as the manager explores the various dimensions. It is a convenient way to let managers explore the data and see relationships. Similar versions can be deployed on in-house Web servers and through SharePoint to enable teams to share data.

## Actions

**How can the cube connect to external data such as Web sites and maps?** [Note: This section probably will not work with the newer VS 2012 Cube Browser.] The basic cube operations display various subtotals, enabling users to drill down to see details and compare data across dimensions. But it is possible to do more. Actions can be assigned to various elements of the cube. For



**Figure 3.49**

Create a new action. Right-click the top-left pane and select New Action. Enter a recognizable name, choose Attribute Members as the target type and Location.City as the target object. Create the URL link to the Web site (Google).

example, clicking a value can call a **drill through** action that displays all of the detail rows that make up the selected subtotal. Data columns can be designed to provide specific information that might be needed to answer questions.

Actions can also be defined to call external programs or open a Web site. Data values from the cube can be passed to the Web site so it can respond with matching data. This technique is useful for displaying maps and for opening SQL Server reports that carry specific information that matches the cube values. To illustrate the process, consider a simple link that opens a Google map for a selected city.

Figure 3.49 shows the basic process for creating a new action. Select the Actions tab in the cube browser. Right-click the top-left Action Organizer panel and choose the New Action option to open the editor. It is important to choose a recognizable action name—this name will be displayed to users. To create a URL mapping action, select Attribute members as the target type and Location.City as the target object. Notice that many other cube objects can be chosen for other purposes, including cell, level, dimension, and hierarchy. Choose URL as the action type and create the URL action that opens a Google map. The Google site has instructions for passing parameters on the URL line. The simplest is to send the City, State, and Country values as if they were entered as a query. The basic format is of the form:

```
http://maps.google.com/maps?q=Sacramento, CA USA
```

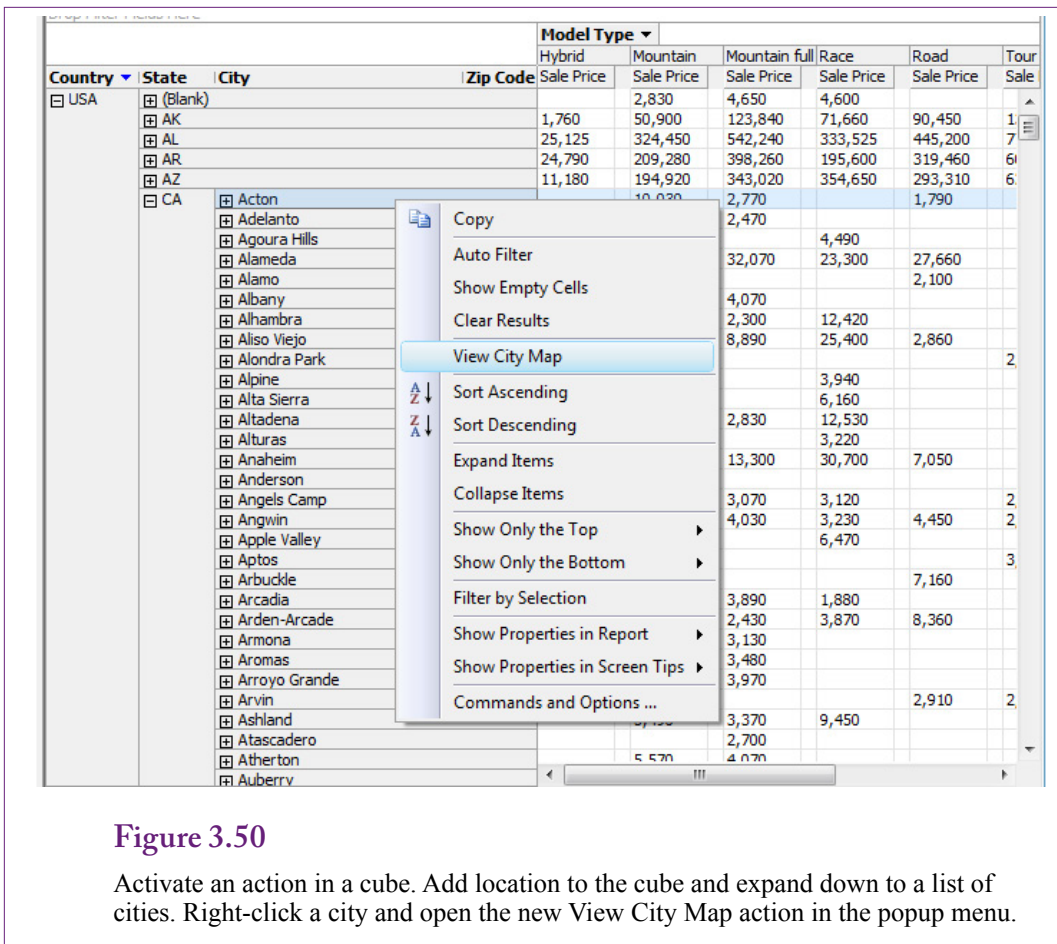


Figure 3.50

Activate an action in a cube. Add location to the cube and expand down to a list of cities. Right-click a city and open the new View City Map action in the popup menu.

The http portion of the URL is passed as a text string so it needs to be enclosed in quotation marks. The City, State, and Country are extracted from the currently selected city. You can drag the [Location].[City] and other values into the expression box, but you have to add the .CurrentMember.Name by hand. The complete expression is

```
"http://maps.google.com/maps?q=" +
[Location].[City].CurrentMember.Name +
", " + [Location].[State].CurrentMember.Name +
" " + [Location].[Country].CurrentMember.Name
```

Google maps (or Bing maps) support additional parameters, enabling the map to be displayed in a variety of ways but they are not important here.

Save everything, process the cube, and browse it. Add the location hierarchy to the rows and expand down to a city. As shown in Figure 3.50, right-click a city to see the pop-up menu. Note the new action: View City Map. Select the action and a browser should open and display that city in Google maps. Relatively sophisticated actions can be built using Web sites and custom programming, but they are beyond the scope of this book. The main point is to remember that a cube can be extended to support many different actions. Complex problems will require the assistance of a programmer.



## Key Performance Indicators

---

### How can simple data be provided to managers on a daily basis?

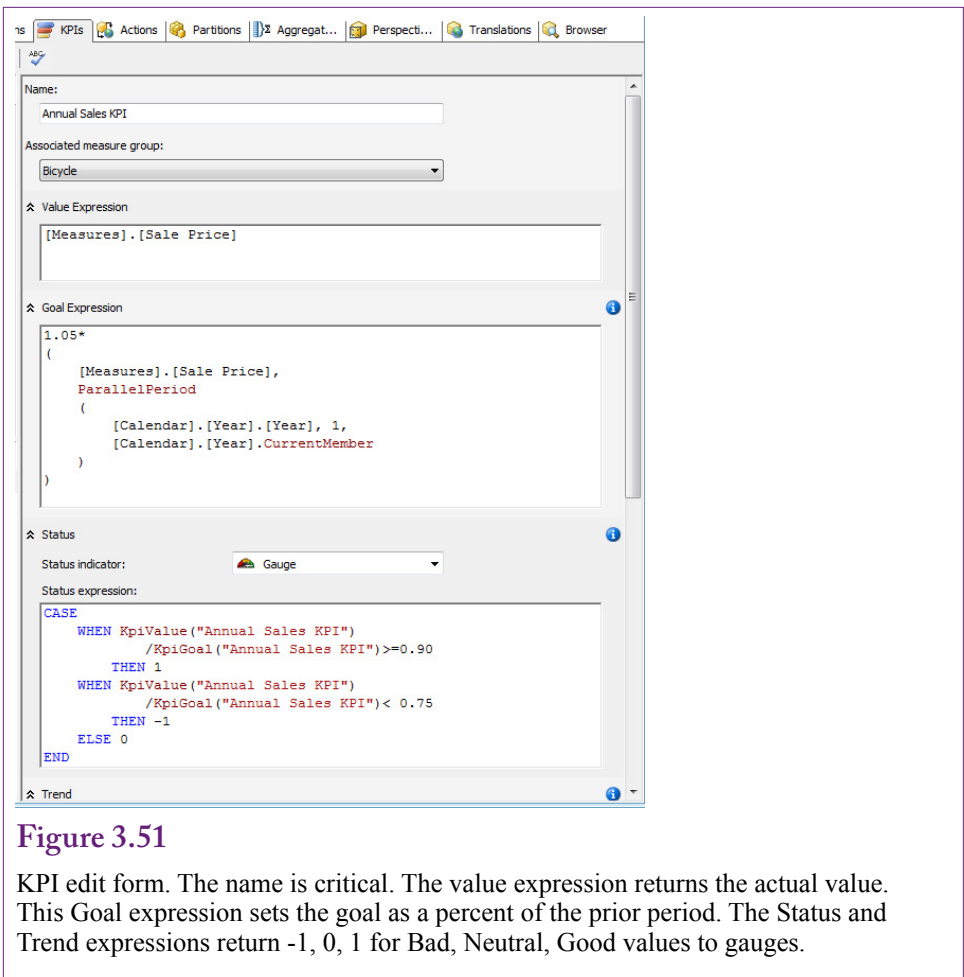
The OLAP cube is a useful tool but it requires active exploration by managers and analysts. It is a useful way to examine data from a variety of perspectives. It is useful when searching for specific comparisons—because it does not require writing SQL and because results are almost immediate. But, cube browsing could be time consuming if managers had to use it to look up similar items every day. For example, perhaps executives simply want to know what happened to sales for the year, quarter, month, or week. Or, regional managers want a quick look at salary expenses for the past month. These types of values could be found through the cube, and the cube can show comparisons to the last period or the same period a year ago. But managers will quickly grow tired of having to fire up the cube browser, select the basic filters, and search for the results. These numbers are so common and so basic that they want a simpler way to see the values quickly.

### Definition

A key performance indicator (KPI) is a piece of data that informs managers about a specific measure. The item represents some aspect of the organization that is viewed as important. The item provides a measure of progress, and its value over time is an important indicator of future results. In business, sales revenue and various expenses are often useful KPIs. In line management, perhaps quality, quantity, and cost numbers are more important. The point is that each level of management, and even each manager, has a different collection of KPIs.

Managers want a way to see the current values of KPIs, to see them on a daily basis, and to be warned of patterns and changes. One solution is the digital dashboard, which provides gauges showing the values and trends of various KPIs. Similar in concept to the dashboard of a car (or cockpit of an airplane), the KPI gauges help managers evaluate the status, direction, and trends of various business factors to guide the company.

Analysis Services provides a method to define KPIs and store those definitions on the server. Various client tools can then query the server, which automatically retrieves the data, runs the KPI code, and returns values that can be used to set the values of gauges. KPIs are defined in terms of several expressions to compute: Value, Goal, Status, and Trend. The Value is the current value of the measure, such as annual sales. A Goal is a target value for the measure. It might be a value specifically entered by managers—providing a target level of production or sales for each region. Or, it might be expressed as a percentage change value—the amount the company wants to increase sales. Status is a value designed to be used by gauges and other indicators. It returns a value between -1 and 1, with zero a neutral indicator. Technically, status values are continuous and can return any level between -1 and 1 (inclusive). However, many systems simplify the results and return one of only three values: -1 (poor), 0 (neutral), and 1 (good). Some client gadgets can handle only those three states. For example, a traffic-light icon uses red, yellow, and green lights to show the status value. The fourth item of Trend is similar. It is also used to set an indicator icon. Typically, the icon is an arrow (up, down, horizontal) to show three trend values (increasing, decreasing, and neutral).



**Figure 3.51**

KPI edit form. The name is critical. The value expression returns the actual value. This Goal expression sets the goal as a percent of the prior period. The Status and Trend expressions return -1, 0, 1 for Bad, Neutral, Good values to gauges.

## Creating KPIs

The benefit to creating a KPI in Analysis Services is that the definitions are stored centrally, so the same data is available to everyone who has access to the KPI. When the measures need to be changed, they are changed in one location. Also, the KPIs use the aggregated cube data, so the results are computed quickly and efficiently.

Creating a KPI has one important challenge: The definitions of the KPI values are written in the **multidimensional expressions (MDX)** query language. MDX is a query language for OLAP cubes first defined by Microsoft, but several vendors support it in their products—including various design tools for client-side displays. MDX is powerful, bears no relationship to SQL, and has some useful functions that make it relatively easy to create KPIs. But, solving complex problems requires an in-depth knowledge of MDX. Instead of trying to cover all of the details of MDX in this chapter, the examples focus on commonly used types of queries. These examples have some limitations, but they are easy to set up and can be modified for many common situations.

To create a new KPI, click the KPIs tab in the cube editor. Right-click the KPI Organizer panel in the top-left of the editor and select New KPI. As shown in



Figure 3.51, the edit form shows the sections that need to be created: Name, Value Expression, Goal Expression, Status Expression, and Trend Expression. Figure 3.51 shows most of the edit form. The Trend expression and some optional parameters at the bottom of the form are not shown.

The Name is critical because it must indicate the purpose of the KPI to managers so they choose the right KPI for their desired purpose. This KPI will use the Bicycle Sale Price total to measure annual sales. KPIs can also be stored in designated folders—making it possible to group related KPIs together. The Additional Properties section at the bottom of the form has options for setting a Display folder and a Parent KPI for KPIs that are hierarchically related (such as Annual, Quarterly, Monthly, and Weekly sales). The section also contains a box for Description to better explain the KPI.

The Value Expression is the main definition of the KPI. This is the value that will be returned to any client that calls the KPI. To obtain sales revenue, open the Measures section in the Metadata list and drag the Bicycle Sale Price measure into the expression box to set the MDX formula:

```
[Measures].[Sale Price]
```

Goal Expressions are useful because they can be used to set the status charts. The presumption is that some Goal exists for the Value. Perhaps each sales region has been given a target goal or each factory a production quota. If these goals are set by management, a separate column attribute needs to be defined in the database to hold these numbers for each time period. In that case, the expression is as simple as the Value expression—simply drag the goal measure into the expression box. Another approach to setting goals is to compute the target as a percentage of the sales in the prior period. A special MDX function called **ParallelPeriod** is used to retrieve the value from the prior period. The function has three parameters: (1) The hierarchy level, (2) The number of periods, and (3) The current value. The syntax for the 5% increase is:

```
1.05*
(
    [Measures].[Sale Price],
    ParallelPeriod
    (
        [Calendar].[Year].[Year], 1,
        [Calendar].[Year].CurrentMember
    )
)
```

The 1.05 value at the top sets the percentage increase. The [Measures].[Sale Price] sets the measure to use, and the ParallelPeriod function returns the matching value for the prior calendar year. The function can be used to compare values using a fiscal year calendar instead—as long as the fiscal year calendar is defined as a hierarchy dimension. Note that the [Calendar] name must match the name of the hierarchy dimension in the OLAP cube.

More importantly, note that the goal expression is only defined for years. Interestingly, the KPI will display output when applied to quarterly data, but the numbers would not make much sense. The goal value is always computed in terms of the annual data, even if the display is based on a different level. Hence, it is important to include “Annual” in the name of the KPI.

The Status expression must return values between -1 and 1 because the values will be used to set values for visual icons. The edit form provides the ability to choose the type of indicator. A gauge is the default choice. Typically, the status indicates how close the actual value is to the goal. Often, it is convenient to use only three categories: poor (-1), neutral (0), and good (1). Some gauges support continuous measures, but it can be challenging to convert real-world numbers into an appropriate scale. A basic CASE statement can be used to assign three values in the Status Expression:

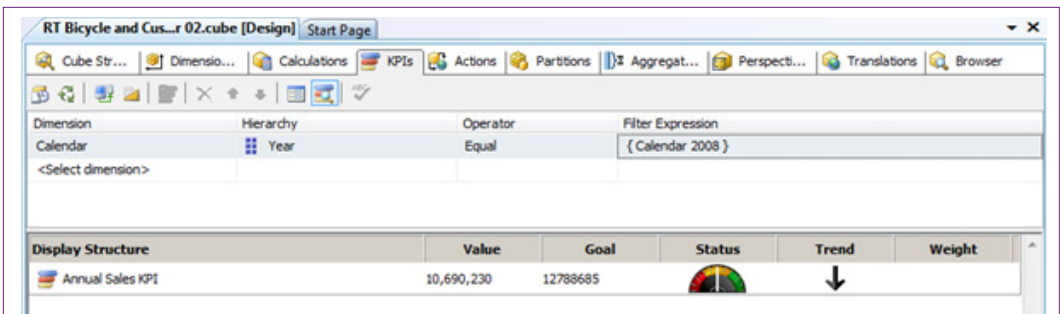
```

CASE
  WHEN KpiValue("Annual Sales KPI")
    /KpiGoal("Annual Sales KPI")>=0.90
    THEN 1
  WHEN KpiValue("Annual Sales KPI")
    /KpiGoal("Annual Sales KPI")< 0.75
    THEN -1
  ELSE 0
END

```

Notice that this formula uses the existing definitions for Value and Goal to compute a simple percentage. If the actual sales are 90 percent of the goal or higher, it is considered good. Sales less than 75 percent of the goal are bad, and everything in the middle is neutral. Again, note that these numbers are subjective, and they are hard-wired into the expressions. A few tricks exist to put the numbers into measures that can be edited by managers, but the tricks require considerable knowledge of MDX.

The Trend Expression also drives an indicator icon—typically an arrow. So it should return values -1, 0, or 1. The most common approach to show a trend is to compute the percentage change from the prior period. If the change is sufficiently positive, the arrow should point up (1). If the change is negative, the arrow should point down (-1). Anything in the middle should be displayed as a horizontal arrow (0). The formula for percent change is  $(\text{new-old})/\text{old}$ , but it can be simplified to just  $\text{new}/\text{old} - 1$ . This version has the benefit of needing to use the prior value only once instead of twice. Remember that the prior value is obtained using the ParallelPeriod function and using it twice makes the expression difficult to read. The Trend expression becomes:



**Figure 3.52**

KPI browser. The two icons in the tool bar under the KPIs tab switch between edit and browser mode. Testing works best when the Calendar dimension is dragged to the filter bar near the top of the form. Select Year as the Hierarchy level and pick a specific year from the Filter Expression list.

```

CASE
  WHEN
    (KpiValue("Annual Sales KPI") /
     (KpiValue("Annual Sales KPI"),
      ParallelPeriod(
        [Calendar].[Year].[Year], 1,
        [Calendar].[Year].CurrentMember
      ) ) - 1
    ) <= -0.05
  THEN -1
  WHEN
    (KpiValue("Annual Sales KPI") /
     (KpiValue("Annual Sales KPI"),
      ParallelPeriod(
        [Calendar].[Year].[Year], 1,
        [Calendar].[Year].CurrentMember
      ) ) - 1
    ) > 0.05
  THEN 1
  ELSE 0
END

```

Again, two “WHEN” statements set the high and low values, leaving everything else in the middle at zero. The syntax for the division (new/old) is a little hard to read because of the parentheses. Just remember that the old value is the term in parentheses that uses the ParallelPeriod function. This example sets the low and high levels to negative and positive 5 percent.

### Browsing a KPI

It is important to understand that Analysis Services stores the MDX Expressions for a KPI. It does not actually compute the values until the KPI is queried. KPIs are designed to be used by other applications running on client computers. For example, a Web page or SharePoint server page might include a reference to several KPIs. In many cases, a KPI can be embedded on a desktop through a specific

gadget. Periodically, the gadget requeries the server to get current values for all of the KPIs. This timing control is built on the client computer. Analysis Services is passive and simply returns the values when asked.

It is also important to test the KPI expressions. It is easy to misplace a parentheses or comma. Fortunately, the Analysis Services KPI editor includes a simple browser to test and display the values. A small icon hiding on the toolbar switches the editor to browser mode. A similar icon next to it returns to edit mode.

Figure 3.52 shows the browser form for the Annual Sales KPI in Calendar year 2008. The KPI icons will display with the default values of all years, but the goal cannot be computed, and the icons do not make much sense. Instead, it is best to select a single year to evaluate the KPI. Drag the Calendar hierarchy dimension from the metadata list onto the filter section at the top of the form. Select Year as the level in the Hierarchy column. If the KPI was built for Quarter, Month, or Week, choose the appropriate level. The operator defaults to Equal, so choose a single year in the Filter Expression list. Be sure to click on the Display list after selecting or changing a year value. The display is updated only after it is clicked. It is good practice to choose a couple of adjacent years; record the values and the goals and manually check the calculations to ensure the goal is computed correctly. It is also interesting to test a few years to see what happens to the gauge and trend icons.

KPIs have additional properties that can be used for more complex cases. The Weight is used for groups of KPIs—typically a set of KPIs that have the same parent. For instance, Sales might be divided into sales by region. The regional KPIs could be assigned percentage weights that represent the relative size of each region. Parent assignments and weights are set in the “Additional Properties” section of the edit form.

## Summary

---

Online analytical processing was created because storing data efficiently and retrieving data quickly lead to conflicts. The indexing, pre-computed totals, and duplication of data needed to retrieve data in seconds or less causes problems with saving new data. OLAP cubes are designed conceptually and physically around the concepts of dimensions and measureable facts. The star design connects dimensional attributes directly to the fact table. The snowflake design allows links from dimensional tables to other tables. Many dimensions have a hierarchy of values. Time or dates are perceived at levels from years to quarters to days. Geographic hierarchies are commonly used to group locations.

To managers and analysts, a hyper cube is essentially a large collection of sub-totals (or averages) based on many possible dimensions. Hierarchies provide support for drill down and rollup to see details or summaries at any level. Filters are used to slice the cube and compare values for different attributes.

Creating a cube is straightforward with Analysis Services and its wizards. The primary steps are: (1) Connect to at least one data source, (2) Create a data source view that connects the fact table to the dimension tables, (3) Choose measures and dimensions for a cube, (4) Improve the dimensions and add hierarchies. Data source views are the key method of providing data to the data mining tools.

Cubes are designed to perform summary calculations—usually sums of numbers. However calculations can be defined at two levels: (1) Aggregate calculations applied to the subtotals, or (2) Row-by-row calculations performed at the query (view) level. Any calculation requiring multiplication or division should

take place at the query level. But, if the detail values are percentages, the aggregation should be changed to averages across the children instead of sums.

Analysis Services is a server—it defines the cubes and efficiently retrieves the data. It might not be the best browser. Excel and Reporting Services provide additional options for browsing cubes, including options to place cubes on internal Web sites. For instance, interactive charts are easier to create in Excel than in Analysis Services. Actions can be defined to provide more detailed tools and options to browsers. For example, it is straightforward to add Web hyperlinks to data so users can link to more detailed data or even Web pages with maps and descriptions. Key performance indicators can be defined on the server and accessed by client applications to display gauges and other icons that indicate trends in important data measures.

## Key Words

---

aggregation	key performance indicators (KPI)
attribute relationship	measure
Business Intelligence Development Studio	metadata
calculation	multidimensional expressions (MDX)
cube browser	multidimensional OLAP (MOLAP)
data warehouse	named calculation
digital dashboard	named query
dimension	online analytical processing (OLAP)
drill down	online transaction processing (OLTP)
drill through	order of operations
extraction, transformation, and loading (ETL)	ParallelPeriod
fact	perspective
hierarchy	relational OLAP (ROLAP)
hybrid (HOLAP)	rolled up
immediate if function (IIF)	snowflake
index	star design
	view

## Review Questions

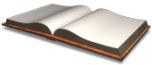
---

1. Why is MOLAP better than ROLAP for most large applications?
2. Why does setting up extraction, transformation, and loading often take a long time?
3. How is the snowflake design different from the star design?
4. What are the primary steps in creating an OLAP cube in Analysis Services.
5. When should time hierarchies be built on the server?
6. What are the major benefits to storing hierarchy data in tables in the original database?

7. What is the purpose of attribute relationships in a hierarchy? What constraint or issue is imposed when building them?
8. What problem arises when defining calculations at the cube level? How are these resolved?
9. What is the point of adding actions to a cube?
10. What are four main components of key performance indicators

## Exercises

---



### Book

1. Create the main Rolling Thunder Sales cube with the time and geography hierarchies.
2. Create the Percent Discount column in the record source view and add it to the existing cube using an average instead of sum.
3. Define two perspectives on the existing cube. One that focuses on sales by employee and one that focuses on customer purchases without the employee data.
4. Modify the model type table to add a second language. Translate the model types (use a Web translator if necessary). Add the new language to the cube.
5. Create an Excel PivotTable that connects to the cube and build a PivotChart.
6. Add the URL action to open Google maps for the location.
7. Define and test a key performance indicator for monthly sales.



### Rolling Thunder Database

8. Create a new cube that focuses on purchases from manufacturers.
9. Create a new cube that focuses on component inventory. Use the cube to identify any problems the firm has with respect to inventory.
10. Identify at least three KPIs other than total sales that could be useful to managers of Rolling Thunder Bicycles.



## Diner

11. Build a cube for the Diners database.
12. Create an Excel PivotTable and PivotChart that will help managers. Identify the primary decisions the chart is designed to improve.
13. What KPIs would be useful for managers of the restaurant?



## Corner Med

14. Build a cube for the Corner Med database that focuses on the patient visit, diagnoses, and treatments.
15. Build a cube for the Corner Med database that focuses on the work output and revenue generated by the employees.
16. Getting payments is always a problem at medical offices. Build a cube, PivotChart, and KPIs that can be used by managers to monitor payments by insurance companies.



## Basketball

17. Build a cube that helps coaches of individual teams track the performance of their players.
18. Build a cube that helps coaches evaluate players across the league to help decide who to recruit. Create an Excel PivotTable.
19. Identify at least three KPIs that might be useful for a coach during the season.



## Bakery

20. Create a cube for the Bakery database that helps managers explore sales. Use an Excel PivotChart to make it easier for managers to understand the data.
21. Note that Sales Date for the bakery also includes the time of day. Create or modify the sales cube to include a dimension for time of day that splits the day into three time periods: morning, lunch, and afternoon.
22. Identify at least three KPIs that would be useful for managers of the bakery.





## Cars

23. Create a cube that enables potential car buyers to evaluate the cars.
24. Modify the cube to include an action link that brings the user to a Web page for the manufacturer's Web site. It is probably not possible to link to a specific vehicle, but check at least one of the independent Web sites to see if there is a way to link to a specific vehicle.



## Teamwork

25. Split the team into subgroups of two people and assign a language to each subgroup. Modify the Rolling Thunder Bicycles database to incorporate each of the new languages, translate the dimension data, and modify the cube to handle each language.
26. Have each team member choose a different basketball team. Using a cube and PivotCharts for analysis, identify the best players on the team. Note, these evaluations must be based on data, not personal opinion. Combine the individual results and make a case for choosing the best player among the group.
27. Using the bakery data, assign product categories so that each team member evaluates at least one category and all of the categories are evaluated. Using the cube analysis and charts, identify any patterns, trends, or problems with sales of each category. Combine the results and identify any categories that should be used to highlight an advertising campaign.

## Additional Reading

---

Harinath, Sivakumar, Matt Carroll, Sethu Meenakshisundaram, Robert Zare, Denny Guang-Yeu Lee, 2009, *Professional Microsoft SQL Server Analysis Services 2008 with MDX*, Wrox/Wiley: Indianapolis. [Detailed how-to discussion of many steps in SSAS. Most of the authors were Microsoft employees working on SSAS, so the book is loosely the documentation Microsoft did not create.]

Jacobs, Adam, 2009, "The Pathologies of Big Data," *Communications of the ACM*, 52(8), 36-44. [One commentary on problems with relational databases. He tried to load 6.5 billion rows into a relational database.]

---

# Probability and Statistics Summary

## Chapter Outline

Introduction, 156	<i>Chi-Square Hypothesis Tests, 198</i>
Probability Basics, 156	<i>Information Measure, 200</i>
<i>Discrete and Continuous Data, 158</i>	Summary, 201
<i>Counting and Combinations, 158</i>	Key Words, 203
<i>Probability Rules, 161</i>	Review Questions, 203
Interdependencies: Joint Probabilities, 164	Exercises, 204
<i>Contingency Tables, 165</i>	Additional Reading, 209
<i>Tree Diagrams, 166</i>	
<i>Bayes Theorem, 167</i>	
Probability Distributions, 171	
<i>Discrete Data, 171</i>	
<i>Continuous Data, 176</i>	
<i>Joint and Conditional Probabilities, 177</i>	
<i>Expected Value (Mean) and Variance, 178</i>	
<i>Important Continuous Distributions, 182</i>	
Statistics, 190	
<i>Samples, 190</i>	
<i>Common Statistics, 191</i>	
<i>Confidence Intervals, 193</i>	
<i>Hypothesis Testing, 195</i>	

## What You Will Learn in This Chapter

- How is it possible to make decisions when so many events are random?
- What is randomness and what rules does it follow?
- How are probabilities handled when events are not independent?
- How can probability concepts be generalized to for handling new situations?
- How are statistics used in data mining to find interesting results?

### Human Biases

In general, people make many mistakes when evaluating data. One particular problem is known as “confirmation bias,” where people seek information that confirms what they already believe. Business people and even professional researchers are prone to this problem. When we create theories, we like to “prove” that they are right. A psychology study with nearly 8,000 participants concluded that people are twice as likely to seek information that confirms their beliefs instead of seeking out conflicting data to disprove them. Psychologist Scott Lilienfeld of Emory University noted that “We’re all mentally lazy. It’s simply easier to focus our attention on data that supports our hypothesis, rather than to seek out evidence that might disprove it.” Consequently, people are reluctant to change their opinions and models, and instead rationalize why things might have gone wrong. And gathering more data makes the problem worse. If you are focusing on confirming beliefs, then new data reinforces those beliefs, reducing the value and diversity of the information available. [Zweig 2009]

Try to collect unbiased data first then analyze it with an open mind. Keep notes during the early stages to record options and uncertainties. Return to those notes later to explore alternatives. Involve other people and look for options and criticisms.

Jason Zweig, “How to Ignore the Yes-Man in Your Head,” *The Wall Street Journal*, November 13, 2009. [http://online.wsj.com/article/SB10001424052748703811604574533680037778184.html?mod=WSJ\\_hps\\_LEADNewsCollection](http://online.wsj.com/article/SB10001424052748703811604574533680037778184.html?mod=WSJ_hps_LEADNewsCollection)

## Introduction

---

**How is it possible to make decisions when so many events are random?** Randomness exists in many aspects of the world and business. Many events are subject to fluctuations and occurrences that cannot be defined with certainty. The business world is not deterministic. Instead, the same action taken at two points in time can lead to different results. One manager can make a decision and the results can work out well. A second manager facing the same problem can make the same decision and have everything fail. Of course, the first manager gets a huge bonus and writes a book proclaiming the brilliance of the decision, and the second manager looks for a new job. But, perhaps instead of brilliant, the first manager was merely lucky. So how is it possible to make decisions, and evaluate decision makers, when the world is random?

The science of probability and statistics was developed specifically to deal with questions of decision making under uncertainty. It defines the concepts of randomness and describes mathematical rules that determine relationships among events. Statistical tools developed over centuries of research form the foundations of data mining and decision making.

The field of probability and statistics is large and complex. It is based largely on mathematics, and some of the most powerful theorems were found with the use of advanced mathematical concepts. Fortunately, it is possible to use the results of this work without needing to understand the heavy mathematics. Yet, to understand the results of some of the tools, it is important to know some of the basic foundations and definitions. This chapter presents the basic concepts needed to use and understand the results of common data mining tools. It begins with a summary of some key probability concepts. It also explains some of the critical concepts in statistics that are commonly used in decision making problems and business intelligence. The chapter will be easier to read if you have already had an introductory course in probability and statistics, but the text does define the fundamental concepts needed for the rest of the book. The main goal of the chapter is to develop a statistical perspective and improve your “intuition” or understanding of how probability and statistics are understood and evaluated in making decisions.

## Probability Basics

---

**What is randomness and what rules does it follow?** Randomness can arise from many sources, but those details are not critical yet. For the moment, **random** simply means that some events or outcomes cannot be predicted with complete certainty. Instead, an outcome is said to have some probability of arising. The concept of probability is critical to understanding randomness and statistics. Technically, two good definitions of probability exist. Ultimately, the two definitions lead to similar conclusions and applications, but it is sometimes useful to look at the world one way or the other. The oldest and most common definition of probability is known as **relative frequency**: The **probability** of an event arising is equivalent to the number of times the event can arise versus the total number of events that can occur. The simplest example is a coin flip with the events of a head or tail appearing on top. With two total events possible (staying on an edge is considered a bad toss and ignored), the relative frequency or probability of either event is  $1/2$ . Rolling a six-sided die is another common example, where the total number of possibilities is 6 and each side has the same probability of arising, so the chance of any single number appearing is  $1/6$ .

Customer Gender	Car Model	Customer Age	Car Weight	Car Price
Male	Focus	21	2588	15520
Female	TT	42	2965	35200
Unknown	Suburban	36	5607	40370

**Figure 4.1**

Examples of discrete and continuous data. Gender and Model are discrete because the values are specifically defined and can be counted. Weight, Price, and even Age are continuous data. Although the values displayed are truncated, the attributes could take on any value.

In probability calculations, the list of total events has to be complete, so the probability of any event must always lie between 0 and 1. Writing probability as  $P$ , and using a generic outcome  $A$ , the probability of any event cannot be less than zero or greater than one:

$$0 \leq P(A) \leq 1$$

The function  $P(A)$  is read as “the probability of  $A$ ” where  $A$  is some event. For instance, the probability of tails occurring from a coin toss is 1 out of 2 or  $1/2$ . Probabilities of all outcomes must add to one. Sometimes it is useful to refer to all other outcomes as the negation, or not  $A$ . It is often written  $P(A')$  and read as “the probability of not  $A$ .” So,  $P(A) + P(A') = 1$ .

The second way to define probability is known as the **Bayesian** approach for reasons that will be clear later. In this method, probability is subjective and defined as the degree of belief of some event happening. The interesting twist is that any individual could have a different belief about the probability of an event. The probability must still fall between zero and one, but it is **subjective**. For instance, perhaps the person flipping the coin knows how to flip it so that heads appears more often than half the time. Alternatively, perhaps the probability belief by one person, or entire organization, is biased, or even wrong. As a child with limited experience, perhaps you believed that when flipping a coin, the head could appear more often than tails. Probability theory reveals that someone with this belief would likely be wrong many times, but the beliefs would change with experience or increased information. Subjective probability is an interesting and useful way to approach some problems, and some types of data mining are based on this approach. Instead of looking at coin flips, consider the question of predicting the level of sales for next year. Each range of outcomes has some probability of occurring that is unknown and could be subjective. As more information is collected (say daily or weekly sales), these probabilities could be adjusted to provide a more accurate forecast.

The issue of subjective probability highlights a second key aspect of probability. The actual probability of an event might be unknown. Some events are simple enough to enumerate or count and the probabilities can be defined. With more complex events, estimating the probability is an important step in forecasting and statistics.

## Discrete and Continuous Data

Two types of data often exist in problems: **discrete** and **continuous** measures. Discrete events can be counted—there might be an infinite number of options, but each one is unique and can be assigned a number. Examples of discrete data include the coin flip, the sides of a die, the gender of a customer, whether a product is defective, a car model, and the types of products produced in a bakery.

Continuous data is measured in real values instead of integers, and it cannot be counted as simple groups. Examples include the gas mileage of a car, the height of a customer, the temperature of a food item, and the weight of a bicycle. Some measures might first appear to be discrete, but the measurement might be truncated, and it could take on any value. For instance, the age of a person is commonly truncated to years; but if the date of birth is available, the age could be measured in days. In fact, the age of a person is continuous data because time could be divided into any level of measurement. Similarly, the price of an automobile would be continuous data—particularly measured in cents. Figure 4.1 shows some simple examples of discrete and continuous data. The Customer Gender and Car Model are clearly discrete because the options are well-defined. Car Weight, Car Price, and Customer Age are continuous data. The values displayed are truncated to integer values, but given appropriate measuring tools, the attributes could take on any value. Car price might (or might not) be limited to cents, which might appear to make it a discrete measure. But, because cents is such a small fraction of the overall value, prices are typically treated as continuous data.

Continuous and discrete data are treated differently in statistics and so the data mining tools are different as well. Interpreting results and making decisions is somewhat different as well. Because some tools work only with discrete data, it will sometimes be beneficial to convert continuous data into discrete groups. **Discretizing** continuous data is an important step in some of the procedures. For example, the weight of a car could be classified into three discrete categories: (1) Light where weight < 2000 pounds, (2) heavy where weight > 4000 pounds, and (3) medium with weights between 2000 and 4000 pounds. It can be difficult to determine the number of categories and the cutoff values. Sometimes they are established by experts or tradition. In other cases, some tools exist to help identify appropriate categories.

The differences between discrete and continuous data appear in many chapters, and the statistical implications are covered in more detail in the statistical sections of this chapter. For the moment, basic probability concepts are easier to understand with discrete events instead of continuous.

## Counting and Combinations

Look again at the basic definition of probability: the number of ways an event can occur divided by the number of total events. For discrete data, this definition requires knowing how to count. Sure, that seems easy; but it means knowing how to count the number of ways a specific event can occur. For simple problems, such as the coin flip or defective/not-defective, the number of cases is easy to specify and counting to two is easy. More complex problems with multiple outcomes and interactions can be harder to count. Also, it can take a while to list thousands or millions of possible outcomes. Mathematicians have developed tools to help count outcomes for certain cases that arise quite often.

### *Counting when Order Matters*

A marketing manager wants to use standard capital letters to assign codes to products. The firm currently has 250 different products, and is planning to introduce another 30 per year. The manager has suggested using a three-letter code (e.g., AAB, ABC, DFG) to identify product types. Is this code large enough to handle all of the products? This problem has two important characteristics: (1) Order of the letters matters, and (2) letters can be repeated or reused. For an example of (1), ABC is different from CBA even though both use the same letters. To illustrate (2), AAB and DDD are both legitimate codes even though letters have been used more than once. This type of problem is one of the simplest to count. With three letter positions available, each one is different, and each can hold any of the 26 letters in the English alphabet. So, a three-letter code has the following number of possibilities:  $26 \times 26 \times 26 = 17,576$ . This number is clearly large enough to handle well over 100 years of new products ( $100 \times 30 = 3000$ ). Would a two-letter code be large enough? Maybe, because  $26 \times 26 = 676$ , but subtract the original 250 products, and the remaining codes would last for a little over 14 years. In a similar question, assume a state assigns car license plates using three letters and three digits. How many unique plates can be made? The answer is  $26^3 \times 10^3$  or about 17.5 million. That seems like a large number, but go back and talk to New Jersey officials in the 1990s—when the state ran out of numbers. The car problem is complicated if the state wants to avoid reissuing numbers for a couple of years.

Counting becomes more complicated as constraints are added to the problem. Consider an example of a presentation where the top officers of the company will be seated on a platform in a row. How many ways can these people CEO, CFO, HRM, CIO, and CMO be arranged? Notice that the list consists of unique items, so duplicates do not exist. Simple **arrangements** are relatively easy to count. Start with the first location. With  $n$  items, there are  $n$  possible choices for the first spot. After one of those is selected, there are  $n-1$  possibilities for the second slot,  $n-2$  for the third and so on down to one person for the last position.

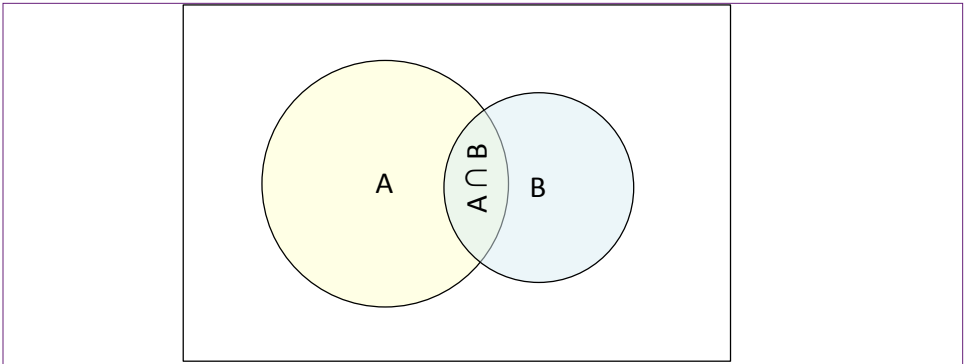
$$\text{Number of Arrangements} = n!$$

The exclamation point means factorial, or  $n \times (n-1) \times (n-2) \times \dots \times 1$ . Technically,  $0!$  is defined as equal to one. So, the presentation would have  $5!$  or 120 arrangements. But, change the problem slightly: The CEO has to sit in the middle seat. How many arrangements exist now? Many counting problems are exercises in logic and definitions. This one is straightforward to solve once you realize the problem now involves arranging only four people. So the answer is  $4!$  or 24 possibilities. This number is much smaller than the  $5!$ , so factorials grow rapidly. Perhaps with only 24 arrangements, the remaining executives will be able to select their preferred seats in less time.

A variation of arrangements is to add a new constraint. A **permutation** is an arrangement of items but some of them are not displayed. Consider a situation where a company has a showroom or Web site for products. A retail Web site wants to showcase its daily sales items on its main page. The company puts 20 products on sale each day and wants to build a display ad that showcases three products in a row, showing each product for 15 seconds before moving to the next. Each time a person opens the main page the display should feature a different group of three products or at least in a different order. How many permutations exist? The count of the number of permutations of  $n$  items taken  $k$  at a time is:

$$\text{Number of permutations } P(n,k) = n! / (n-k)!$$





**Figure 4.2**

Venn diagram of two events A and B. If events A and B can overlap—both happen together, then the circles overlap and the overlap is the portion of space where both A and B occur.

This formula can be derived much the way the arrangements formula was created. Start at the first position with  $n$  items, the next has  $n-1$  and so on. The difference is that the progression stops when you run out of spots ( $k$ ). Write out the formulas for  $n!$  and  $(n-k)!$  to see that dividing creates:  $n(n-1)(n-2)\dots(n-k+1)$ , or the numbers in sequence from  $n$  until all  $k$  spots are filled. In the example, the answer is  $20! / (20-3)! = 20! / 17! = 20 \cdot 19 \cdot 18 = 6,840$  different displays.

#### *Counting when Order does not Matter: Combinations*

Another constraint that often arises in problems is that order might not matter. Typically, in lottery games, picking the three numbers is important, but they could be chosen in any order—particularly when the numbers are unique and drawn without replacement. In business, it can be important to know which items are purchased together, but it rarely matters if one item was placed into the basket or on the counter before another one. A **combination** is a collection of items ( $k$ ) out of a total set ( $n$ ) where order does not matter.

Consider the shopping basket example. If a company sells 20 products, how many different shopping baskets exist that contain exactly 5 items? The order of the items is unimportant, so a basket containing items A, B, C, D, E is equivalent to one with E, D, C, B, A. The answer is to take the number of permutations (where order does matter) which is given by  $n! / (n-k)!$ , and divide by the number of possible arrangements ( $k!$ ). This formula appears in several areas of mathematics (e.g., Pascal's triangle and the binomial formula) and is important enough to have its own symbol:

$$\text{Number of combinations} = C(n, k) = \binom{n}{k} = \frac{n!}{(n-k)!k!}$$

In the example,  $n=20$  and  $k=5$ . Excel has a COMBIN function that computes the value directly. To find the value by hand, compute the permutation then divide by  $k$  factorial to get  $20 \cdot 19 \cdot 18 \cdot 17 \cdot 16 / 5 \cdot 4 \cdot 3 \cdot 2 = 15,504$  different baskets.

Other problems exist with different constraints, but most are derived from these formulas.

### *From Counting to Probability*

The counting formulas are used for discrete data to determine both the denominator (all possible ways) and numerator for various probability computations. Consider the example in the previous section. A company has 20 products, identified as A, B, C, ..., T. Assume the items are equally-likely to be purchased ( $p_i = 1/20$ ) and the purchases are independent. What is the probability that a customer who purchases exactly 5 items buys both items A and B? The solution is to find the number of ways that A and B and three other items can be purchased and divide by the number of ways that any 5 items can be combined. The denominator is the total combination and is straightforward:  $C(20,5)$ . The numerator is slightly tricky. Write it as five slots with two of them filled: A, B, \_\_, \_\_, \_\_. Because A and B are fixed, the numerator asks for the number of ways to find 3 items out of 18 remaining products or  $C(18,3)$ . Compute the two combination counts and divide to get:  $816 / 15,504 = 0.0526$ . There is slightly more than a five percent chance that any two specific items will be purchased in a 5-item cart when 20 products are available. The trick is to identify exactly what needs to be counted and which rule applies based on the conditions of the problem.

### Probability Rules

Specific rules have been found that make it easier to work with probabilities and probability functions. They deal with situations where multiple events can arise. The two most powerful are the addition and multiplication rules. Before defining the rule, it is important to understand how two events can interact. Figure 4.2 shows the common Venn diagram for two events. Event B was deliberately drawn with a smaller circle to indicate that the two events do not need to be equal. The point where the circles overlap represents the fact that both events could occur. If there is no overlap and the two events must occur separately, the events would be called **mutually exclusive**. Events are rarely mutually exclusive—unless a situation is defined that specifically prevents them from happening at the same time. For instance, consider two possible events: a person could attend a theater performance or go to a concert. These two might appear to be mutually exclusive, but more information is needed. Does the problem call for the two events happening at the exact same time, in remotely different locations, with no inclusion of video streaming, and require the person to attend the full performance of both events? It is these often-unstated assumptions that tend to make probability questions challenging. See (Mlodinow 1998) for some curious examples of interpreting conditions and applying probability to everyday life.

#### *The Addition Rule: Union of Events*

The first rule can be observed from the Venn diagram. The probability **addition rule** states:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

In words, the probability of A union B is equal to the sum of the probability of A plus the probability of B minus the probability of A and B both occurring. A union B is often read as the probability of A or B. In the Venn diagram, the probability of A is represented by the area of the entire A circle. Similarly,  $P(B)$  is the area of the B circle. The problem with simply adding the two probabilities is that the overlap portion is counted twice—once for A and once for B. Hence, the subtraction term in the formula removes the second in-

Last Name	FT Made	FT Missed	Total
Ariza	156	77	233
Brown	27	6	33
Bryant	657	104	761
Bynum	180	78	258
Farmar	59	37	96
Fisher	141	25	166
Gasol	443	136	579
Odom	199	124	323
Powell	44	12	56
Radmanovic	23	4	27
Vujacic	75	7	82
Walton	52	23	75
Total	2056	633	2689

**Figure 4.3**

Contingency table for 2009 LA Lakers free throws by players with more than 30 games. The cells contain the count of the free throws made and missed for each player. Margin totals are displayed for the rows and columns.

stance. If A and B are mutually exclusive, they cannot both occur together and there is no overlap, so  $P(A \cap B)$  is zero and the formula reduces to  $P(A) + P(B)$ . To be safe, it is best to remember the entire formula with the subtraction term.

### *The Multiplication Rule: Joint Events*

The second rule also requires an understanding of events. It deals with the issue of two events occurring together:  $P(A \cap B)$ , the intersection of A and B which makes them **joint events**. The description of joint events is one of the more complicated aspects of probability and a critical element in much of data mining and business intelligence. This section deals with the simplest version of joint events. The following sections explore some of the complications and their implications for probabilities.

Two events are **independent** if the occurrence of one event does not affect the other. For example, with a balanced coin, the probability of tossing heads does not affect the probability of tossing heads a second time. The two tosses are independent. In business terms, the probability of a customer in New York buying a specific product is generally unrelated to the probability of a different customer in Los Angeles buying the same product. As long as the customers do not know each other, share Web comments, and so on. The simple **multiplication rule** states:

If A and B are independent,  $P(A \cap B) = P(A) * P(B)$ .

When events are independent, the joint probability is obtained by multiplying the two independent probabilities. Simple examples are easiest to see with games. Say event A is rolling a fair die and obtaining a 6. Call event B rolling the die a second time and obtaining a 6. The two events are independent—

Last Name	FT Made	FT Missed	Total
Ariza	0.058	0.029	0.087
Brown	0.010	0.002	0.012
Bryant	0.244	0.039	0.283
Bynum	0.067	0.029	0.096
Farmer	0.022	0.014	0.036
Fisher	0.052	0.009	0.062
Gasol	0.165	0.051	0.215
Odom	0.074	0.046	0.120
Powell	0.016	0.004	0.021
Radmanovic	0.009	0.001	0.010
Vujacic	0.028	0.003	0.030
Walton	0.019	0.009	0.028
Total	0.765	0.235	1.000

**Figure 4.4**

Contingency table written as probabilities. Each cell value was divided by the grand total.

whatever happens on the first roll has no effect on the second. A standard die has six sides and the probability of any one number appearing is  $1/6$ . The joint probability of obtaining a six on the first roll and a six on the second roll is  $(1/6)(1/6) = 1/36$ .

As a business example, two machines (A and B) produce parts and occasionally create defects.  $P(\text{defect from A})$  is  $1/200$ .  $P(\text{defect from B})$  is  $1/300$ . Assume the machines and the defect rates are independent; for example, they are not both run by the same person. The probability of finding a defect from both A and B is  $1/200 * 1/300 = 1/60,000$ .

### *A Scary Example*

Try a scary example. A complex machine, perhaps a jet or a space shuttle, has 50,000 components. Each part has a  $1/10,000$  chance of failing so the probability of success is  $1 - 1/10,000$  or  $0.9999$ . In business shorthand, this success level is sometimes referred to as *four nines*, and it is a relatively high success rate. Assume the chance of failure in each component is independent but if one part fails the entire system fails. The probability of system success is the joint probability that each part is successful:  $P(S_1 \cap S_2 \cap \dots \cap S_{50000})$ . From independence, these probabilities can be multiplied:  $P(S_1) * P(S_2) * \dots * P(S_{50000})$ . Each individual probability is high, but the joint probability of success is  $0.9999^{50000}$  or  $0.0067$ . Less than a one percent chance the system will not fail!

How can a complex machine possibly succeed? The answer lies with the same analysis: redundancy. Build the system so that each component has a backup that is independent of the original. For convenience, assume components and backups have the same failure rates ( $1/10,000$ ). A component fails only if both the original

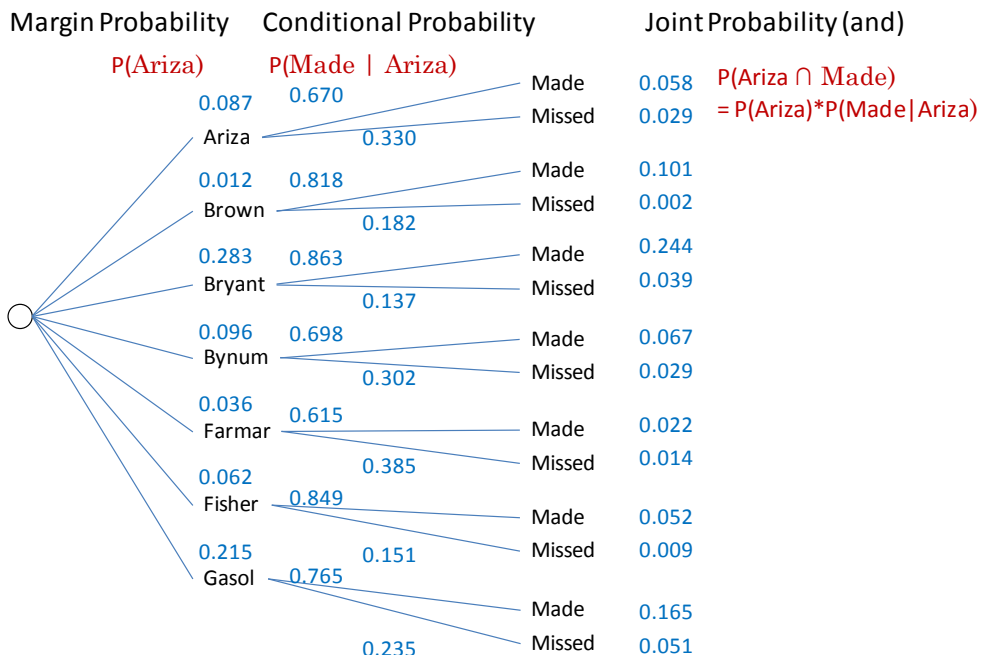
and its backup fail. So  $P(\text{failure1}) = P(F1a \text{ and } F1b) = P(F1a)P(F1b) = 1/10,000 \wedge 2 = 10^{-8}$ . So the probability of success for the entire system is  $P(\text{Success}) = (1 - 10^{-8})^{50,000} = 0.9995$ . This success probability is much higher than the original—but it carries a huge cost because everything is doubled: size, cost, weight, and so on.

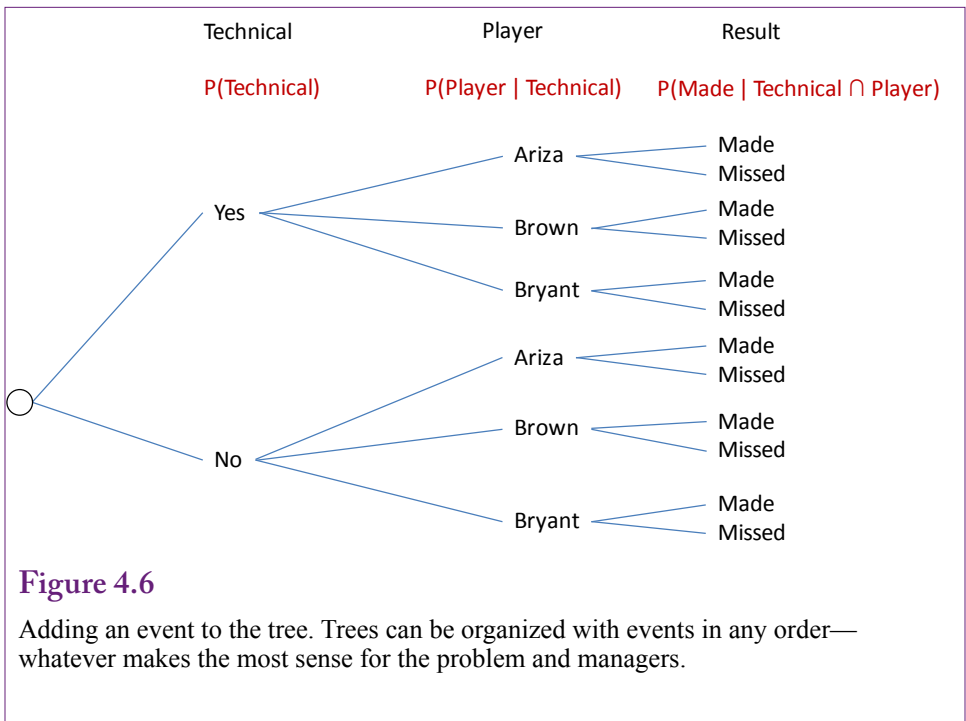
## Interdependencies: Joint Probabilities

**How are probabilities handled when events are not independent?** Independent events are important, but business problems usually involve events that are related. Most business intelligence applications are designed to find and evaluate these interdependencies. Several aspects of probability were specifically designed to deal with joint probabilities and lay the foundation for evaluating relationships among attributes and events. This section reviews the basic concepts and defines the important terms. A detailed discussion and proofs can be found in most introductory probability books. This section relies on the use of discrete data and relatively small problems. All of the concepts and results can be applied to continuous data and more complex problems.

**Figure 4.5**

Partial tree diagram of the Lakers' free throws. The nodes represent values of the attributes. Here, players and whether free throws were made or missed. The numbers are essentially conditional probabilities, except for the values on the leaf nodes which are the joint probabilities. Reading from left to right, Ariza took 8.7 percent of the free throws. When he was shooting, he made 67 percent for a joint probability of  $0.087 * 0.670 = 0.058$ .





## Contingency Tables

The easiest way to understand the concepts and terms involved with interdependencies is to examine the smallest problems that involve only two attributes or events with discrete data. This type of problem is easily displayed in a **contingency table** that places one attribute as rows and the second as columns. Typically the tables contain counts of the number of observations that fall into each cell, but sometimes the tables will display the percentages.

Figure 4.3 provides a sample contingency table using the basketball database. To keep the table relatively small, the data shows the players from the 2009 Los Angeles Lakers who played more than 30 games that season. (The 2009 Lakers were the world champions.) The columns list the total free throws made and missed by each player for the entire season—including the playoff games. If you are not interested in sports, consider the problem an issue in management. A similar table could be made for any company that keeps statistics on successes for its employees, such as jobs completed on time or number of closed sales.

Each cell in the contingency table represents a **joint probability** where the specific row and column event occur. When the cells contain counts, the probability is found by dividing by the grand total. In the example, the joint probability that Player=Ariza and a free throw is made is computed as  $156/2689 = 0.058$ . Be careful, this value is not the same as Ariza's free throw percentage or success rate. It is the probability that a player on the floor taking free throws is Ariza and that the free throw is made. The figure also shows the **margin totals** for the rows and columns. The margin totals are used to compute the percentages or probabilities within the row or column and to show how the specific row or column relates to the overall total. Figure 4.4 shows the same contingency table with the cell values and margin totals computed as probabilities by dividing by the grand total.

Notice that the row totals show the percentage of times that each player takes a free throw, not the percentage made. The column totals show the overall team's success rate. The huge value of 76.5 probably helps explain why the team won the championship—and that is a question that could be examined with later data mining tools.

A common value that is useful in making decisions is the probability of making the free throw for a specific person. For example, the coach (manager) wants to know the probability that a free throw will be made if the player is Bryant. In probability terms, this value is written  $P(\text{FT Made} \mid \text{Player}=\text{Bryant})$ . This expression is read: The probability the free throw will be made *given* the player is Bryant. It is known as a **conditional probability**, because it is conditioned on the information provided ( $\text{Player}=\text{Bryant}$ ). In general terms, the conditional probability is found by:

$$P(A \mid B) = P(A \cap B) / P(B)$$

In the example, the probability the free throw is made given Bryant is at the line is found as  $P(\text{FT Made} \mid \text{Bryant})/P(\text{Bryant}) = 657 / 761 = 0.863$ . These values are taken from the original table of counts which is closest to the way most people read the sports table. But the same result can be found using the Figure 4.4 probability table:  $0.244/0.283 = 0.863$ . The definition of conditional probability is important and you need to remember it. The trick is to remember that you divide by the probability of the item in the condition (B).

The same formula can be rewritten as the **general multiplication rule**:

$$P(A \cap B) = P(A \mid B) * P(B)$$

This rule can always be used to find the joint probability of two events. If events A and B are independent,  $P(A \mid B)$  simply reduces to  $P(A)$ . Independence means that the probability of A stays the same, regardless of whether event B occurred.

## Tree Diagrams

Contingency tables are useful because they show all of the relevant details, but they get unwieldy when the problem gets large—with too many rows and columns. Also, they can show the values of only two attributes. Tree diagrams are another way to display relationships in dependent events. They take up space on the page, but tree viewer software typically enables analysts to zoom in and out to examine the nodes.

Figure 4.5 shows the tree for the Lakers 2009 free throw data for the first seven players. The nodes represent the values of the attributes (Player and Result). The probabilities on the lines or on the specific node are the conditional probabilities. The tree is read from left to right, so any probability to the right is conditional on everything that came before. In the example,  $P(\text{Ariza})=0.087$  from the starting point, which corresponds to the percentage of times that Ariza was shooting the free throw. For the next step,  $P(\text{Made} \mid \text{Ariza}) = 0.670$ ; or  $P(\text{Missed} \mid \text{Ariza}) = 0.330$ . The final probability is the joint probability, and it is found by multiplying every probability gathered along the way from the node:  $P(\text{Ariza} \cap \text{Made}) = P(\text{Ariza}) * P(\text{Made} \mid \text{Ariza}) = 0.087 * 0.670 = 0.058$ . The joint probabilities match those in the contingency table cells.

The tree diagram can be extended to handle more events. Each event or attribute would create a new level, where every value of the attribute gets repeated for each existing node. In the example, additional data might be found to indicate



	FT Percent	Frequency Fouled	
LastName	P(FT Made   Player)	P(Player)	multiply
Ariza	0.670	0.087	0.058
Brown	0.818	0.012	0.010
Bryant	0.863	0.283	0.244
Bynum	0.698	0.096	0.067
Farmar	0.615	0.036	0.022
Fisher	0.849	0.062	0.052
Gasol	0.765	0.215	0.165
Odom	0.616	0.120	0.074
Powell	0.786	0.021	0.016
Radmanovic	0.852	0.010	0.009
Vujacic	0.915	0.030	0.028
Walton	0.693	0.028	0.019
			0.765

**Figure 4.7**

Basketball data in more common format, providing limited data. The denominator in Bayes' theorem requires multiplying the two columns and computing the sum.

whether the free throw was due to a technical foul or not. This addition would double the number of end nodes. The conditional probabilities are added to the new connectors or nodes and the final joint probability is the multiplication of every value along the tree to that specific node. Figure 4.6 shows one way to extend the tree in the basketball example. Events can be organized in any order to make the tree readable by managers. In most cases, if a technical foul is called, the coach can choose the player to shoot the free throw, so it makes sense to perform the split on technical first. Learn to read decision trees; they are heavily used in Microsoft's Analysis Services.

## Bayes Theorem

Thomas Bayes, a British mathematician and Presbyterian minister in the early 1700s, derived an interesting equation using the probability definitions. The proof is easy, and even the theorem is straightforward. The ultimate implications have a profound impact on statistics and on data mining.

For the simple explanation, begin with the definition of joint probability:

$$P(A \cap B) = P(A|B)P(B)$$

But, A and B are any two events, so the two terms easily could be reversed:

$$P(B \cap A) = P(B|A)P(A)$$

Put the two definitions together by setting the two right-hand sides equal:

$$P(A|B)P(B) = P(B|A)P(A)$$

Rearrange slightly to get Bayes' Theorem:

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}$$

It looks straightforward. The key lies in understanding how it can be used.

### *Simple Example of Bayes*

In the simplest cases, Bayes' theorem is not needed. Consider the basketball example and look at Figure 4.4 again. Initially, the contingency table was used to find the conditional probability  $P(\text{FT Made} | \text{Bryant})$ , or the probability that a free throw would be made if it is taken by Kobe Bryant. Consider a slightly different version of the question. You are watching a Lakers basketball game on TV and you notice that someone makes a free throw, but you cannot identify the person before the coverage cuts to a commercial. What is the probability the player is Bryant? Or, in a management context, you encounter a new client who praises a salesperson but does not remember the name. With the same data defined as completed sales, what is the probability the salesperson was Bryant?

If the complete contingency table or decision tree is available, it is possible to find this new conditional probability directly. In the table, find  $P(\text{Bryant} \cap \text{FT Made}) = 0.244$ . Find the margin probability  $P(\text{FT Made}) = 0.765$  and divide to get:  $P(\text{Bryant} | \text{FT Made}) = 0.244 / 0.765 = 0.320$ .

Even using Bayes' formula, if all of the data are available, the computation is straightforward. The formula requires  $P(\text{FT Made} | \text{Bryant})$ ,  $P(\text{Bryant})$ , and the total  $P(\text{FT Made})$ . By the formula, the computation is:  $0.863 * 0.283 / 0.765 = 0.320$ , which should be the same as the direct calculation, except for some rounding errors

### *Common Application of Bayes*

The real strength of Bayes' theorem is that many problems do not provide complete data. Some data might be too expensive to obtain, or timing problems might make it difficult to obtain complete data. Figure 4.7 shows a more common version of the basketball data. The first column contains the free-throw percentage made by each player. In probability notation, it is  $P(\text{FT Made} | \text{player})$ . The second column is often harder to obtain, but rough values tend to exist. It shows the relative frequency of each player getting fouled—which means taking free throws. In probability terms it is  $P(\text{player})$ . With this limited data, Bayes' theorem becomes useful. Still considering the Bryant question, the numerator is simply  $P(\text{FT Made} | \text{Bryant}) * P(\text{Bryant})$  which is found by multiplying the two columns. The denominator  $P(\text{FT Made})$  or total probability seems to be harder to find. However, it can be rewritten as

$$P(\text{FT Made}) = \sum_i P(\text{FT Made} | \text{player} = i)P(\text{player} = i)$$

That is, the denominator is the sum of the probabilities for all of the players. As shown in the table, simply compute the multiple for each player row and add them up. To obtain the probability Bryant made that last free throw, take the value for Bryant and divide by the total to get:  $0.244 / 0.765 = 0.320$ . This table is easy to extend to find the probability that any of the other players made the shot, but the highest probability lies with it being Bryant.

To obtain 10 heads in 10 tosses: H H H H H H H H H H  
 Only one way to get it and  $P(H)=1/2$ , so  $P(10H)= (1/2)^{10}$

To obtain 9 heads: H H H H H H H H H T  
 $P(H) = 1/2$  and  $P(T) = 1/2$ , so start with  $(1/2)^9 (1/2)$   
 But the tail could appear in one of 10 locations, so multiply by  $C(10, 1)$

For 8 heads: H H H H H H H H T T  
 Start with  $(1/2)^8 (1/2)^2$  and multiply by  $C(10,2)$ , the combination of ways of to get the two tails.

For 7 heads:  $(1/2)^7 (1/2)^3 C(10,3)$

Total these four values to get  $P(7 \text{ or more heads} \mid \text{fair}) = 0.171875$

### Figure 4.8

Counting heads. Computing the probability of obtaining 7 or more heads in 10 tosses of a coin is a good exercise in counting. The tosses are independent so the probabilities can be multiplied. The trick is to understand that the tail can appear in different tosses, so the values need to include the combination term.

This summation approach is a common aspect of problems that require Bayes' theorem. The point of the theorem is that data is often provided in pieces, but the formula makes it possible to determine other information based on those pieces. To look at the approach one more time, examine the decision tree in Figure 4.5 again. It clearly shows the probabilities and conditional probabilities. Multiply each line from the root to the ending leaf node. The resulting value is the joint probability. Pick the value for the chosen player (e.g., Bryant) to use as the numerator. Complete the multiplications for every node labeled FT Made, and add up the values. The resulting sum is the overall total used for the denominator.

#### *Information Value of Bayes' Theorem*

Bayes' Theorem is often used in data mining applications. The usage derives from using the theorem to look at probabilities in a different light. A slight rearrangement of the theorem yields:

$$P(A \mid B) = P(A) \frac{P(B \mid A)}{P(B)}$$

This rearrangement leads to a completely different interpretation of probability: Probability is a subjective belief that can change over time as new information is gained. In the formula,  $P(A)$  is the a priori belief in some event A.  $P(A|B)$  is the updated posterior probability—the new belief about A given that information B has been provided. The posterior belief is computed from the prior belief by multiplying by the normalized information term  $P(B|A) / P(B)$ . Typically, decision makers begin with a simple naïve belief about the probability—as neutral as possible, and the addition of information is used to successively create better estimates.

In general, business managers do not need to worry about the mechanics of how the prior belief is updated to become the new probability. However, it is worth examining a simple problem just to gain some understanding of the process. A friend shows you a coin and claims that it is fair (50 percent chance of heads and tails). You mostly trust your friend, but have some skepticism (say 10 percent). You flip the coin 10 times and 7 heads appear. How should you adjust your belief that the coin is fair?

By Bayesian updating, the goal is to find:

$$P(\text{Fair} \mid 7 \text{ or more heads}) = P(\text{Fair}) * P(7+\text{heads} \mid \text{Fair}) / P(7+\text{heads}), \text{ where} \\ P(7+\text{heads}) = P(7+\text{heads} \mid \text{Fair}) * P(\text{Fair}) + P(7+\text{heads} \mid \text{not Fair}) * P(\text{not Fair})$$

Begin with the a priori value of  $P(\text{Fair}) = 0.9$ . Most of the other values can be computed, but there is no absolute way of knowing how biased the coin might be. It appears not to be completely biased (always head), so just pick a value and guess that  $P(7 \text{ or more heads} \mid \text{not Fair})$  is 90 percent. The other value to calculate is the probability of obtaining 7 or more heads with a fair coin. Why “7 or more” instead of just 7? Because the coin could be unfair if it shows the 7 heads you did receive or any amount more. Computing this probability is a good exercise in counting.

Start with the probability of obtaining 10 heads (in 10 tosses) and work down to 7. Figure 4.8 shows the basic process and why it is a good exercise in counting. The case of 10 heads is easy because there is only one way to obtain all heads. Each toss is independent, so the probabilities can be multiplied. With a fair coin, the probability of 10 heads is  $(1/2)^{10}$ , or 0.000977, which is a small number. The case of 9 heads and 1 tail shows the counting issue. The initial probability part is straightforward. With 9 heads and 1 tail, start with  $(1/2)^9 * (1/2)^1$  to get the base, but the tail can appear in any one of the 10 tosses, so this base must be multiplied by 10 which is the number of ways of arranging the one tail in the 10 trials. Essentially, the additional rule for mutually exclusive events is applied ten times to the same number.

Follow the same process for 8 and 7 heads. For 7 heads the probability is  $(1/2)^7 * (1/2)^3 * C(10,3) = 0.1171875$ . Technically, the arrangements of tails and heads are not combinations. With only two possibilities for the ten trials (H or T), the 10 values consist of 7 items the same (H) and 3 items the same (T). The number of ways to arrange 10 items is  $10!$ , but because 7 items are the same, this number needs to be divided by  $7!$ . Because the other 3 ( $10-7$ ) are the same, this value also needs to be divided by  $3!$ , which results in:  $10!/(7!3!)$  which is equivalent to  $C(10,3)$ .

Add up the values to obtain the probability of getting 7 or more heads of 0.171875. Plug this value into Bayes’ formula:

$$P(\text{Fair} \mid 7+\text{heads}) = 0.9 * 0.171875 / (0.171875*0.9 + 0.9*0.1) = 0.632184$$

Remember that you started with a 90 percent belief that the coin was fair. After obtaining so many heads, your adjusted belief that the coin is fair has dropped to 63 percent. Information obtained from testing has improved the probability belief. This same approach is used for business problems. The example could be restated as a management question—such as whether you believe a machine is operating correctly, an employee is performing acceptably, a customer will purchase an item, and so on. Business problems quickly become more complicated, but the underlying adjustment process is the same.

## Probability Distributions

**How can probability concepts be generalized to make it easier to handle new situations?** The fundamental aspects of probability are useful, but it would be time consuming to go back to the basic rules for every problem encountered. It turns out that many problems fall into certain categories, defined by specific assumptions and characteristics. Statisticians have developed tools to handle these specific cases. Once a new problem is categorized, it is straightforward to apply the tools and understand most of the details about the new problem. Probability distributions are one of the key statistical tools. Several standard distributions are used in many applications. In the early days, tables of these distributions were published and used to solve problems. Today, computerized functions quickly provide values for many probability functions. For convenience, the standard statistical functions are programmed into spreadsheets such as Excel. Also, distribution functions are the only way to examine probabilities with continuous data. However, discrete cases are simpler so they are examined first.

### Discrete Data

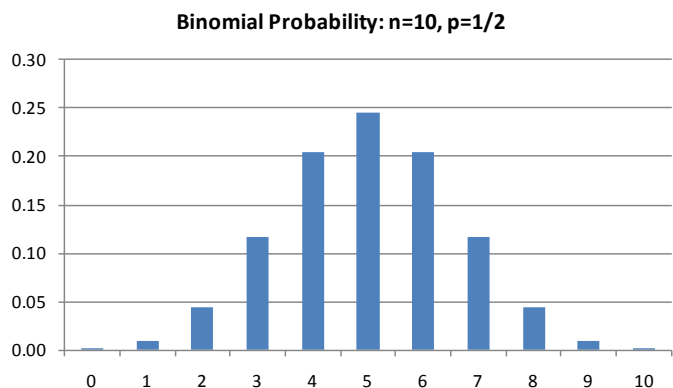
Recall that discrete data consists of values that can be counted. Simple examples include binary cases with yes/no or success/fail as the only options. Broader cases include categorical data such as gender, product category, country or state, and so on. Discrete cases also include classic games problems such as the number of heads appearing in 1,000 tosses of a coin. Discrete data can contain many values, even an infinite number, but the values are all distinct (and countable).

Return to the example from the prior section in Figure 4.8 involving the number of heads appearing in 10 flips of a coin. This type of problem is a classic example of one of the common probability distributions. Probability is defined in terms of an **experiment**, which consists of a set of events or trials and a sample space. A sample space is a set of all possible outcomes from an experiment. Ex-

**Figure 4.9**

Binomial distribution example.  $P=1/2$ ,  $n=10$ .

Number of Heads	Probability
0	0.00098
1	0.00977
2	0.04395
3	0.11719
4	0.20508
5	0.24609
6	0.20508
7	0.11719
8	0.04395
9	0.00977
10	0.00098



periments might be clinical—where the investigator has complete control over the experiment. Or, particularly in social and business environments, experiments might consist of solely of observations on real-world activities. For instance, over a two-month interval a company collects data on customers and sales totals. The investigator might have no control over the experiment, or perhaps variables such as price or advertising were altered to see what changes occurred. In any case, it is possible to define all of the possible outcomes of the experiment.

A **random variable** is an assignment of a number to every possible outcome in the sample space. For example, if the experiment is to flip a coin ten times, a random variable could be the total number of heads that appear. In other experiments, a random variable could be defined as the weight of a group of workers, or the total sales made to a customer. Note that before the experiment is conducted or evaluated, there is no way of knowing the exact value assigned to the random variable—hence the name “random.” Discrete random variables take on a limited (countable) number of values. A discrete random variable  $X$  can be assigned any value  $x_i$  from a set of values. Each possible outcome can be assigned a number called a probability  $p(x_i) = P(X=x_i)$  that must meet the basic conditions:

$$p(x_i) \geq 0 \quad \text{for all } i \text{ items in the set}$$

$$\sum_{i=1}^{\infty} p(x_i) = 1$$

The function  $p$  is the **probability function** and the pairs  $(x_i, p(x_i))$  define the **probability distribution** of  $X$ . More recently, the term **probability mass function** has become popular instead of just probability function.

### *Binomial Distribution*

The concept of distributions is easiest to see with an example. Consider the coin tossing example. A fair coin has a 1/2 probability of displaying a head on any individual toss. Throw the coin 10 times and count the number of heads in that experiment. The outcomes range from 0 to 10. Figure 4.9 shows the probability distribution along with a chart of the probabilities. The probabilities at each value are computed using the basic probability rules. The example of 7 heads and 3 tails was outlined in the prior section. Each trial flip is independent of the others, so flipping a total of 7 heads means the probability of one head AND another head and so on up to seven. Because the events are independent,  $P(H1 \cap H2)$  is  $P(H1)*P(H2)$  for seven times. Generically, the probability of obtaining one success (head) is defined as  $p$ , which is 1/2 in this example, but will be different for other problems. In general, the probability of obtaining  $k$  successes  $P(X=k) = p^k$ . But, with  $n$  trials,  $k$  successes appear only if  $k-n$  failures appear, so the formula needs the probability of obtaining exactly  $k$  success and  $n-k$  failures:  $p^k p^{n-k}$ . The other catch is that it does not matter when the successes or heads appear. The experiment is only interested in the totals. The formula to this point defines one way of obtaining the specified number of successes. The base probability must be multiplied by the number of ways of arranging the outcomes. Because only two different outcomes exist, the ways of arranging them are  $n! / n! (n-k)!$ , where

k	p(k)	cdf(k)
0	0.006738	0.006738
1	0.033690	0.040428
2	0.084224	0.124652
3	0.140374	0.265026
4	0.175467	0.440493
5	0.175467	0.615961
6	0.146223	0.762183
7	0.104445	0.866628
8	0.065278	0.931906
9	0.036266	0.968172
10	0.018133	0.986305

**Figure 4.10**

Poisson distribution example. Mean = 5. The probability of seeing 10 or more arrivals in the time interval is 1-0.968 or about 3 percent.

the divisors handle the duplicates of the two outcomes. The result is the binomial distribution:

$$\text{binomial}(k, p, n) : P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

k: Number of observed successes.

p: Probability of observing success on a single trial.

n: Total number of trials.

Excel: BinomDist(k, n, p, false)

The binomial distribution is used for any experiment that (1) has a binary result (success or failure), (2) a repeated number of trials, and (3) a fixed probability of success on each trial. The third requirement applies to experiments where the items are drawn with replacement. For problems where items are drawn from a finite set and not replaced, the probability changes with each draw and it is necessary to use the hypergeometric distribution. However, it is rarely used in data mining so it is not covered here.

What if a problem has more than two outcomes? The multinomial distribution handles cases with multiple outcomes. This probability function generalizes the binomial, where the outcomes are considered as categories ( $x_1, \dots, x_k$ ):

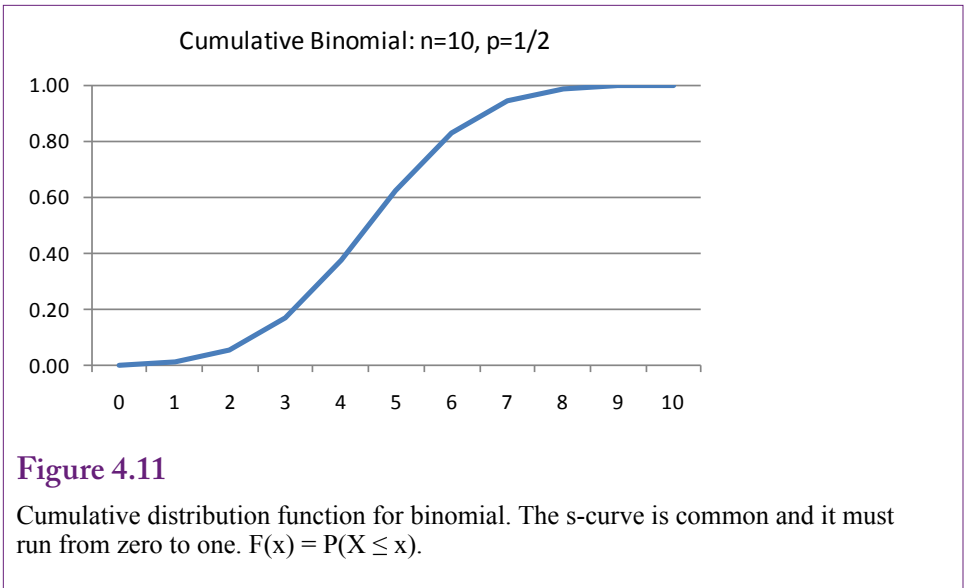
$$P(X_1 = x_1 \text{ and } \dots X_k = x_k) = \frac{n!}{x_1! \dots x_k!} p_1^{x_1} \dots p_k^{x_k}$$

k            Number of outcomes or categories

$x_1 \dots x_k$     Number of observed successes in each category

n            Total number of trials





### *Poisson Distribution*

The **Poisson distribution** is another useful discrete distribution. It is commonly used in business and operations management problems to evaluate the number of arrivals during a fixed time period. For example, retail stores and fast food restaurants need to know how many customers might arrive during a lunch hour so that a sufficient number of checkout clerks can be hired to handle the peak load. The probability function is given by:

$$P(X = k) = \frac{e^{-\alpha} \alpha^k}{k!}$$

In the function,  $\alpha$  is a parameter that must be provided. It is the average number of arrivals expected during that time period. This value is generally estimated by counting the number of customer arrivals over time; or by pulling the data from the database using the time stamps on receipts. To illustrate, consider a restaurant that averages 5 customers per minute during the lunch hour time slot. What is the probability that the company is swamped with 10 or more customers during one minute? Figure 4.10 shows the probability and cdf values for the number of arrivals ranging from 0 through 10. There is about a 2 percent chance that exactly 10 customers will arrive within any minute. The cdf can be used to compute the probability of 10 or more customers because that value is 1 minus the probability of 9 or fewer customers. Hence, the answer is  $1 - 0.968$ , or about 3.2 percent chance of having that many customers arrive in one minute.

### *Cumulative Distributions*

An important aspect of probability distributions is that they are point values that return the probability of only that one discrete value. For instance, Binomial(7, 1/2, 10) returns the probability of seeing exactly 7 heads in 10 coin flips. However, many problems are written so that they require cumulative values. Such as what is the probability of seeing 3 or fewer heads? The problem with point values is that

<b>p</b>	0.5	0.5	0.5
<b>n success</b>	7	70	700
<b>trials</b>	10	100	1000
<b>Binomial</b>	0.1171875	2.32E-05	5.07E-38

**Figure 4.12**

Binomial probability with an increasing number of trials. Observe what happens to the point probability as the number of trials increases. With continuous data, the point probability at any point is zero.

in many cases, the probability is small for seeing exactly one specific result. This point is even more critical in the next section on continuous variables.

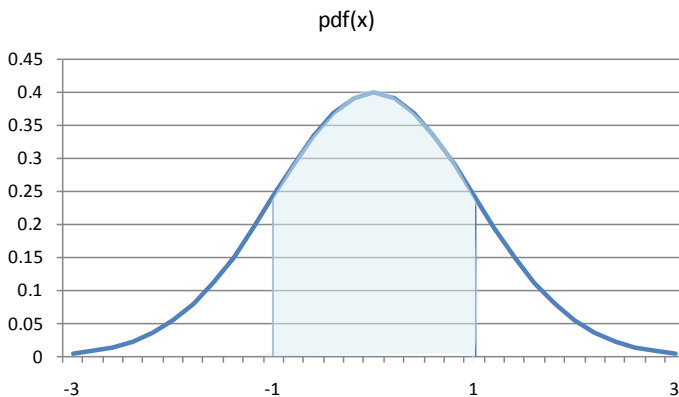
The **cumulative distribution function (cdf)**  $F(x)$  is defined as  $P(X \leq x)$ . With discrete random variables, it is the sum of all of the values from zero up to  $x$ .

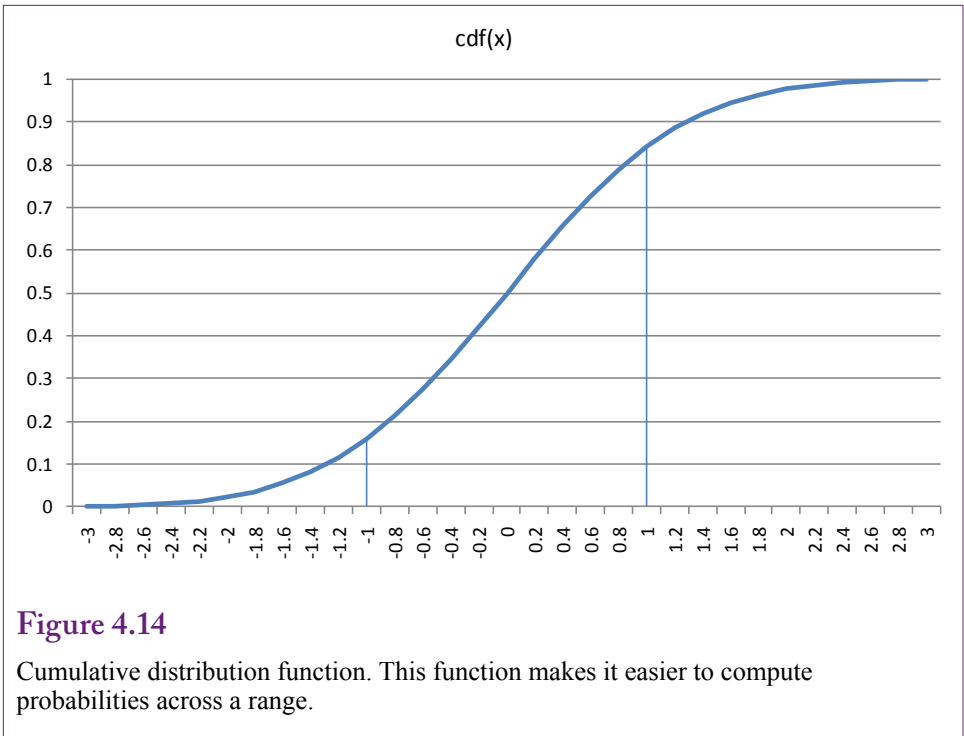
$$F(x) = \sum_{u: x_j \leq x} p(x_j)$$

Figure 4.11 shows the cdf for the simple binomial example. Changing the parameters will alter the cdf, but the basic s-curve shape is common for most distributions. The values must range from zero to one and the function is non-decreasing. Cumulative probability tables are often used to lookup specific values. Computer functions are typically available to compute cumulative distributions. For instance, the Excel function BinomDist uses the last parameter to specify that the function should return a cumulative value (true) or just the point value (false).

**Figure 4.13**

Probability density function for continuous data. Probability is computed as the integral of the pdf over the specified range—which is the area under the pdf curve. It is also the difference of the cdf values:  $F(1) - F(-1) = 0.841 - 0.159 = 0.683$





Some problems ask for the probability of  $X$  falling between two values  $P(a \leq X \leq b)$ . This value can be computed by finding the cumulative probability up to value  $b$  and subtracting the cumulative probability up to value  $a$ . Be careful with discrete distributions. In the sample binomial distribution, what is the probability that the number of heads falls between 3 and 7 (inclusive)? Because  $N$  is only 10, this value could be computed by using the simple probability distribution function (non-cumulative) and adding the results. If  $N$  is large, this total can be tedious, even with a spreadsheet. With the cumulative approach, the answer is found using:  $P(X \leq 7) - P(X \leq 2)$  or  $0.9453 - 0.0547 = 0.891$ .

### Continuous Data

With these basic concepts, it is now time to examine distributions for continuous data. The first, most important, concept to understanding continuous data is that the point probabilities are all equal to zero. Recall the binomial example for 10 trials. The probability of obtaining exactly seven heads was almost 12 percent. Check Figure 4.12 to see what happens to that point probability as the number of trials increases. With continuous data, the probability of any specific point is always zero. Think about it in terms of the number of possible values. It has to be impossible to hit exactly one specific number out of infinity.

With continuous data, probability is defined only in terms of the cumulative distribution function. Probability can be found only for ranges of values. Technically, a **probability density function (pdf)** or probability mass function exists, but it is defined only in terms of the cumulative values:

$$f(x) \geq 0 \text{ for all } x$$

$$F(x) = \int_{-\infty}^{+\infty} f(x) dx = 1$$

Integration can be interpreted as the area under the pdf curve. It is also the value of the cumulative distribution function. Figure 4.13 shows a sample pdf for a continuous distribution. The shaded area shows the method of computing the probability  $P(-1 \leq X \leq 1)$ . Given the specific form of the pdf, this probability could be computed directly. However, the pdf functions tend to be difficult to integrate, so the values are found from cumulative tables or using computer numerical formulas. In the example, the probability can be found using the cdf:  $F(1) - F(-1) = 0.683$ . The probability of  $X$  falling between  $-1$  and  $1$  is about 68 percent.

To obtain those specific values, you need to know that the pdf plotted is for the standard normal distribution. Note that because the probability at any point is zero, you do not need to worry about the end points and inequalities, which are a concern with discrete distributions. Figure 4.14 shows the cumulative distribution function for the example. The approximate cumulative probabilities can be read from the chart, but more precise values can be found in tables or with computer functions.

## Joint and Conditional Probabilities

*Note: This section shows theory and could be skipped.*

Most probability density functions are discussed and displayed in terms of a single variable. These functions are easy to plot and to tabulate values for both the pdf and cdf. However, most business problems involve multiple attributes or variables. Fortunately, the computational tools are built to handle multiple variables. Even better, the behavior of the functions is the same for multiple variables. So, once you understand the functions in terms of single variables, that knowledge can be applied to multiple variables. However, multiple variables add a few complications.

First, as shown with contingency tables and tree diagrams, multiple variables require a joint density function. In discrete terms, the joint probability function could be expressed as  $P(X=x \text{ and } Y=y) = p(x,y)$ . Continuous variables require defining probability from a joint density function that is integrated to obtain the cumulative distribution:

$$f(x, y) \geq 0 \quad \text{for all } x, y.$$

$$\iint f(x, y) dx dy = 1$$

For more variables, another integral (summation) is added for each variable. Recall that the contingency table included margin probabilities that were computed as the sum of probabilities for a given value of  $x$  or  $y$  (row and column totals). The same principle leads to marginal density functions for continuous variables, replacing summation with integrals:

$$\text{marginal pdf for } x : g(x) = \int f(x, y) dy$$

$$\text{marginal pdf for } y : h(y) = \int f(x, y) dx$$

Consequently, conditional probabilities are defined similarly to the probability case:

$$\text{conditional pdf } g(x|y) = \frac{f(x, y)}{h(y)}, \quad \text{for } h(y) > 0$$

Net Return	Probability	X*p
1000	0.05	50
700	0.7	490
-2000	0.25	-500
	Total/EV	40

**Figure 4.15**

Expected Value computation. Three outcomes exist for an investment. The net return and probability are given for each outcome. The expected value is found by multiplying each value by its associated value and adding. The expected value here is positive so the investment is worthwhile.

With these definitions, Bayes' Theorem also applies to continuous data. With continuous random variables, Bayes' Theorem leads to a method to update the entire density function not just a simple probability. Methods that use this approach begin with a neutral pdf (often a uniform distribution), evaluate the data and adjust the density function. This new pdf is plugged in as the new a priori function and the process repeats until the data is used up. The result is a pdf that incorporates all of the information from the dataset and can be used to compute the probabilities of any range of values.

### Expected Value (Mean) and Variance

Probability distributions need to be general so they can be applied to many different types of problems. Most of the distributions are defined in terms of **parameters** that fit the distribution to specific problems. For example, in a non-statistical context, a line can be defined as  $y = mx + b$ . The variables  $x$  and  $y$  represent the data, leaving  $m$  and  $b$  as parameters. The values for  $m$  and  $b$  are estimated from the data and represent the slope and intercept values. Probability functions have similar parameters that enable them to be fit to a specific problem. Two of the most common parameters are the mean and variance.

To understand probability functions and parameters, it is important to understand the concept of expected value. Given a random variable  $X$ , the **expected value** of  $X$  is defined to be

*Expected Value*

$$E[X] = \sum_{i=1}^{\infty} x_i p(x_i) \quad \text{X is discrete}$$

$$E[X] = \int_{-\infty}^{+\infty} x f(x) dx \quad \text{X is continuous}$$

Expected value is easiest to see with discrete values because most people find it easier to compute sums than integrals. Expected value is an extremely useful tool for solving basic business problems. Figure 4.15 shows a simple example of an investment that has three possible outcomes. The random variable is the net return, shown for each possible outcome. Probabilities exist for each of the outcomes—they could be subjective probabilities. Compute the expected value by multiplying each outcome value by its associated probability and summing the values.

Outcome	Net Return	probability	X*p	X-E(X)	p(X-E[x])^2
1	1000	0.05	50	1000	50000
2	700	0.70	490	700	343000
3	-2000	0.25	-500	-2000	1000000
			40		1,393,000

Outcome	Net Return	probability	X*p	X-E(X)	p(X-E[x])^2
1	150	0.2	30	100	4500
2	700	0.3	210	700	147000
3	-400	0.5	-200	-400	80000
			40		231,500

**Figure 4.16**

Variance calculation. A second investment option has the same expected value but the variance is considerably lower.

The result is a 40 gain, so this investment is acceptable. Note that 40 is not the amount you would make or lose from the investment. You will actually obtain one of the three values shown in the net return (1000, 700, or -2000). In fact, with a 70 percent probability on a single investment, the most likely outcome is that you would receive 700. However, the expected value represents the average amount you would receive if the experiment/investment were repeated a huge number of times. Most of the time (70 percent), you would make 700, but 25 percent of the time you would lose 2000. In the end, you would make 40 on average.

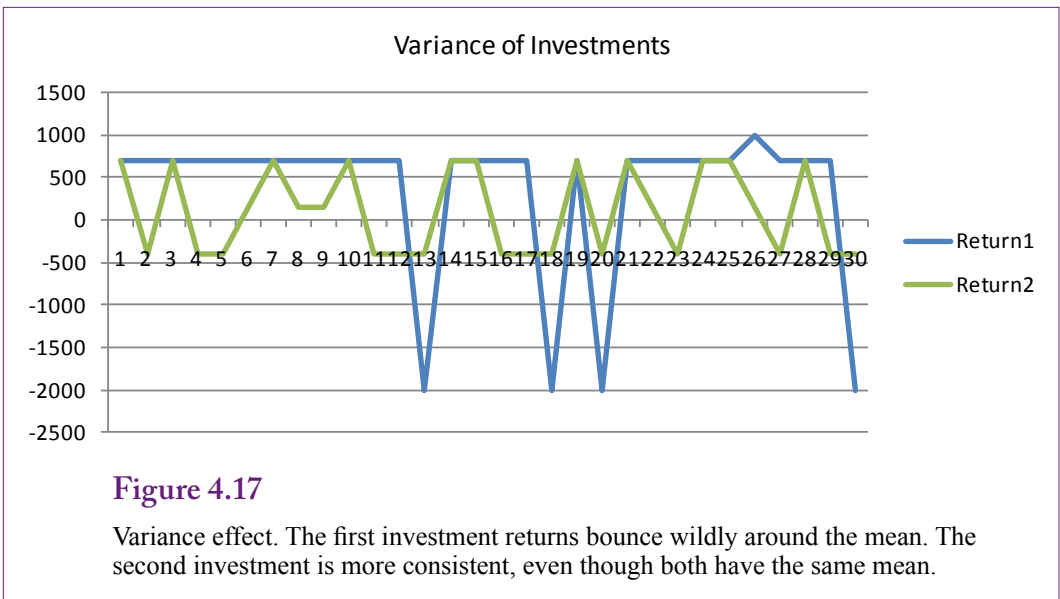
Consider one more example using a binomial distribution. The probability of success, say heads on a coin toss, is  $p$ . If success is measured as 1 and failure (tails) as zero, the expected value of one toss is  $1/2$ . If the coin is flipped 10 times, the total expected number of heads is:  $1/2 + 1/2 + 1/2 + \dots + 1/2$  for ten times which is  $10 \cdot (1/2)$  or 5. In general, the expected value of the binomial distribution is  $n \cdot p$ . If you are curious, the expected value of the Poisson distribution can be computed to be alpha ( $\alpha$ ), its single parameter.

The expected value is also known as the mean of the distribution. It is the center point. In physics terms, it is the center of mass of the distribution.

### *Variance*

Expected value is a powerful tool and relatively easy to understand. Unfortunately, too many people focus on just the mean when making decisions. After all, the mean indicates the expected outcome if the experiment is performed many times. But, there might be many ways of getting to that mean. Consider the investment example again and assume it is an investment you can make repeatedly. For the first few times, you are relatively lucky—after all, 75 percent of the time the investment returns positive values. Then, bang, you lose 2000. The investment is characterized by relatively large gains and losses.

Consider a second investment option that has the same expected value. Figure 4.16 shows a second investment option that has been configured to have the same expected value (40) as the first option. Compare the probabilities of the two



**Figure 4.17**

Variance effect. The first investment returns bounce wildly around the mean. The second investment is more consistent, even though both have the same mean.

investments. They are roughly the same in each of the three outcomes for the second investment. They are widely different for the initial investment. Look at the net returns for the three outcomes. Again, in the second case the net returns are closer together in all three outcomes.

This concept is known as **variance** and it has the statistical definition:

$$V(X) = E[X - E(X)]^2$$

That is, the variance is the expected value of  $X$  minus the mean, squared. Figure 4.16 shows the direct way to calculate the variance. Subtract the mean from each value, square that difference and multiply by the probability to get the expected value. Add up the values to get the variance. The base formula is useful for understanding that variance measures the deviation from the mean, but an easier calculation method can be found by modifying the formula to:

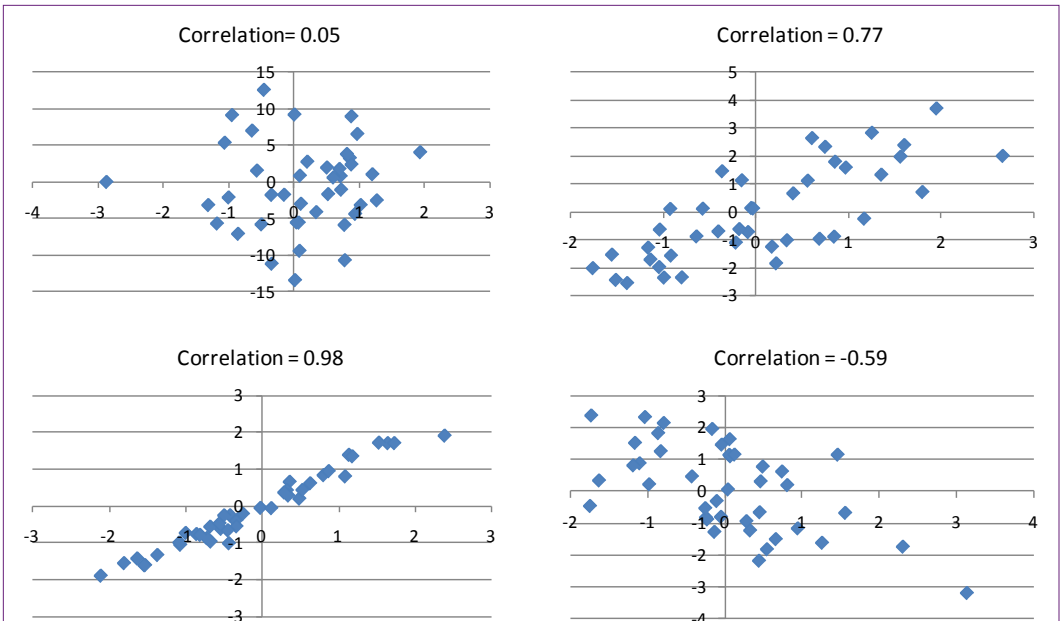
$$V(X) = E(X^2) - [E(X)]^2$$

With this version, the mean  $E(X)$  and the term  $E(X^2)$  are computed in one pass through the distribution. Then square the mean and subtract it from the first term.

The point is that when  $X$  values have large deviation from the mean or center, the distribution has a higher variance. Figure 4.17 shows the effect using the two investment options. The first investment has large swings or deviations about the mean. The second investment is more consistent. Both distributions have the same mean. Looking only at the mean, it might be tempting to conclude the two investments are equally valuable. Incorporating the variance provides more information. Most investors would choose investment two that has less variance.

Look again at the original definition for variance. It uses  $(X - \text{mean})$  squared. The variance is measured in squared units of the random variable. For instance, if the data is dollars, the mean is measured in dollars, but the variance is measured in squared-dollars. Consequently, a more useful definition is to define **standard deviation** as the square root of the variance. The standard deviation is in the same units as the original data, yet still represents variation from the mean.





**Figure 4.18**

Sample correlation coefficients. At zero, the scatter plot is essentially random. As the correlation approaches one, the relationship is closer to a straight line. Negative values exhibit the same relationship, but the slope of the line is negative.

The mean and standard deviation appear in many distributions and many statistical situations. For the distributions, the mean is usually denoted as  $\mu$  and the standard deviation as  $\sigma$ . Variance is the square of the standard deviation or  $\sigma^2$ .

### *Correlation Coefficient*

When a distribution has more than one random variable (dimension), then the possibility exists for correlation between two variables. These correlations are critically important to data mining. The business analyst wants to find correlation patterns to see which variables move together. If two variables are independent, the correlation will be zero. If positive changes in one variable (say X) are matched by positive changes in a second variable (Y), the two are said to be positively correlated. With negative correlation, the variables move in opposite directions. These concepts are defined statistically with the correlation coefficient:

$$\rho = \frac{E\{(X - E[X])(Y - E[Y])\}}{\sqrt{V(X)V(Y)}}$$

For computational purposes, the correlation coefficient can be rewritten:

$$\rho = \frac{E(XY) - E(X)E(Y)}{\sqrt{V(X)V(Y)}}$$

Distribution	Function	Mean	Variance
Binomial	$\binom{n}{k} p^k (1-p)^{n-k}$	$np$	$np(1-p)$ or $npq$
Poisson	$\frac{e^{-\alpha} \alpha^k}{k!}$	$\alpha$	$\alpha$

**Figure 4.19**

Summary of mean and variance for common discrete distributions.

The correlation coefficient is similar to the definition of variance, except that it measures the deviation from the mean for  $X$  and multiplies it by  $Y$ 's deviation from the mean. The denominator uses the two variances to normalize these deviations, so the correlation coefficient varies between  $-1$  and  $1$ . Values of  $-1$  or  $1$  represent perfect correlation. If the  $X$  and  $Y$  values were plotted on a chart, they would fall on a single line when the correlation is perfect. This concept is explored in more detail in the regression chapter.

Figure 4.18 shows some sample correlation coefficients. The correlation coefficient measures the relationship between pairs of variables. Near zero, the scatter plot is essentially random—the variables are independent. As the coefficient approaches one (or negative one), the scatter plot becomes closer to a straight line. Negative values exhibit the same effect but the slope of the line is negative. Keep in mind that correlation is a statistical measure only. It does not imply causation. Higher correlation numbers simply mean the variables move together, but it does not mean that changes in one will always cause changes in the other variable. Causation has to be determined by a model that provides a theoretical explanation for the joint movement.

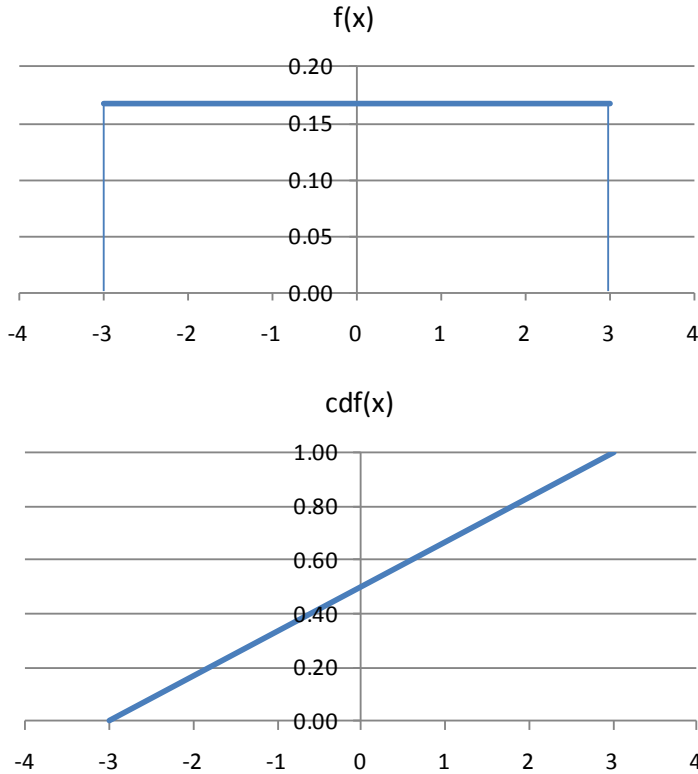
### *Discrete Distributions*

The parameters of a probability function often determine the mean and variance of the distribution. The expected value and variance can be found by applying algebra to the underlying distribution function. Sometimes the process is tedious, but the values have already been computed for the standard distributions. They are listed here without proof, because they are useful to know when working with the selected distribution. Also, they provide some insight into the distributions.

Figure 4.19 shows the mean and variance for the common discrete distributions. For the binomial distribution, the variance is often written as  $npq$ , where  $q$  is the probability of failure or  $1-p$ . In the coin-flipping example where  $p=1/2$ ,  $pq$  is equal to  $1/4$ , so the variance for tossing the coin 12 times is  $npq = 3$ . The following sections describing continuous distributions will explain the value of knowing the standard deviation.

### **Important Continuous Distributions**

Discrete distributions are useful for understanding the basic concepts and for solving specific types of problems. However, many random variables involve continuous data. More importantly, the continuous distributions are critical to solving cer-



**Figure 4.20**

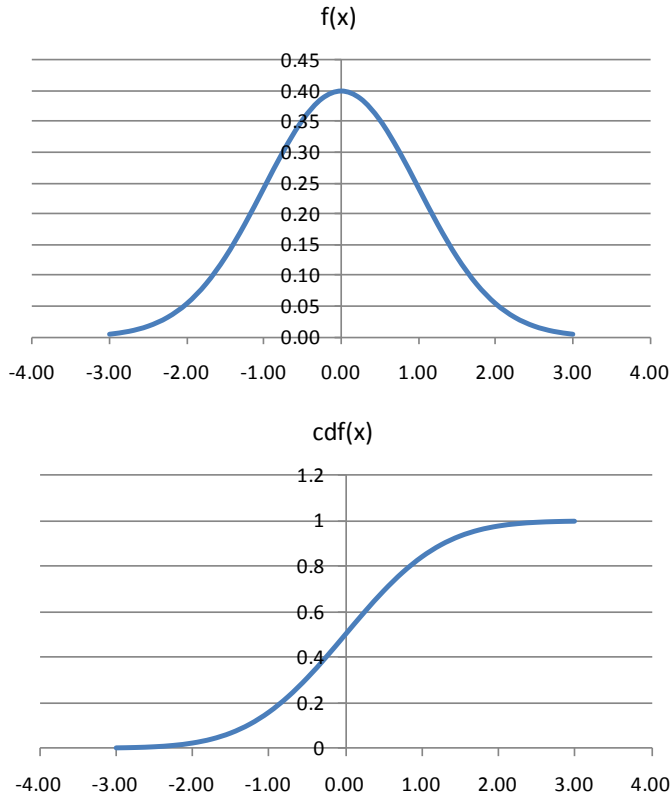
Uniform distribution. Data is evenly distributed across a fixed interval. Sometimes used as a neutral starting point for Bayesian analysis.

tain problems. In particular, the normal or Gaussian distribution is the foundation for much of statistics—partly because many of the other distributions, including binomial, converge to that distribution as the number of observations increases. This section presents the most important of the continuous distributions. The section is merely a summary. Details can be found in any statistics textbook.

Remember that probability for continuous variables depends on a specified range. The probability density function defines probability (pdf) as the area under the curve. Consequently, the cumulative distribution function (cdf) is often used to compute actual probability values.

#### *Uniform or Random Distribution*

The **uniform distribution** is useful for two reasons. First, it is easy to understand, and second, it represents a basically equally random distribution of data. It is often used as an a priori distribution in Bayesian analysis because it imposes no specific structure on the data. It does require specifying finite minimum ( $a$ ) and maximum ( $b$ ) points to limit the range. Figure 4.20 shows the pdf and cdf for the uniform distribution. The distribution has two parameters:  $a$  and  $b$ , the starting and ending points of the range. Data is evenly distributed, so fixed range has the same probability. Hence, the distribution is sometimes called random because it



**Figure 4.21**

Normal or Gaussian distribution. With a large number of observations, most problems will follow a Normal distribution. The distribution is defined by the mean and standard deviation, but all problems can be converted to the standard normal  $N(0,1)$  with a mean of zero and standard deviation of one.

imposes no emphasis on any particular values. The pdf and cdf are mathematically easy:

$$f(x) = 1/(b-a)$$

$$cdf(x) = (x-a) / (b-a)$$

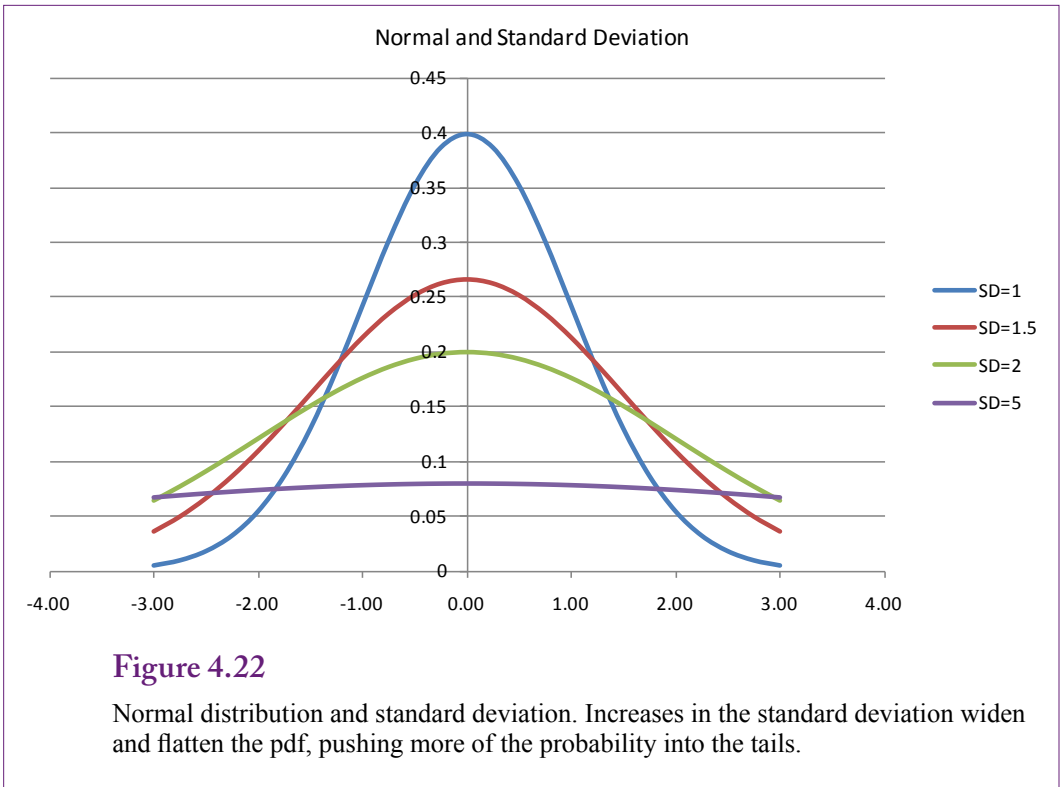
$$\text{Mean: } (a+b)/2$$

$$\text{Variance: } 1/12 (b-a)^2$$

The uniform distribution has limited use except for providing a neutral starting point for some investigations.

#### *Normal or Gaussian Distribution*

By far, the most important continuous distribution is the Normal or Gaussian distribution. One of its most important values is that most of the other distributions converge to the Normal distribution as the number of data points increases. So it is the benchmark for most statistical analyses. This section explores some of the fundamental properties of the Normal distribution, but many of the concepts apply



**Figure 4.22**

Normal distribution and standard deviation. Increases in the standard deviation widen and flatten the pdf, pushing more of the probability into the tails.

to other distributions as well. Figure 4.21 shows the pdf and cdf for the standard normal distribution. The Normal distribution is defined by the mean and standard deviation, but all problems can be converted to the standard normal which has a mean of zero and a standard deviation of one. It is commonly written  $N(0,1)$ .

The equations for the pdf and cdf are hugely important in statistics and mathematics, but they are mathematically difficult. In particular, there is no simple solution to the cdf. The integration and probability values can be found only through computer numerical analysis.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left[\frac{x-\mu}{\sigma}\right]^2\right)$$

$$cdf(x) = \frac{1}{2} + \frac{1}{2} \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)$$

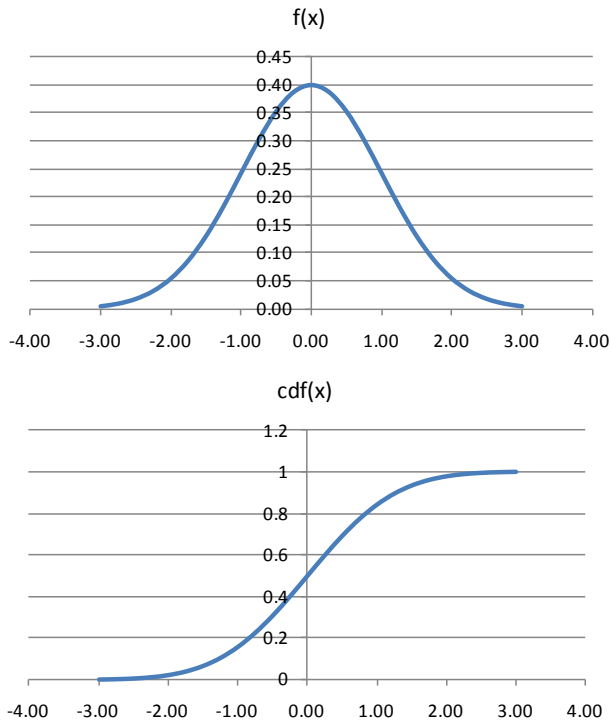
$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

Mean:  $\mu$

Variance:  $\sigma^2$

Conversion to standard normal:  $Z = (X - \mu) / \sigma$

Excel: `NormDist(X,  $\mu$ ,  $\sigma$ , cumulative)`



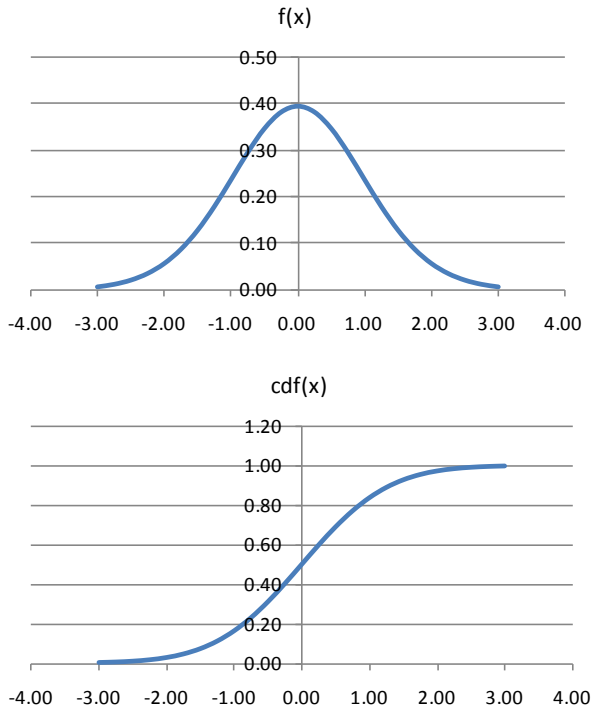
**Figure 4.23**

Percentage of data falling within one standard deviation of the mean. Area under the pdf between  $[-1, 1]$ , or  $\text{cdf}(1) - \text{cdf}(-1)$  is 0.6827. The percentage for two standard deviations is 0.9545 and for three it is 0.9973.

The Excel function returns the value of the pdf if the entry for cumulative is set to false, otherwise it returns the cdf value.

The mean establishes the center of the distribution—it is symmetric about the mean. As shown in Figure 4.22, increases in the standard deviation widen and flatten the pdf. Consequently, more of the area under the curve or the probability is pushed into the tails away from the center. If the standard deviation is very high, the distribution resembles the uniform distribution, providing little information about the underlying data.

A key aspect to understanding the Normal distribution is to recognize that when a random variable follows the Normal distribution, most of the observations are close to the mean. A smaller standard deviation means that the data are clustered even closer to the mean. To see how close, Figure 4.23 shows how to compute the percentage of data falling within one standard deviation of the mean. The value can be found quickly using the cdf, because it is simply  $\text{cdf}(1) - \text{cdf}(-1) = 0.6827$ . Think about that for a second. In a Normal distribution, over two-thirds of the data fall within one standard deviation of the mean. Check the values for two and three standard deviations: 0.9545 and 0.9973. In a large enough group, with any random variable, almost every value will fall within three standard deviations of the mean. Find a group of people, measure everyone's weight and height. Very few outliers will exist. Give an exam to 100 students and 95 percent of the values will fall



**Figure 4.24**

T Distribution with 25 degrees of freedom. It is similar to the normal distribution. Standardize the data  $T = (X - \mu) / \sigma$ . Degrees of freedom are typically  $n - 1$ .

within two standard deviations of the mean, and there is only a 0.3 percent chance of someone scoring outside of three standard deviations. These conclusions arise because of the role of standard deviation. If the standard deviation is high, a huge variation in values can still exist.

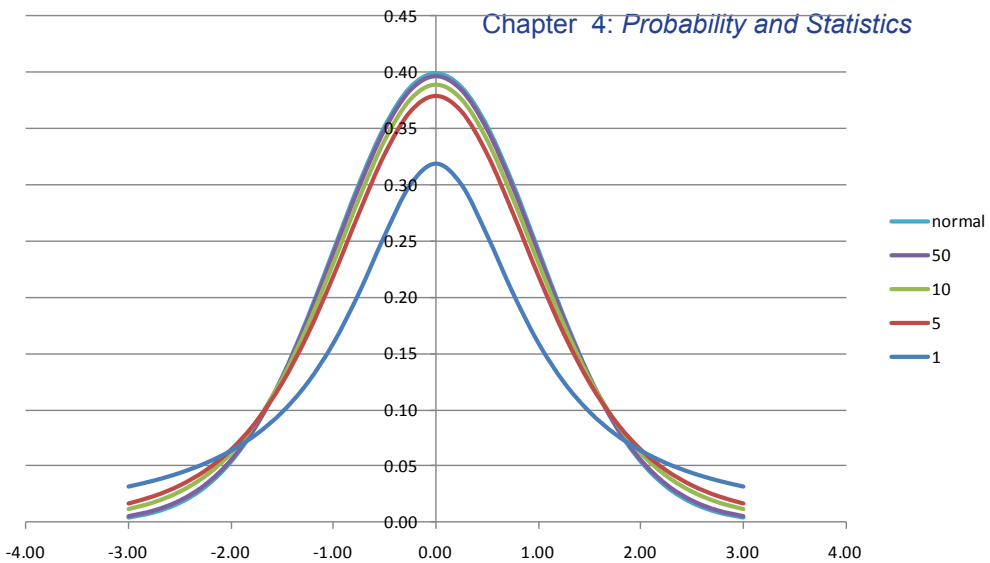
The mean is important because it shows where the distribution is centered. But the standard deviation describes how tightly the data is clustered around that mean. For example, perhaps daily sales have a mean of 10,000 and a standard deviation of 1000. If the data follow a normal distribution, it is easy to predict that tomorrow the sales total will be between 8,000 and 12,000 with a 95 percent probability. On the other hand, if the standard deviation is 4,000; the 95 percent forecast would be from 2,000 to 18,000, which is not very useful.

The multivariate normal distribution has also been studied extensively, but it is basically a generalization of the simple normal distribution.

### *T-Distribution*

What happens if there is not enough data to justify using a normal distribution? The Student's T distribution was specifically created to handle problems with small samples. It functions the same way as the Normal distribution, but has one more parameter: degrees of freedom. For most problems, the degrees of freedom are one less than the number of observations, or  $n - 1$ . Figure 4.24 shows the pdf and cdf for the T distribution with 25 degrees of freedom. It looks the same as the





**Figure 4.25**

T Distribution with different degrees of freedom. Even at  $df=50$ , the pdf is almost on top of the normal distribution. At lower degrees of freedom, or smaller samples, more of the probability is moved to the tails.

normal distribution. Just to be complete, the equation for the pdf is given here, but even that is relatively complex because of the gamma function. The cdf is even more complex. Both functions are typically computed using numerical analysis.

$$f(x) = \frac{\tilde{\Gamma}\left(\frac{d+1}{2}\right)}{\sqrt{d}\tilde{\Gamma}\left(\frac{d}{2}\right)} \left(1 + \frac{x^2}{d}\right)^{-\left(\frac{d+1}{2}\right)}$$

Mean: 0 (standardized)

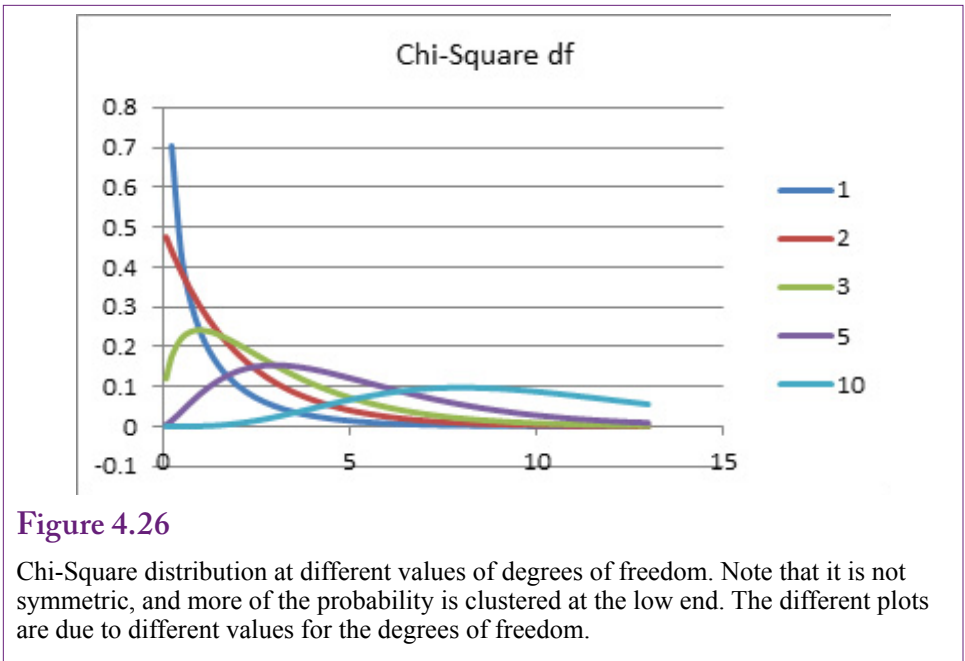
Variance:  $d/(d-2)$   $d > 2$

$d$ : degrees of freedom, usually  $n - 1$

Excel: TDIST( $X$ ,  $df$ , tails); but it returns the tail,  $P(T > z)$

The Excel formula TDIST is different from the normal distribution function. TDIST only works for positive values of  $T$  and it does not return the pdf or the cdf. Instead, it returns  $P(T > x)$ , or the probability in the right-hand tail. This value can be converted to the cdf by subtracting it from 1 to get  $P(T < x)$  for values of  $x$  greater than zero. Because of the way the T distribution is typically used, the Excel formula makes sense, but it is important to realize exactly what is being computed.

To demonstrate the effect of the degrees of freedom, Figure 4.25 plots the T distribution at several different values for the degrees of freedom. With a small number of observations, the pdf is flatter and wider—pushing more of the probability to the tails. Outliers, or extreme effects are more likely to arise. Going the other direction, even at  $df=50$ , the plot of the pdf is almost identical to the plot of the normal distribution.



The T distribution, or the normal for large problems, is heavily used in hypothesis testing—which is covered in the statistics sections of this chapter. The T distribution is more accurate for smaller problems.

### *Chi-Square Distribution*

Figure 4.26 shows the Chi-square distribution at varying degrees of freedom. Even a quick glance at the pdf indicates that this distribution is different from the others. It is defined only for positive values of  $X$  and positive values of the degrees of freedom. The distribution is not symmetric—more of the probability is concentrated near zero. The distribution is often written with the Greek letter Chi:  $\chi^2_n$ , or Chi-square with  $n$  degrees of freedom.

$$f(x) = \frac{\left(\frac{1}{2}\right)^{\frac{d}{2}}}{\Gamma\left(\frac{d}{2}\right)} x^{\frac{d}{2}-1} e^{-\frac{x}{2}}$$

Mean:  $d$

Variance:  $2d$

$d$ : degrees of freedom

Excel: `ChiDist(X, df)`  
returns right tail  $P(X > x)$

The Chi-square distribution has some specific uses that are covered in the statistics section. It can also be applied in cases that involve squared random variables. A random variable with an  $N(0,1)$  distribution, when squared will have a Chi-square distribution with one degree of freedom. In fact, if the degrees of freedom exceed about 45, and a random variable  $Y$  has a Chi-square distribution, a new variable can be defined as  $\sqrt{2Y}$ , and it will have a normal distribution  $N(\sqrt{2n-1}, 1)$ . For now, simply remember that the Chi-square distribution is useful for squared data.

### Central Limit Theorem

One of the most important results in the theory of probability and statistics is called the **central limit theorem**. The theorem states that for almost any random variable  $X$ , the average of the  $X$  values will have a standard deviation of  $\sigma/\sqrt{n}$  and the standardized  $Z$  value will have a normal distribution for sufficiently large number of observations. In other words, for almost any data the average of that data will follow a normal distribution.

$$\frac{\bar{X} - \mu}{\sigma / \sqrt{n}} \sim N(0, 1)$$

This theorem means that almost any data evaluated in the context of data mining can use the Normal distribution. Small samples should use the T-distribution. A few specialized problems benefit from the Chi-Square distribution, but largely, if you understand the Normal distribution, the concepts will apply to most problems.

## Statistics

---

**How are statistics used in data mining to find interesting results?** The discipline of statistics is used by researchers to determine relationships, find patterns in data, and to test research hypotheses. Many of the tools and concepts developed for use in research have proven useful in data mining. However, some key differences in attitude exist between research statistics and data mining. Data mining is an exploratory process. Its purpose is to provide information to analysts to generate intuition and provide directions for additional research. Consequently, many of the statistical tools are applied differently than they would be for scientific research. In scientific research, certain protocols need to be followed to ensure the data observations are independent and unbiased. Data collection is a critical step in the research. Analyzing data for scientific research also constrains how some procedures are used. This section reviews the key concepts in statistics that often arise in data mining. However, it does not deal with the protocols that are critical to scientific research.

Probability theory in the earlier sections described basic rules of probability and provided the foundations of probability distributions. Probability explains how the population or total set of possibilities is related. Statistics deals with a **random sample** of observations from a population. For example, the population might be the set of all people who are potential customers. A sample might consist of actual customers, or of a set of people randomly chosen and contacted by the marketing department. A **statistic** is a random variable that defines some measure on the random sample. Because a statistic is a random variable, it follows some probability distribution. With enough observations, the central limit theorem states that the normal distribution is a good approximation. The measure can consist of almost anything, such as demographic characteristics of people (age, height, weight, income, and so on), or business outcomes including intention to purchase, amount of money spent each month, or profitability.

### Samples

For most research projects, obtaining a good random sample that is representative of the population is critical. A sample is a selection of observations taken from the population of potential data. It costs money and time to obtain research data, so researchers try to obtain the best sample possible at reasonable costs. With data

mining, the concept of a sample is different. Generally, organizations have already collected the data—through traditional transaction processing systems. In many cases thousands, millions, or even billions of rows of data exist. The analyst typically has access to as much data as the computer can process. However, two common situations arise in data mining where samples become useful.

First, sometimes there is simply too much data to analyze it efficiently. If a company has billions of rows of transaction data and needs to perform intensive analyses, it might take too long to obtain results. Some data mining algorithms are designed to handle huge amounts of data efficiently; others that are based on traditional statistics tend to struggle with huge amounts of data. In these cases, it makes sense to take a random sample of observations from the main data set. The analyst can explore the sample data fairly quickly. Any conclusions reached can then be tested against larger samples, or possibly even the entire data set. With preliminary results, it is easier to run one or two tests against the large data set.

Second, data mining faces the risk of over fitting. **Over fitting** is a polite term for stating that the tools and the analyst have pushed too far and tailored the results to the specific set of data. It is one of the key risks that researchers need to avoid, and it is the reason the term data mining was originally a derogatory term. Given a small set of data and enough tools, it is possible to define a model that almost exactly describes every point in the sample. But that model will work only with that specific data set. The model and conclusions could fail completely when applied to other data. One way to reduce the risks of over fitting is to withhold a random sample of the dataset. After the tools are run on the first set of data, they can then be tested against the random sample. If the results are radically different on the second set, then over fitting is a problem. A few data mining tools go even further and split the data into multiple samples. The tools then compute results on each set and combine them to obtain the overall values. By default, most SQL Server tools automatically withhold 30 percent of the data to use as a test sample.

A third possible issue exists with samples, but it is rare in data mining. Sometimes not enough data exists to handle complex analyses. In these cases, a **bootstrap** process can be used. The sample data are analyzed as a distribution and new data is randomly generated that matches that distribution. The initial sample is expanded by randomly adding data that matches the underlying distribution. The process enables the tools to perform more detailed analyses of the data, but there is always a risk that the expanded sample does not completely match the real distribution.

## Common Statistics

Some common statistics are computed for almost any measure. Remember that probability distributions have parameters. These parameters provide a mechanism to adjust the distribution to specific cases. In particular, the Normal distribution is completely defined by two parameters: mean and variance or standard deviation. Hence, it is important to estimate values for these two parameters. They are estimated by the **sample mean** and the **sample variance**, which makes them the two most important statistics to be computed for any problem. The sample mean is the simple arithmetic average:

$$\text{sample mean } \bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$$

The sample variance is the squared deviation from the mean:

$$\text{sample variance } S^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2$$

Note that the variance has  $n - 1$  in the divisor instead of simply  $n$ . This term is important to keep the estimate unbiased. It also leads to the use of  $n - 1$  as the degrees of freedom when using the T-distribution. Loosely stated, the degrees of freedom are reduced by one (from  $n$ ) because the same data were already used to estimate the sample mean. Because the sample mean is used in the definition of the sample variance, one degree of freedom is lost.

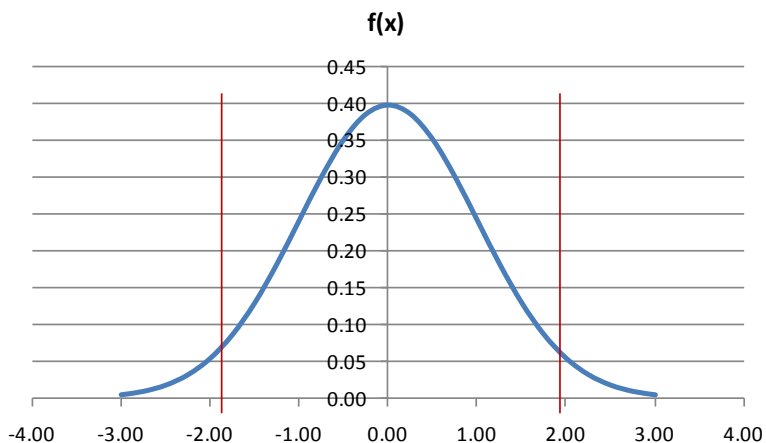
The sample mean and variance commonly appear as parameters in other distributions. They are relatively easy to calculate, and most data mining and statistical packages perform the computations automatically. Even SQL has internal functions to compute these values. Other common statistics include the minimum and maximum values.

With multivariate data, it is also common to compute the sample correlation coefficient to determine the degree of relationship between two variables  $X$  and  $Y$ . The coefficient is defined as

$$r = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2 \sum_{i=1}^n (Y_i - \bar{Y})^2}}$$

**Figure 4.27**

Find 95-percent confidence interval. Normal distribution, 95-percent in the middle leaves 0.025 probability in each tail. Inverse cumulative normal says that occurs at 1.96, or about 2 standard deviations.



However, it is easier to compute by simplifying the equation:

$$r = \frac{\sum_{i=1}^n X_i Y_i - \sum_{i=1}^n X \sum_{i=1}^n Y_i}{\sqrt{n \sum_{i=1}^n X_i^2 - (\sum_{i=1}^n X_i)^2} \sqrt{n \sum_{i=1}^n Y_i^2 - (\sum_{i=1}^n Y_i)^2}}$$

Data mining tools can compute this coefficient automatically. Variations of it are commonly used in regression analysis. It is not necessary to memorize the formula, but it is important to remember that a value of zero means the two variables are uncorrelated or independent. Values of 1 and -1 indicate perfect correlation, but perfect correlation is rare.

### Confidence Intervals

Much of the probability and statistics discussion ultimately reduces to one basic concept that is used for two key tools: Confidence intervals and hypothesis testing. The two concepts are two ways of looking at the same question. Remember that a sample is just that—a sample of observations that might or might not be accurate. A key question is to evaluate the sample data and determine its accuracy. Another way to examine the question is to treat the average as a forecast, and then ask about the accuracy of the forecast. Will actual values be close to the forecast, or does a wide range of possible outcomes exist?

A confidence interval is typically created by assuming the statistic follows a normal or T-distribution. Define a Z (or T) value as the difference between the average and true mean divided by the standard deviation of the average:

$$Z = \frac{\bar{X} - \mu}{SD(\bar{X})}$$

The true mean exists but cannot be observed directly, so rewrite the equation:

$$\bar{X} - (Z)SD(\bar{X}) \leq \mu \leq \bar{X} + (Z)SD(\bar{X})$$

### Figure 4.28

Find 95-percent confidence interval for average bill in Diners database. A simple totals query retrieves the count, average, and standard deviation of the bill total. Compute the standard deviation of the average and multiply by the Z value of 1.96.

```
Avg(X) = 83.26
StDev(X) = 73,958
N = 73,233
StDev(Avg) = 73.958/sqrt(73,233) = 0.272
Z = 1.96
95-percent CI: 83.26 - 1.96(0.272) , 83.26 + 1.96(0.272)
                = (82.73, 83.79)
```

As shown in Figure 4.27, choose the value of  $Z$  so that this interval has a specified probability of containing the true mean. For instance, it is common to pick a 95-percent CI. To get 95-percent of the probability contained in the interval leaves 0.025 percent in each of the two tails. In the old days, you would read through probability tables to find the  $Z$  value that leaves 0.025 in each tail. Today, you can use the computerized inverse cumulative Normal function (NormInv in Excel) to learn that point occurs for  $Z=1.96$ , which most people round off to 2. An interval that is plus-or-minus two standard deviations wide has a 95-percent chance of containing the mean. In terms of forecasting, future values would have a 95-percent chance of falling within that interval. Some people also like to examine the 99-percent confidence interval. The  $Z$  value for a two-tailed 99-percent CI is 2.58.

Finding a confidence interval for the mean requires computing the mean and the standard deviation of the mean. Be careful when reading that sentence. It says “standard deviation of **the mean**,” not standard deviation of  $X$ . The typical variance or standard deviation functions compute the standard deviation of the original data ( $X$ ). But the average is the sum of the  $X$  values divided by the number of observations  $n$ . Based on some probability rules that are beyond this book, the variance of the mean is  $V(X)/n$ . Take the square root to get the standard deviation of the mean:

$$SD(\bar{X}) = SD(X) / \sqrt{n}$$

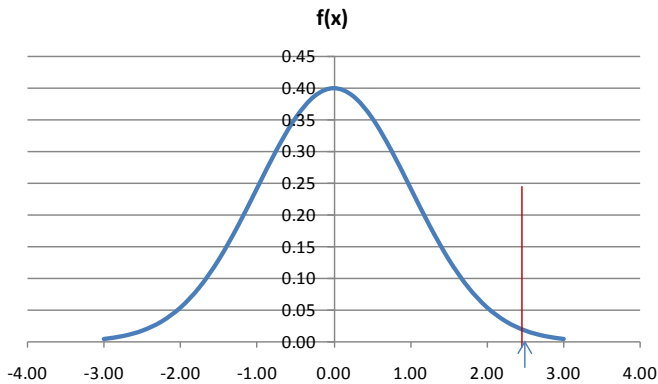
The process is straightforward, and all of the components have been defined. Consider a simple example using the small Diners database. The owner/chef wants a 95-percent confidence interval for average purchases on a single sale. The data are already stored by individual table or sale and includes the BillTotal. Create a simple totals query:

```
SELECT Avg(Diners.BillTotal) AS AvgOfBillTotal,
       StDev(Diners.BillTotal) AS StDevOfBillTotal,
       Count(Diners.DinerID) AS CountOfDinerID
FROM Diners;
```

Figure 4.28 shows the results of the query with the average and standard deviation of the Bill Total. Divide by the square root of the number of observations to obtain the standard deviation of the average. Plug these values and the  $Z$  value of 1.96 into the CI formula to learn that the confidence interval is (82.73, 83.79). This interval is tight because the standard deviation is low from the large number of observations. Remember that it is the interval for the mean, not for a specific bill. It means that over this time period, the average is an accurate indicator of the overall mean. This number (83.26) could then be used to compute reliable estimates of the revenue for the restaurant. For example, assuming the number of customers can be forecast, multiply by the average bill total to obtain the revenue estimate.

Use the same data to compute the 99-percent confidence interval. The only number that changes is the  $Z$  value, which becomes 2.57 instead of 1.96. The resulting interval is (82.56, 83.96). It is wider than the 95-percent CI, which will always happen. To be more confident, the interval has to be wider. But, because the standard deviation of the mean is so low, the effect is negligible—about 25 cents or less than one percent of the mean.





**Figure 4.29**

Hypothesis test. With large  $n$ , compute  $Z = (\text{Avg-hypothesis})/\text{SD}(\text{Avg})$ . Say it is 2.5 in the example. Under the null hypothesis that the mean is zero, what is the probability 2.5 can be observed?  $N(2.5, 0, 1) = 0.99379$  (area to left of 2.5), so tail is  $1-0.99379 = 0.00621$ . Double it for two tails. Still less than a 1.5 percent chance of error if reject null hypothesis.

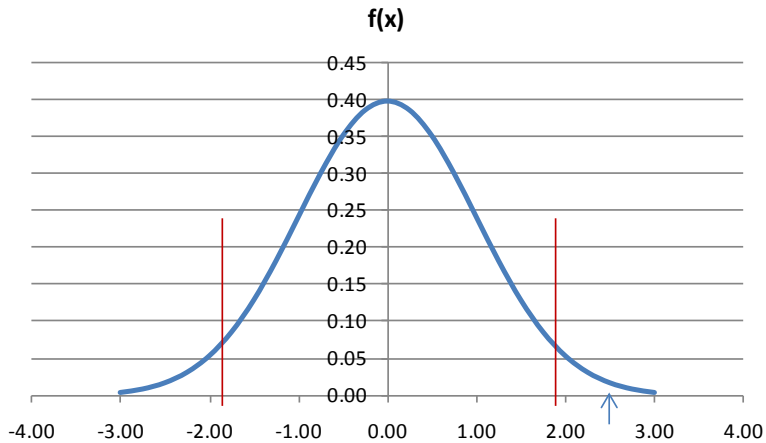
The process for computing confidence intervals for the mean is similar for any data. Find the average, standard deviation, and count. Compute the standard deviation of the mean and multiply by the desired Z-value. Add and subtract this value from the mean.

The process of creating confidence intervals is applied to many types of problems. The main challenge lies in finding the correct value for the standard deviation. The formula shown in this section applies to the mean. If you wanted to create a confidence interval for a forecast of the original data, the standard deviation is much higher—it includes the standard deviation of the mean and the original data. Many data mining packages create confidence intervals for forecasts and they automatically compute the standard deviations. In most cases, you simply need to understand how to interpret the confidence interval.

## Hypothesis Testing

Because it uses the same concepts, mean, standard deviation, and Z-value, **hypothesis testing** is similar to creating confidence intervals. The focus is slightly different, so a few additional concepts and terms are necessary. The goal of hypothesis testing is to decide if a proposed statement (hypothesis) can be rejected. Note the wording—rejected, as opposed to accepted. Technically, probability can only reject statements; which can lead to some interesting configuration of problems. However, most hypotheses in data mining are straightforward. The typical statement is that there is no relationship between variables, or that the effect of a particular dimension is zero. If this type of statement is rejected, then the relationship does exist or the dimension does have a significant impact.

Figure 4.29 shows one way to look at the hypothesis test. Assume enough observations exist so that the mean follows a normal distribution. Compute the test statistic for the problem as  $Z = (\text{average} - \text{hypothesis})/\text{SD}(\text{average})$ , where the hypothesized value is zero. Say the result is  $Z=2.5$ . Under the assumption that the



**Figure 4.30**

Hypothesis test is similar to a confidence interval. Define the level of Type I error to obtain the critical values (Normal with 5 percent in two tails is 1.96). Compute the Z statistic (Avg. – hypothesis)/SD(Avg). If the statistic exceeds the critical value (in absolute value) and falls outside the confidence interval, reject the null hypothesis.

hypothesis is true (zero mean), what is the probability that this result or higher could arise by chance. From the cumulative standard normal formula,  $N(2.5, 0, 1) = 0.99379$  which is the area to the left of the value 2.5. The area to the right in the tail is  $1 - 0.99379 = 0.00621$ ; or less than one percent probability. A two-tailed test is more common—where the computed average could be higher or lower than the true mean. In that case, double the tail probability (one for each side), and the result is less than 1.3 percent. So, rejecting the mean could result in being wrong no more than 1.3 percent of the time.

This probability of being wrong is a key element in hypothesis testing. Technically, two types of errors can arise in a hypothesis test:

Type I error: Reject the null hypothesis when it is true.

Type II error: Accept the null hypothesis when it is false.

Of these, the Type I error is usually considered the most important. In dealing with probabilities, no absolute answers exist—random chance could always crop up with atypical results. Most scientists want to be conservative and only reject a hypothesis if strong evidence exists. So, most hypothesis testing begins with a statement of the amount of Type I error you are willing to accept. Common values include 5 percent or 1 percent. A 1 percent error might seem “better,” but it increases the amount of Type II error—there is a tradeoff. The only ways to reduce both types of error is to increase the number of observations and decrease the standard deviation.

Defining a level of Type I error simplifies hypothesis testing. The process becomes:

1. Define the null hypothesis.
2. Define a level of Type I error.

Count: 13,497  
 Average: 153.75  
 SD(Bill): 93.2479  
 $SD(Average) = SD(Bill)/\sqrt{N} = 0.8026$   
 $Z = (153.75 - 83.26)/0.8026 = 70.49/0.8026 = 87.82$

### Figure 4.31

Statistics for Bill Total on Saturdays. Compute the Z-statistic and notice that it is huge! Clearly  $Z = 87.82 > 1.96$ , so reject the null hypothesis. Diners on Saturday definitely spend more money on average than diners overall.

3. Look up the critical values as a Z or T statistic, usually as a two-tail test.
4. Compute the Z (or T) statistic:  $(Avg. - hypothesis)/SD(Avg)$ .
5. If the statistic is larger (absolute value) than the critical value, reject the null hypothesis.

Figure 4.30 shows how the hypothesis test is similar to a confidence interval. When the level of acceptable Type I error is set, the **critical values** can be found (typically 1.96 or 2.58 for 5 percent or 1 percent error). If the test statistic falls outside of this interval, the null hypothesis is rejected because that result can arise with less than the specified probability. That is, the mean is most likely not equal to the hypothesized value (e.g., zero). In the example, the test statistic is 2.5, which is greater than 1.96, so the null hypothesis is rejected.

This approach to hypothesis testing is relatively easy to perform in practice—particularly when the hypothesized value is zero. Most computer programs display the value of the desired item or average, along with the standard deviation. Simply divide the item's value by its standard deviation to obtain the T (or Z) statistic; and many programs display this value automatically. If the absolute value of the ratio is larger than 1.96 (roughly 2), reject the null hypothesis.

Consider a simple example, again using the Dining database. Perhaps in exploring the data, you noticed that people seem to spend more money on some days than on others. Is this difference significantly different from the overall average? To answer this question, formulate a basic null hypothesis: The average bill on Saturdays is equal to the overall average bill of 83.26 found in the previous section. Set a Type I error rate of 5 percent. Remember that the number of observations is in the thousands, so the Normal distribution can be used instead of the T distribution; hence, the critical value is 1.96. Create a database query to compute the average and standard deviation of the bill total on Saturdays.

```
SELECT Count(Diners.DinerID) AS CountOfDinerID, Avg(Diners.
BillTotal) AS AvgOfBillTotal, StDev(Diners.BillTotal) AS
StDevOfBillTotal
FROM Diners
WHERE (Diners.DOW='Sat');
```

Figure 4.31 shows the results of the query, with a count of 13,497 leading to an average Bill Total of 153.75 and a standard deviation of 93.25. Note that the

average bill for all days is about \$83 and the average for Saturdays is about \$154. Is this difference large enough to be significant? The hypothesis tests answers this question and it does it by including the standard deviation. The figure shows that the Z statistic computes to 87.82 which is a huge number. It is clearly greater than the critical value of 1.96, so the null hypothesis should be rejected, and you decide that people do indeed spend more money on Saturdays than other days.

A statistician would observe that this hypothesis test could be improved. For example, perhaps a better statement would be to compare Saturday averages to averages on other days—leave the Saturday values out of the overall total. It would not change these results, because the average without Saturday is going to be even lower, leading to a larger Z value. But, it could make a difference for other days. Similarly, managers might want to make comparisons between two specific days of the week. Comparing two items to each other leads to problems with defining the standard deviation—because both days might have different values. Statisticians have found several ways to define the standard deviations for comparisons. These distinctions do not arise very often in data mining, but you should ask for help from a statistician if you encounter problems that require comparisons of two sets of numbers. Be particularly cautious if the values are paired—such as observations taken from the same person at different points in times. These cases are not considered here, but the formulas for the variances can be found by searching for paired T test or pooled variance for cases when the variables are independent.

### Chi-Square Hypothesis Tests

The T-distribution and Normal distribution are useful for most problems—particularly tests of means. However, some problems are best solved with other distributions. In particular, the Chi-Square distribution is used in two common types of problems. (1) Testing a variance, and (2) goodness of fit. Because variance is defined as the square of the deviation from the mean, the variance follows a Chi-Square distribution instead of a normal distribution—particularly with a relatively

**Figure 4.32**

Goodness of fit example. A company wants to see if managers are promoting men and women in equal proportions. Assume each manager has the same promotion opportunities. Under the null hypothesis of no discrimination, the total ratio can represent the expected number of promotions.

	Female		Male		Total	
	N	Promoted	N	Promoted	N	Promoted
Manager1	27	15	32	12	59	27
Manager2	21	9	29	11	50	20
Manager3	16	4	21	5	37	9
Manager4	31	7	41	9	72	16
Manager5	9	3	7	2	16	5
Manager6	17	7	11	2	28	9
Manager7	19	5	7	1	26	6
Total	140	50	148	42	288	92
ratio		0.357		0.284		0.319

	Female			Male		
	Obs	Expected	X2	Obs	Expected	X2
Manager1	15	8.625	4.712	12	10.222	0.309
Manager2	9	6.708	0.783	11	9.264	0.325
Manager3	4	5.111	0.242	5	6.708	0.435
Manager4	7	9.903	0.851	9	13.097	1.282
Manager5	3	2.875	0.005	2	2.236	0.025
Manager6	7	5.431	0.454	2	3.514	0.652
Manager7	5	6.069	0.188	1	2.236	0.683
			7.2347			3.711794

**Figure 4.33**

Goodness of fit results. To test the chi-square totals, find the chi-square value that has 5 percent in the right-tail with 6 degrees of freedom. The value is 12.59 and can be found using the Excel formula: ChiInv(0.05, 6). Both test statistics are well below this critical value so the null hypothesis cannot be rejected. But, it might be worth looking deeper into some promotions by Manager1.

small number of observations. Hypothesis testing is similar to that for means, but the statistic is:

$$X^2 = (n-1) s^2 / \sigma^2$$

The statistic uses the Chi-Square distribution with  $n - 1$  degrees of freedom. It is used to test whether the variance is equal to (or greater than) some value. This type of test sometimes arises in data mining but it is rare.

**Goodness of fit** is a more interesting application of the Chi-Square distribution. How do you know if the data match a particular distribution? Or, how do you test if data from two sets have the same distributions? The goodness of fit test is one answer. Divide the problem into segments. For each segment, use the probability distribution to compute the expected number of observations in that segment. Then count the observed number of items in that segment. Compute the Chi-Square statistic as:

$$X^2 = \sum_{j=1}^J \frac{(O_j - E_j)^2}{E_j}$$

This statistic has a Chi-Square distribution with  $J-1$  degrees of freedom. If the number of observed values is consistently different from the expected number, the statistic has a high value and the Chi-Square test will reject the null hypothesis that the data matches the expected distribution. A similar version of this test is used to test for independence between two variables—using a contingency table.

Chi-square goodness of fit is a useful way of comparing data across a company. Figure 4.32 shows a potential use for goodness of fit in a data mining example. A company wants to know if managers are promoting men and women in equal proportions. Retrieving the appropriate data is the first data mining challenge. Under the null hypothesis of no discrimination, and assuming all managers have

Set	H(X)
Uniform random: $p=1/25$	10.87
Random draw $0 < p < 1$	9.5
$N(0,1)$ from -3 to 3	8.1

**Figure 4.34**

Sample values of Shannon's entropy measure for various sequences of 25 probabilities. Uniform random always has the highest entropy value.

equal opportunities to promote workers, the ratio of the total promoted can serve to compute the expected values for each category and manager.

Figure 4.33 shows the computation of the Chi-square statistics. In the end, the totals seem relatively small. To check, find the Chi-square value with 6 degrees of freedom that has 5 percent probability in the right-hand tail. That value of 12.59 is the critical value for testing within each gender across all managers. Both values are below that critical level, so statistically, the null hypothesis cannot be rejected. It is also possible to test both groups at the same time by adding the two separate sums and checking the Chi-square critical value with 13 degrees of freedom. That critical value is 22.36, which is clearly higher than the total of 10.9. On the other hand, it might be worth checking into promotions by Manager1. Summing the two cells ( $4.7+0.3$ ) provides a statistic value of about 5. The critical value with 1 degree of freedom is 3.8, so the null hypothesis would be rejected on this data.

Statistical tests of this nature can be powerful tools to automatically scan the data looking for anomalies. But, keep in mind that standard statistics practices are geared towards research. It is quite possible that small patterns exist that will not be revealed with a 5 percent Type I error rate. That is, patterns can exist but not be strong enough to pass a rigorous research test. One solution is to increase the Type I error rate, reducing the critical values and highlighting more potential anomalies. Just be cautious when interpreting the results, and use other information to make critical decisions. Data mining used in this manner is useful for exploration and providing directions for further study. It generally cannot be used for proving a case or making a final decision.

### Information Measure

There is a statistic that is commonly used in data mining that is rarely presented in introductory statistics classes. In 1948, Claude Shannon (Shannon 1948) published a paper dealing with communication theory. A key aspect of communication is the ability to deliver information, so Shannon formulated a mathematical definition of information. His main point is that information must contain something new, or perhaps surprising. The classic example is a coin flip. If a coin has two heads and no tail, it will always come up heads. Hence, the coin flips provide zero information—there is no surprise. At the other extreme, if a coin is fair, the probability that a head is revealed constitutes new information because there was no way to predict the value ahead of time.

Shannon's **information measure** or **entropy** is defined for a random variable  $X$  as

$$H(X) = E[I(X)]$$

In the definition,  $E$  is the standard expectation function from probability and  $I$  is an information measure of the  $X$  variable. Information is based only on the probability of the event occurring, and events with more surprise have more information ( $1/p$ ), so the function  $I$  is defined as

$$I(X) = \log(1/p) = -\log(p)$$

Because Shannon was dealing with bits transferred, he used the base-2 log and information was measured in bits. However, the definition is neutral and could use any base for the log function. To understand the information function, consider the simple coin toss. With a fair coin,  $p = 1/2$ . When a head is tossed, the information revealed is  $\log_2(1/0.5) = \log_2(2) = 1$ . In a communication sense, it takes one bit to send the information that a head was tossed. Similarly one bit is needed to hold the information that a tail was tossed. The bit could be 1=heads, 0=tails. In a larger problem, rolling a fair die, a specific number (e.g., 5) has  $1/6$  probability of appearing, so  $-\log_2(1/6) = 2.585$ , or almost 3 bits to carry the information content of the results of a roll of a die.

Combining these definitions, the standard measure of Shannon's entropy is:

$$H(X) = -\sum_{i=1}^n p_i \log(p_i)$$

where  $p_i$  is the probability of  $x_i$  appearing, and the values are summed over all possible values of  $X$ .

The formula is easy to compute, but it requires the probability values for each of the outcomes. To understand it a little better, build a simple spreadsheet and compute the entropy measure for a variety of sets. Figure 4.34 shows the results for three different random variables. A uniform random distribution would have the same probability ( $1/25$ ) at each point. A random draw was created to define 25 probabilities, and the normal pdf was computed at 25 points from -3 to 3. Uniform random variables will always have the highest entropy value, because the probabilities are equal and the results cannot be predicted, so the surprise value is the highest. Consequently, Shannon's entropy is a measure of randomness. Consequently, the measure is often used to evaluate different models to see which one reduces entropy (randomness) the most. The entropy measure is sometimes used to help determine if a dimension variable has enough information to be useful in further analysis. For instance, if gender were a dimension but all of the data was based on women, the Shannon information value would be minimal and there would be little point in including that dimension. Conversely, the information gain on a target variable is a useful way to evaluate the impact of adding new dimensions.

## Summary

---

Probability and statistics are useful and powerful tools that form the foundation of data mining. The rules of probability provide the mathematical foundations that lead to probability distribution functions. These distribution functions are used to evaluate statistics and eventually to make hypothesis tests about the data. Random variables can contain discrete or continuous data and the distributions are different. The binomial and Poisson distributions are commonly used for discrete data. The Normal distribution for continuous data is the most important distribution because the central limit theorem says that all distributions approach the Normal



when the number of observations is large. The T-distribution corrects for degrees of freedom biases in small samples. The Chi-square distribution is used to perform hypothesis tests on the variance, but it is more commonly used to test for equality of distributions and independence.

The Normal and T-distributions are commonly used for testing hypotheses on the mean. The process is the same for both, but the T-distribution requires the number of degrees of freedom—which is usually  $n - 1$ . The basic process is to compute the test statistic as  $(\text{Avg.} - \text{hypothesis})/\text{S.D.}(\text{Avg.})$ . The null hypothesis is typically that the mean is zero, so the computation simply becomes the estimated average divided by its standard deviation. This ratio is generally reported by data mining tools. If the ratio is larger than the critical value (1.96 for Normal at 5 percent error) then the null hypothesis is rejected. Values that exceed the critical level tend to be important in data mining problems.

Bayes' Theorem is another useful result from probability theory that is used in various aspects of data mining. It defines how to compute a conditional probability. Derived from basic probability rules, the theorem is best understood in terms of subjective probabilities. Beginning with a prior believe about probability (or its distribution), Bayes' Theorem shows how to use data observations to update the probability to a new posterior probability.

Probability and Statistics have detailed high-level mathematical definitions. Some basic definitions are presented in this chapter simply to illustrate these foundations. Fortunately, statistical data mining tools incorporate this knowledge and perform the computations automatically, so it is not necessary for managers to know all of the details. Instead, it is critical that managers using data mining tools understand the fundamentals and be able to interpret the results and conclusions. Two of the most difficult problems faced by analysts are (1) choosing the appropriate tools, and (2) understanding the results and limitations of the tools. A basic knowledge of probability and statistics is required to perform these tasks.

## Key Words

---

addition rule	multiplication rule
arrangements	mutually exclusive
Bayesian	over fitting
bootstrap	parameters
central limit theorem	permutation
combination	Poisson distribution
conditional probability	probability
contingency table	probability density function (pdf)
continuous	probability distribution
critical values	probability function
cumulative distribution function (cdf)	probability mass function
discrete	random
discretizing	random sample
expected value	random variable
experiment	relative frequency
general multiplication rule	sample mean
goodness of fit	sample variance
hypothesis testing	Shannon entropy
independent	standard deviation
information measure	statistic
joint events	subjective probability
joint probability	uniform distribution
margin totals	variance

## Review Questions

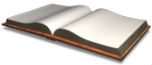
---

1. What is the probability of two events both occurring if they are independent?
2. When counting outcomes, what is the difference between arrangements, permutations, and combinations?
3. What is the general probability rule for computing the probability that event A or event B can occur?
4. What is Bayes' Theorem and why is it important in data mining?
5. How can a tree diagram represent Bayes' Theorem? Hint: Draw a simple tree and explain it.
6. What is the difference between discrete and continuous data?
7. What is expected value?
8. When evaluating data, why is it so important to look at the variance as well as the mean?
9. What is the purpose of the correlation coefficient?

10. When should a T-distribution be used instead of a Normal distribution?
11. How is a confidence interval similar to a hypothesis test?
12. What is the test statistic for comparing an observed distribution of data to another specified distribution?
13. When does Shannon's entropy measure reach its peak values?

## Exercises

---



### Book

1. A company has just hired 15 interns and wants to assign them to teams of 5 people each. The team members will be rotated every month. One manager has suggested it would be nice to set up all possible teams. How many possible teams can be made from these 15 people?
2. A company produces a complex machine that has 10 key components. The probability of any individual part failing is given in the table. Fortunately, the system is not dependent on all parts equally. Some parts are essentially backups for others. The system fails if: (a) C1, C5, and C7 all fail, (b) C8, C9, and C10 fail, (c) C2 and C4 fail, or (d) C3 and C6 fail. What is the probability any individual machine will fail? If the company sells 10,000 machines, what is the expected number of failures?

Individual failure rates:

1	2	3	4	5	6	7	8	9	10
1/100	1/300	1/500	1/50	1/800	1/700	1/100	1/200	1/50	1/600

3. A factory produces identical components from two machines. Machine A is older and is known to produce defects at the rate of 1/1,000. Machine B produces defects at the rate of 1/5,000. After a daily run of 5,000 parts produced from A and 10,000 parts from B, an inspector discovered that a part in a combined box is defective. Draw the decision tree and compute the probability that the defective part came from Machine B.
4. Use Excel to create a table and a chart for a binomial distribution where the probability of success is 3/4 and there are 20 trials.

5. A bank is thinking about adding a second ATM machine. On Fridays at the end of the month during lunch hour, an average of 20 people an hour use the existing ATM. The one machine can barely handle those 20 people because each takes an average of 3 minutes per person to complete their transactions. Compute the probabilities of from 21 – 30 people arriving to use the ATM. The manager has decided that if 30 people arrive too often, a second ATM will be needed.
6. Given the values in the following table, find the probability that the mean is greater than zero using both a Normal and T-distribution for each of the row entries.

Avg	SD(X)	N
5	5	100
5	5	10
5	20	100
2	10	50
2	10	100

7. Given the data in the following table, create the 95-percent confidence interval for the mean for all the data points.

9.12	11.45	5.07	3.86	6.32	2.45	4.40	4.57	7.30	5.81
3.17	10.87	-0.33	2.90	-1.30	5.41	3.47	2.31	5.48	4.66
8.57	5.12	4.07	3.68	5.34	9.33	-0.45	2.51	7.30	4.52

8. Given the following values reported by the data mining software, determine whether the mean is significantly different from zero in each of the row cases.

Avg	SD(X)	N
2.5	70	100
12.3	35	20
-5.4	125	50
18.4	65	100
21.7	25	50

9. Your ten coworkers have accused your boss of making decisions by flipping a coin. In an attempt to disprove this hypothesis and argue that your colleagues do not understand your boss, you have asked each of them to count the number of “yes” responses out of the next ten questions asked of the boss. The following table records the number of times the boss said “yes” out of ten requests. Using the data, is your boss really flipping a coin to make decisions?

# Yes	0	1	2	3	4	5	6	7	8	9	10
Count	0	0	0	0	1	1	2	3	2	1	0

10. Create an Excel spreadsheet to compute the following values:
- Normal probability for  $X > 32$  when the mean=25 and std. dev. = 5.
  - Binomial probability finding three or fewer errors in 10 trials where the probability of any given error is 1/10.
  - A salesperson is paid \$100 for every new customer who places an order in the next month. If the average salesperson gets 3 out of 50 people to place an order, how much money can a salesperson expect to make by calling 1000 people?
  - Someone has estimated the following probability distribution for events. What is the Shannon Entropy measure?

Event	Total destruction	Irreparable damage	Major damage fixable in a month	Minor damage	Complete success
Probability	0.10	0.15	0.15	0.25	0.35



### Rolling Thunder Database

- Compute sales by state for 2010 and decide if sales to CA are different from the average.
- Compute sales (count) by gender for 2012 and determine if men buy more bicycles than women.
- Are sales of bicycles by model type for 2011 evenly distributed based on count?
- For 2012, use the relative frequencies to estimate the probability of bicycle model type. For each model type, estimate the probability of the bicycle having a carbon frame given the model type. Then use Bayes' Theorem to compute the probability that a carbon bike just sold is a race bike.
- For 2012, what is the correlation between bicycle size (FrameSize) and sale price for road and race bikes (combined)?



## Diner

16. Is the purchase of desserts equally probable on each day of the week?
17. Build a frequency distribution of total sales (BillTotal) by day of week. Does it resemble any of the distributions covered in this chapter? Hint: If necessary, use the Excel Frequency function.



## Corner Med

18. Create a decision tree with estimated probabilities, from patient gender, age (<5, 6-50, >50), and tobacco use.
19. Create a frequency distribution histogram by count of the number of visits covered by each insurance company.
20. Create a frequency distribution by counting the number of visits for each top-level ICD10 procedure code (first letter).



## Basketball

21. For each team, compute the number of wins for non-playoff games in 2010-2011 and compute the team's free throw percentage. Across the teams, what is the correlation coefficient between these two variables? Hint: Use the TeamGameTotals view.
22. Compute each team's win percentage for the regular season (82 games). Using this data as  $p$ , compute Shannon's entropy. Compare the resulting value to the entropy of all 30 teams winning exactly 50 percent of their games.
23. Use a Chi-Square test to determine if each division has the same potential/record. That is set the null hypothesis that each division wins the same number of games in the 82-game season for 2010-2011.
24. Across all teams, did guards score more points on average than centers did in 2010-2011?



## Bakery

25. In terms of average sales value (quantity \* sale price) per day, do breads sell more than the average of all other products? Hint: Compute sales per day by category.
26. Using relative frequencies, what is the probability that bread and cake are purchased at the same time. Hint: Count the total number of sales that include at least one bread item and one cake.



## Cars

27. Compute the correlation between MPG and acceleration (seconds to 60 mph). Comment on the results.
28. Create a frequency distribution histogram for the MPG values. Comment on the chart. Does the result resemble any of the distributions covered in this chapter?
29. Divide weight into three categories: light, medium, and heavy. Do the same for horsepower and price. Create a decision tree for these three attributes and include the probabilities for each node. Comment on any patterns.



## Teamwork

30. Choose one basketball team. Compute the average free throw percentage and standard deviation for the entire team for one season. Assign at least two players to each person on in your group. Compute the player's free throw average and determine if it is significantly different from the average for the team.
31. Split the bakery data into days of the week and assign one day to each team member. Have each person compute the observed probability of a cake and bread item being purchased at the same time on the specified day of the week. Compare the team's results for the different days.
32. Look at the cars by the predefined category. Assign one category to each team member. Compare the average MPG, Price, and Weight to the overall averages. Are they statistically different for the specified group? Combine the team's results and comment on any results.



## Additional Reading

---

Mlodinow, Leonard, 2008, *The Drunkard's Walk: How Randomness Rules Our Lives*, Pantheon Books: New York. [A collection of examples on how difficult it is to apply probability to everyday life. No mathematics needed.]

Shannon, Claude, 1948, "A Mathematical Theory of Communication," *Bell System Technical Journal*, 47, 379-423. [Shannon's original definition of information.]

Trevor Hastie, Robert Tibshirani, and Jerome Friedman, 2001, *The Elements of Statistical Learning*, Springer: New York. [An outstanding book on data mining, with an emphasis on theory. A graduated-level book that requires a strong mathematics background.]

Zellner, Arnold, 1971, *An Introduction to Bayesian Inference in Econometrics*, Wiley: New York. [A classic book on Bayesian theory, particular focus on subjective probabilities and how they can define traditional analyses. Graduate level with mathematics.]

# Business Analysis

**What common data mining tools are available and how are they configured and used?**

Almost any statistical tool has been used in a data mining context. However, several key techniques are used to solve common problems. From a business perspective, the main applications fall into four categories: (1) Clustering items into groups based the values of attributes, (2) Determining which items or events occur together, such as identifying which items are purchased together, (3) Evaluating dimensions to see how various attributes influence facts, such as identifying the factors that affect sales, and (4) Analyzing time series data to spot patterns and forecast data.

This section explains how to organize data for the specific type of analysis, set up the analysis tool, and interpret the results. Setting up the data usually requires knowing something about SQL; however, the specific examples include the necessary SQL.

---

<b>Chapter 5</b>	Cluster Analysis
<b>Chapter 6</b>	Association and Market Baskets
<b>Chapter 7</b>	Evaluation of Dimensions
<b>Chapter 8</b>	Time Series Analysis

---

# Cluster Analysis

## Chapter Outline

- Introduction, 218
- Business Situation, 220
- Model, 221
  - Distance or Dissimilarities*, 222
  - Combinatorial Searches with K-Means*, 224
  - Statistical Mixture Model with EM*, 227
  - Hierarchical Clusters*, 229
  - Other Statistical Methods*, 233
- Data, 236
  - Attributes and Observations*, 236
  - Continuous and Discrete Data*, 237
  - Missing Data*, 238
- Clustering on Products: Cars, 238
  - Goals*, 238
  - Data*, 239
  - Microsoft Clustering*, 241
  - Results from Microsoft Clustering*, 243
  - Prediction*, 246
  - Larger Model and Parameter Changes*, 248
- Traditional EM Clustering, 251
  - Goals and Data*, 252
  - Results*, 254
  - K-Means Clusters*, 255
- Comparison, 256
- Customer Clustering with Categorical Data, 258
  - Data*, 258
  - Microsoft Clustering Results*, 259
  - Weka Clustering Results*, 260
- Summary, 262
- Key Words, 263
- Review Questions, 263
- Exercises, 264
- Additional Reading, 267

## What You Will Learn in This Chapter

- Are all customers the same?
- Why would a business use clustering analysis?
- How does clustering work and how are the results interpreted?
- What type of data is used in clustering?
- How are clusters identified with multiple dimensions?
- How are traditional EM methods different from Microsoft Clustering?
- Are the results from the multiple tools and methods really different?
- How does categorical data change the results and interpretation of clusters?

### Alberta

Cluster analysis is a common tool in marketing. In particular, it is used to categorize customers into groups. By identifying common features, it becomes possible to find people with similar features and target advertising to their needs. For example, two professors categorized five types of domestic tourists in the province of Alberta, Canada. Domestic tourists are those from within the region (Alberta) who travel and spend money on vacations and visits within the same region. The researchers used focus groups to identify potential attributes—particularly reasons given for traveling. Telephone surveys were used to collect data, and the cluster analysis identified five major groups of travelers as outlined in the table. Clustering software identifies the five clusters based on the attributes. The researchers added the descriptive titles of the clusters. The Alberta marketing organization then used these clusters and resulting preferences to create TV, radio, and newspaper ads to target each of the groups. [Hudson and Ritchie 2002]

1. Young Urban Active Outdoor N=520,730	2. Indoor leisure traveler N=586,285	3. Children-first traveler N=446,516	4. Fair-weather friends N=445,280	5. Older cost-conscious traveler N=754,501
M=49% F=51% Avg. age: 37.5 Married: 66% >\$100,000: 16%	M=31% F=69% Avg. age: 40.2 Married: 70% >\$100,000: 9%	M=50% F=50% Avg. age: 42.5 Married: 75% >\$100,000: 19%	M=59% F=41% Avg. age: 44.2 Married: 62% >\$100,000: 13%	M=44% F=56% Avg. age: 44.8 Married: 62% >\$100,000: 11%
School holidays Cost/value Safe/secure	Safe/secure Cost/value Weather	Children's sports Safe/secure Cost/value	Family/friends Weather	Safe/secure Cost/value Weather

Row 1: label assigned by researchers and the count.

Row 2: Cluster rules and percentages

Row 3: Top key words

Clusters are a key tool for unsupervised learning. With minimal configuration, the tools can find groups of items that are similar. These groups reduce complexity, allowing decision makers to focus on a few key attributes.

Simon Hudson and Brent Ritchie, 2002, "Understanding the Domestic Market Using Cluster Analysis: A Case Study of the Marketing Efforts of Travel Alberta," *Journal of Vacation Marketing*, 8(3), 263-276.

## Introduction

**Are all customers the same?** It is unlikely that all customers are the same. If so, the organization needs to work on expanding its offerings. But, are there groups of customers that act similarly? Several marketing companies provide services to interview and identify types of customers. These groups are given names and sample pictures so managers and salespeople can visualize each customer group. In many cases, product lines are designed specifically for each target group. Customer grouping is a classic application of **clustering**. The goal is to find clusters or groups so that people who fall within a specific group have more in common with members of that group than with any other cluster.

The concept of “more in common” is defined by the attributes or dimensions available. The values of the attributes are evaluated in terms of a distance measure. For example, if age is an attribute and two customers have the same age, then the distance measure is zero and they will be placed in the same group based on that attribute. Customers who are farther apart in age are likely to be placed into different clusters. Of course, with multiple attributes distance has to be measured across all attributes. Typically, the distance measures are summed across attributes.

Clusters are useful to summarize or reduce the number of dimensions. A company could have millions of customers with dozens of attribute measures. But if clustering can reduce them into 5 – 10 categories, then marketing can focus on those groups. The same concepts apply to other topics, such as products, inputs, regions, or almost anything with multiple dimensions. Instead of trying to treat every dimension separately, clustering looks for internal correlations where collections of various attributes are shared by a large enough number of items.

**Figure 5.1**

Simple cluster example with two attributes. The two clusters were artificially created so the separation is clear.

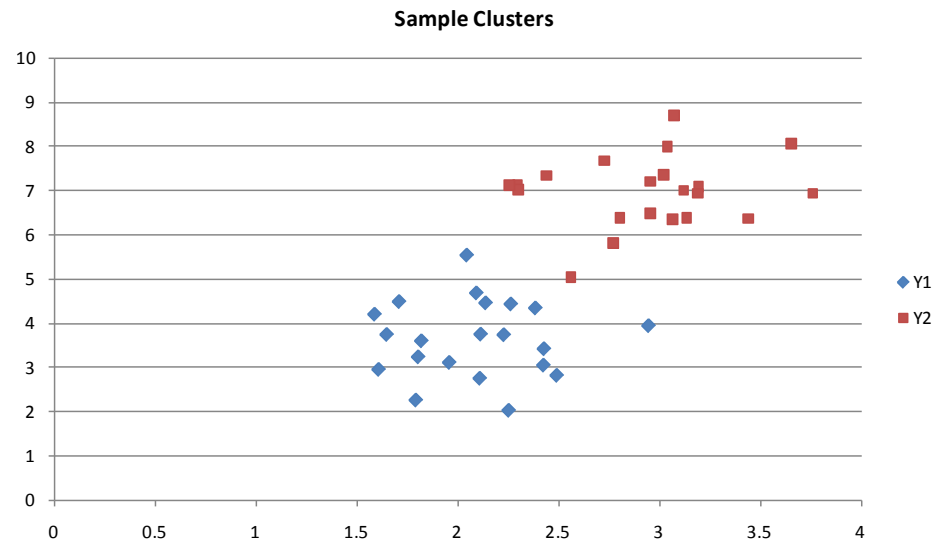


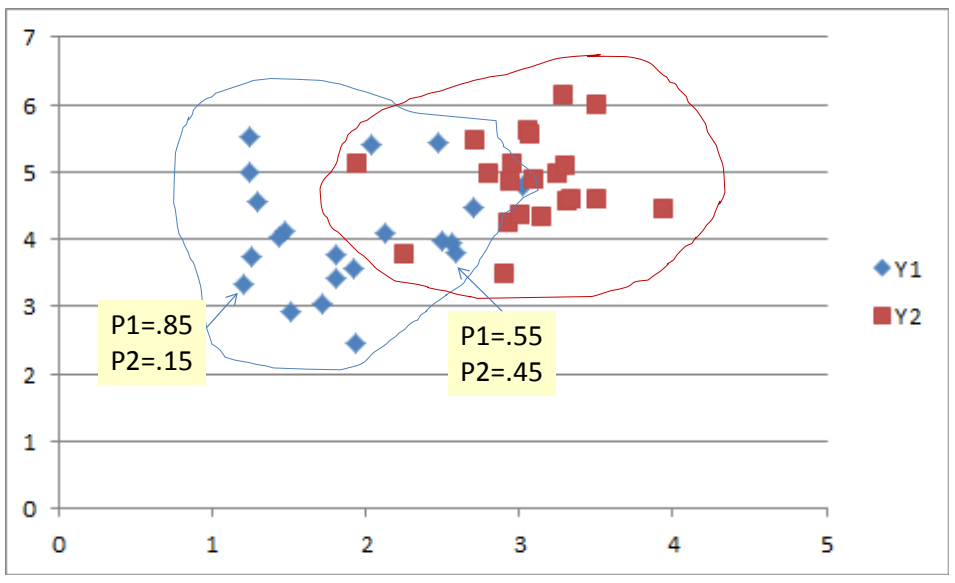
Figure 5.1 shows an example of a simple problem with two attributes and two clusters. The data values were deliberately created to ensure a clean separation of the clusters. Most real-world problems are less clear and the lines between clusters are not always clear. Consequently, different measurement methods can lead to different definitions of the clusters. The choice of method used to estimate the clusters also affects the results. With some methods, a point can be associated with more than one cluster. The association is defined in terms of a probability level. These statistics-based methods lead to grayer definitions, but the results are often more realistic. In many cases, people or items will fall into middle-of-the-road categories and be defined in terms of characteristics of multiple groups.

Figure 5.2 shows a more complex and more realistic version of two clusters. Clusters often contain significant overlap. Some clustering tools will force a point to belong to exactly one cluster. Other methods assign a probability to each point for each cluster. Points that are highly separated will have a higher probability of belonging to a single cluster (say 85 percent). Points that are within the overlap area will have almost equal probabilities (say 55 percent) of belonging to a cluster. In some ways, this approach is more realistic. Think about asking a person which group he or she prefers. Many people will classify themselves as being in the “middle,” where they feel the ability to select either group depending on other circumstances.

Most of the clustering tools use automated **unsupervised learning** to obtain the results, so the process of performing clustering analysis is relatively straightforward. However, interpreting the results can be challenging. Also, different

**Figure 5.2**

Overlapping clusters. Each point has a probability of belonging to each cluster. The sample numbers are for demonstration, they were not computed. But points that are more highly separated will have a higher probability of belonging to one cluster. Points with more overlap will have more-equal probabilities.



tools, diverse measures, and the choice of attributes can all lead to different cluster definitions. Another potential problem is that some of the tools require long processing times as the number of dimensions (attributes) increases. Analysts and business managers need to focus on the ultimate question of whether the resulting clusters are useful and provide meaningful information that can be used to increase profits. Often, the analysis requires exploration and experimentation with various attribute combinations and methods until the results make sense and provide valuable information.

## Business Situation

---

**Why would a business use clustering analysis?** Clustering is useful for any situation where objects or events are defined in terms of attributes and there are useful reasons to identify groups of objects by those attributes. Certainly, customer groups are the most common example in business. But customers are not the only useful application. Any collection of people or companies would be a candidate for clustering analysis. Many attributes exist on employees that could provide useful classifications. For example, employee evaluations, job training, team memberships, specific skills, years of experience, and so on might be used to cluster or categorize employees. How those clusters are used depends on the situation—from promotions to salaries to teamwork assignments. Similarly, suppliers can be evaluated on attributes of timeliness, quality, pricing, and associated measures. Of course, clustering is valuable only if there are many members in the original group. Also, as you will see, sometimes clustering produces only a small number of obvious groups.

Beyond people, clustering can also be useful in various aspects of production. For example, products might be grouped already into categories, but are those the correct categories? Perhaps products have changed over time and some items were simply thrown into categories and production runs based on a hasty decision that was convenient or made sense at the time. Cluster analysis based on item attributes can more precisely define which products are similar and dissimilar. Likewise, entire production lines and factories could be evaluated and grouped according to various measures, such as speed, quality, and cost. Remember that the key to obtaining useful results is to use attributes that match the goals of the problem.

Every area in business can define objects in terms of attributes and benefit from identifying clusters. Clusters are used to evaluate financial investments using common attributes such as term, risk, and return (or price). Accountants might group fixed assets or other cost structures. The legal department might look for clusters of cases. Beyond customers, marketing might examine customer complaints or problem reports on products. MIS could find groups of similar users to identify software and support needs. The technique can also be used to evaluate projects or even security threats.

Keep in mind that clustering has also been used successfully in several science disciplines. If the organization conducts research or production that requires scientific observations, clustering can be a powerful tool for solving problems.

Clustering has other important uses in data mining. Many tools encounter difficulties in estimating models when the number of attributes (dimensions) is too large. Clustering is a useful method for identifying which collections of attributes are the most important. The problem is simplified by creating a small number of clusters that identify different groups. These clusters can be used as dimensions—



instead of the dozens or hundreds of raw attribute values. As a side note, clustering is also used in some data-compression systems. By finding similar clusters of pixels, images can be reduced in size by storing the cluster data instead of the raw data. These methods are lossy and the quality degrades somewhat but clustering results in substantial space savings.

## Model

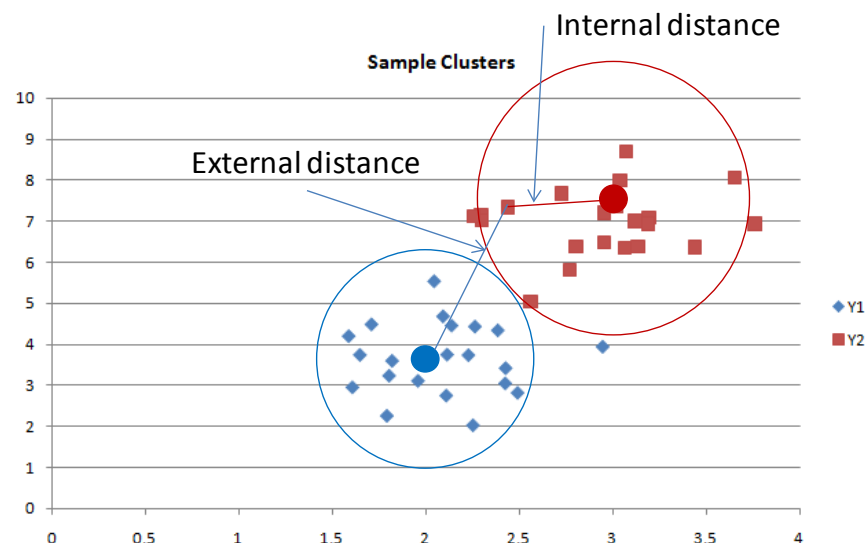
### How does clustering work and how are the results interpreted?

The question of how clustering works is somewhat complicated because several versions of clustering tools exist. The differences are significant not just variations in search algorithms. The overall goal is the same: Find items that fall into groups with similar attributes. But the fundamental differences influence the interpretation of the results. Additionally, the definition used to measure distance can vary even within the same tool, changing the results.

Three types of clustering tools are examined in this chapter: (1) Combinatorial, exemplified by the K-means algorithm, (2) Statistical exemplified by the EM mixture algorithm, and (3) Hierarchical. Hierarchical clustering can be applied to most methods but it deserves special consideration. Other specialized clustering methods exist, including the patient rule induction method (PRIM) which seeks groups with the highest frequencies, and statistical models such as factor analysis and principal components.

**Figure 5.3**

The importance of distance in clustering. A point is placed into a cluster when the attribute distance with the center of the cluster is small relative to the distance to other clusters.



## Distance or Dissimilarities

**Distance** is a key element in clustering. The definition alone indicates the importance—an item is placed into a cluster where it is similar to other items in that cluster and dissimilar to items in other clusters. Figure 5.3 shows the basic concept. Assume two potential clusters exist and the system needs to determine where to place a new point. Compute the distance from that point to the center of each cluster. The point should fall into the closest cluster. The processes for identifying the number of potential clusters and where they should be centered are different questions that need to be answered. But, the concept of distance measures on attributes is critical to all clustering methods.

Measuring distance is a classic issue in mathematics and statistics. Clustering complicates the problem in a couple of areas. One important area is that distance needs to be measured across multiple attributes. The standard approach is to treat the attributes independently and simply sum the difference measures. With a total of  $p$  attributes, the distance from one point ( $x_i$ ) to another ( $x_k$ ) is the sum of the distances of each attribute:

$$D(x_i, x_k) = \sum_{j=1}^p d_j(x_{i,j}, x_{k,j})$$

Some systems weight the attributes (multiply by  $w_j$ ), and the weights often add to one. In most cases, it is better to avoid using weights. The goal is to find the natural clusters and applying weights distorts the value of the individual attributes, hiding natural differences. Of course, it is still necessary to define the distance measure for each attribute value ( $d_j$ ). The most common approach is to use a **Euclidean** measure—square the difference of the values:

$$d_j(x_{i,j}, x_{k,j}) = (x_{i,j} - x_{k,j})^2$$

The squared-difference is used because (1) it ensures values are always positive, and (2) it is mathematically convenient because it is differentiable and the derivative is continuous. On the other hand, large differences are treated more importantly than small differences. Because of the exponential (squared) term, large differences quickly dominate the results. This approach might cause conceptual problems in some applications. It is possible for one or two observations to dominate and define entire clusters. Consequently, more recent tools offer the use of absolute value to define the difference:

$$d_j(x_{i,j}, x_{k,j}) = \text{abs}(x_{i,j} - x_{k,j})$$

With absolute value, the measure is always positive, and larger differences are still important. However, larger differences are not exponentially more important. An outlier can still influence the location of a cluster, but the effect is not as dramatic. Should all systems switch to using absolute value? The major drawback to absolute value is that it is more difficult to handle mathematically, so the computations and cluster search are slower than with the Euclidean definition. Yes, computers are fast today, but clustering problems can require huge amounts of processing time as the number of clusters and number of observations increase. And, for many problems, the effect of outliers is relatively weak. Still, if a tool provides the option to measure in linear instead of squared differences, it is worth compar-

ing the results. If the clusters change substantially, the next step is to investigate the outliers to see which measure handles them the best for the specific problem.

Several other measures have been proposed, and some systems provide choices of five or more distance measures. It can be difficult to choose among them. Unless there is a specific objective to the search that closely matches a different measure, it is best to stick with the traditional squared differences, linear/absolute differences, or possibly the **correlation coefficient**:

$$\rho(x_i, x_k) = \frac{\sum_j (x_{ij} - \bar{x}_i)(x_{kj} - \bar{x}_k)}{\sqrt{\sum_j (x_{ij} - \bar{x}_i)^2 \sum_j (x_{kj} - \bar{x}_k)^2}}$$

The means in the formula are computed across the attributes for a single observation. The correlation coefficient incorporates interaction effects among the attributes through the multiplication in the numerator. When the attributes are not independent but combine to produce unique differences, the correlation coefficient is a more accurate measure of the distance between observations.

### *Ordinal Attributes*

The basic distance definitions assume that the attribute data is continuous. The standard distance measures only work with real-valued attributes. Yet, business objects often include discrete attributes. It is possible to define distance measures in these cases, but there are some drawbacks. One type of attribute that is straightforward is an **ordinal measure**: 1, 2, 3, and so on. Perhaps items are ranked or a survey response consists of 5-levels, such as the Likert scale: 1=strongly agree, 2=agree, 3=neither agree nor disagree, 4=disagree, and 5=strongly disagree. These attributes arise relatively often and can be converted to continuous measures by dividing by the highest value:

$$\frac{i-1/2}{M}, i=1,2,\dots,M$$

The one-half term is a standard correction to center the interval. However, this particular conversion does not seem to be popular in terms of availability. Some systems might simply interpret the raw data as continuous or treat it as categorical data. Yet, if the data is known to be an ordinal measure, this transformation is the best approach. To ensure it is applied, you might have to build a query and compute the new column using SQL.

### *Categorical Attributes*

How is it possible to measure distances between **categorical attribute** values such as “Male” and “Female?” Perhaps for gender, the system could use the physical distance between Venus and Mars, but that number would be large compared to other attributes, so gender would dominate any combined measure. The example seems facetious, but it highlights the problem with categorical data—any measurement system is arbitrary. The key is to create one that adds the least amount of distortion. Categorical attributes are often called **nominal** dimensions because the distance values are assigned arbitrarily.

The simplest approach is to categorical data is to define a square matrix that lists all possible values of the categorical attribute. Values on the diagonal rep-

Distance	Female	Male
Female	0	1
Male	1	0

Distance	Dept A	Dept B	Dept C	Dept D
Dept A	0	1	1	1
Dept B	1	0	1	1
Dept C	1	1	0	1
Dept D	1	1	1	0

**Figure 5.4**

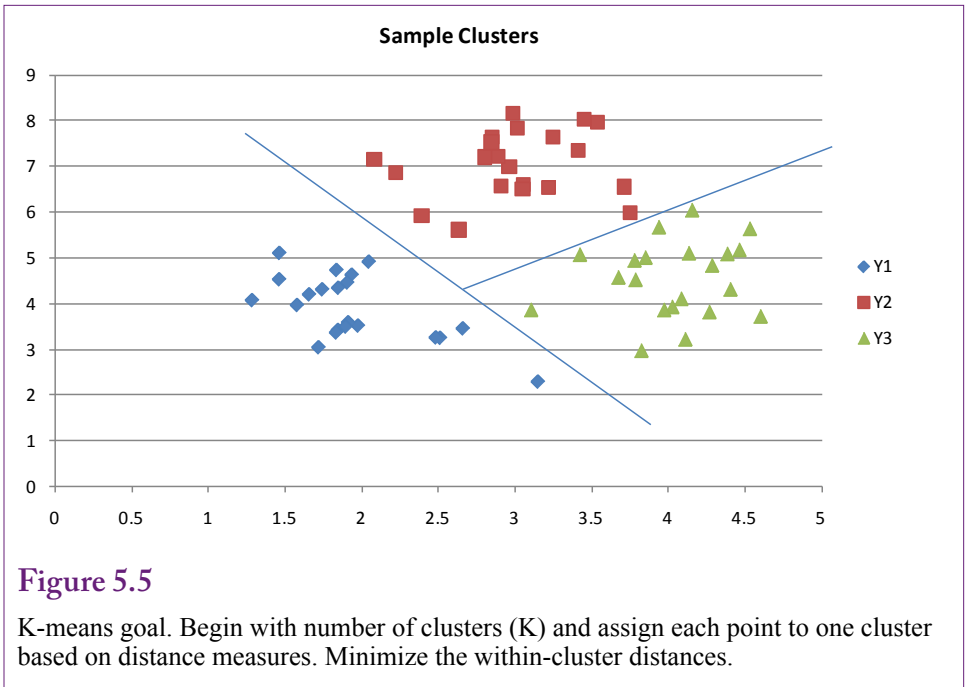
Distance tables for categorical data. When attribute values are equal the distance is zero. Off the diagonal, the distance is one. This choice works well for two attributes. For larger value sets it might be possible to assign different weights but they would be based on subjective values.

resent matching attribute values, so the distance is zero. Values off the diagonal could have different weights, but the most neutral is to assign a value of one to each difference. Figure 5.4 shows two examples. The gender matrix with only two values is the simplest. Choosing 1 for the off-diagonal distance does not affect the results because it could be scaled without affecting any interrelationships. The larger problem with four departments could be more complicated. Choosing 1 for every off-diagonal distance is the most neutral, but it might not be accurate. It is conceivable that the various departments are highly dissimilar. Perhaps C and D are more similar than C and A. But, the computer has no way of knowing these relationships. If the differences are important to the problem, the appropriate paired-distances could be altered to reflect the dissimilarities; but these changes are subjective and need to be made manually. If a particular tool does not support entering the matrix values by hand, the same effect can be created by assigning numbers to the categories and basing the cluster analysis on the numbers instead of the categories. This process effectively converts the categories directly into continuous data that reflect the subjective distances.

The point is that analysts and managers need to think about the data attributes—particularly for categorical data. If each category is roughly equal to the others, the default distance measures used by the clustering tools will suffice. If managers are aware of additional information that differentiates the values, the category should be converted into numerical data that reflect these valuations.

### Combinatorial Searches with K-Means

**Combinatorial search** methods begin with a target of finding  $K$  clusters, hence the reason for calling it **K-means**. The mean is the center of each cluster. The goal is to find the best way to split the data to assign each point to exactly one cluster. In one sense, the search method attempts to compare all possible combinations of points into each cluster to determine the best groupings. Of course, with any reasonable-sized problem, it is impossible to test every possible combination. Hence, the algorithms find shortcuts that reduce the number of comparisons. Figure 5.5 illustrates the basic result. The objective is to minimize the total within-cluster



**Figure 5.5**

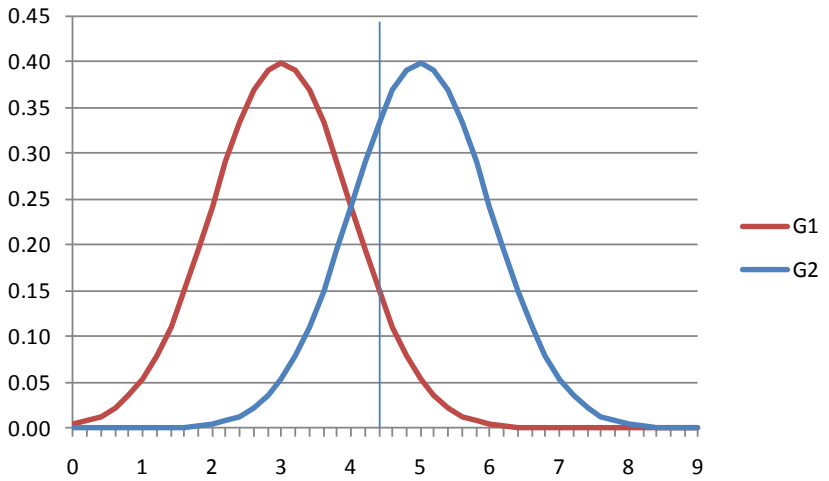
K-means goal. Begin with number of clusters ( $K$ ) and assign each point to one cluster based on distance measures. Minimize the within-cluster distances.

distances. The Euclidean squared distance measure is most commonly used. The algorithm is iterative and begins with the number of desired clusters. The system is initialized with  $K$  prospective clusters—typically defined in terms of the means or central points. From the starting point the routine follows two basic steps:

1. For given cluster assignments, minimize the total cluster variance to find the central point or mean with respect to all  $p$  attributes.
2. Given the current set of means, assign each of  $N$  points to a cluster so that the within-cluster Euclidean distance is minimized.

The process repeats the steps until the points no longer shift to new clusters. The purpose of listing the steps in this chapter is to highlight a few important elements of the process. The most obvious question is how to set the number of clusters ( $K$ ). A related question is how to set the starting point for each of the clusters. A much subtler issue is that the search method cannot guarantee that the best set of clusters is found. The process finds a local optimum, but, particularly with many attributes, the local optimum can be considerably different from the global optimum.

The question of setting the number of clusters ( $K$ ) has a couple of answers. Sometimes there is a business reason for choosing a specific value. Perhaps managers already believe a certain number of clusters exist. Or, in the classic case, there are  $K$  salespeople and clustering is used to identify  $K$  groups of similar customers so that each salesperson deals with a similar group of clients. Alternatively, a heuristic method is often used to automatically select the number of means. The system starts with  $K=1$  and computes a measure of the within-cluster distances. It then tests for  $K=2$ , up to  $K_{\max}$ . The within-cluster distances will decrease as  $K$  increases. In the extreme situation, if  $K=N$  (the number of observations) then

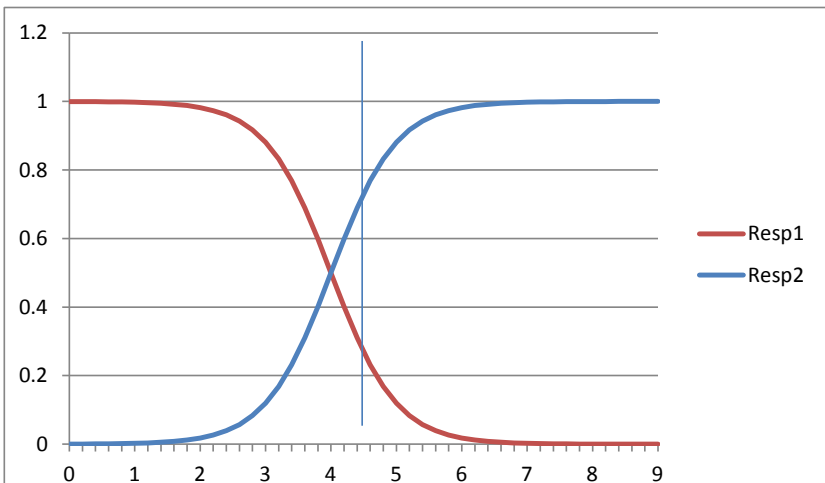


**Figure 5.6**

Statistical model of clusters. A simple example on one dimension with two potential clusters. The two clusters have different means but the same standard deviation. Consider a point to be classified sitting at 4.2 as shown by the vertical line.

**Figure 5.7**

Responsibility functions. They are the relative density functions:  $g_0/(g_0+g_1)$ . It is now possible to read the responsibilities of each cluster for the specified point (0.31 and 0.69).



every point belongs to a separate cluster and the distance is zero. However, somewhere along the way from 1 to  $K_{\max}$ , the system should hit the “natural” number of clusters. Moving beyond this point simply subdivides the natural clusters and there is little gain. Plotting the distance total on a chart against  $K$  should reveal a kink at the point where the natural number of clusters is exceeded. Tools can automate this process by computing a **gap statistic** that numerically identifies the break point and selects  $K$ . It is not a perfect measure, but it does provide an automated solution.

The question of starting points also lacks a definitive answer. Some systems randomly choose starting points. Others begin with one cluster point and add a new one by minimizing the distance measure assuming the other points are fixed. This process repeats until all  $K$  initial points are set. In either case, the choice can be automated, and is rarely altered by the analyst. Additionally, some tools experiment with different starting points. The goal is to find solutions from a variety of starting points. By running the algorithm from multiple starting points, it has a better chance of finding a global optimum.

Overall,  $K$ -means is a popular method for identifying clusters. It can be automated and run with minimal help from the analyst. The analyst chooses attributes, possibly redefining categorical variables to incorporate subjective knowledge. A key point is that the  $K$ -means algorithm always assigns each point to a single cluster. These assignments are based purely on minimizing the variation within each cluster, and the algorithm assigns observations by comparing various combinations.

## Statistical Mixture Model with EM

The  $K$ -means algorithm assigns points to a cluster by attempting to find the cluster that fits best with each point. A different way to approach the problem is to assume that the underlying population consists of unseen clusters that follow some probability distribution. Each cluster has its own distribution, and any observed item must have come from some combination of these distributions. Essentially, the method assigns a probability to each point for belonging to each cluster. The distributions still evaluate distance measures on the attributes. A **mixture model** defines a linear combination of the probability functions, where the density functions are combined with weighted averages and the weights sum to one:

$$f(x) = \sum_{i=1}^K \alpha_i G(x, \mu_i, \sigma_i)$$

Typically, the distributions are assumed to be Gaussian (normal), and each cluster can have a different mean and standard deviation. To understand how this process differs from  $K$ -means and how it affects the results, it is easiest to diagram a simple problem with a single dimension (attribute) and two possible clusters.

Figure 5.6 shows the distributions for two clusters. The distributions are both Gaussian and have the same standard deviation (1.0) but different means or centers. The point to be classified is at 4.4 as shown by the vertical line. This point appears to be most strongly within the second cluster, but there is still a fairly high probability that it belongs to the first group. Instead of assuming that the point must belong to only one group, the mixture model defines a mixture parameter that indicates the association with both clusters. The main purpose of the method is to identify the value of this mixture parameter for each point and use these values to determine the means and standard deviations of each cluster distribution.



Start with initial mixture value ( $\pi = 0.5$ ), and initial distribution parameters for each cluster.

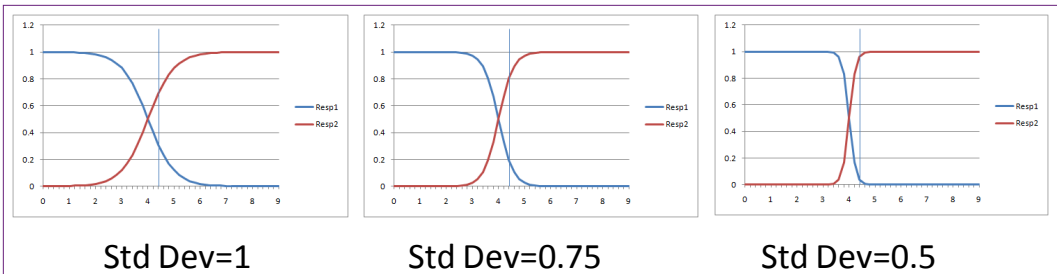
$\hat{\gamma}_i = \frac{\hat{\pi}G(\hat{\mu}_2, \hat{\sigma}_2, y_i)}{(1-\hat{\pi})G(\hat{\mu}_1, \hat{\sigma}_1, y_i) + \hat{\pi}G(\hat{\mu}_2, \hat{\sigma}_2, y_i)}$	Compute responsibilities for each observation and compute parameter.
$\hat{\mu}_1 = \frac{\sum_{i=1}^N (1-\hat{\gamma}_i)y_i}{\sum_{i=1}^N (1-\hat{\gamma}_i)}$	Use parameters to estimate mean and variance for cluster distributions, and new estimate of mixture value.
$\hat{\sigma}_1^2 = \frac{\sum_{i=1}^N (1-\hat{\gamma}_i)(y_i - \hat{\mu}_1)^2}{\sum_{i=1}^N (1-\hat{\gamma}_i)}$	Repeat until parameters are stable.
$\hat{\mu}_2 = \frac{\sum_{i=1}^N (\hat{\gamma}_i)y_i}{\sum_{i=1}^N (\hat{\gamma}_i)}$	
$\hat{\sigma}_2^2 = \frac{\sum_{i=1}^N (\hat{\gamma}_i)(y_i - \hat{\mu}_2)^2}{\sum_{i=1}^N (\hat{\gamma}_i)}$	$\hat{\pi} = \sum_{i=1}^N \hat{\gamma}_i / N$

**Figure 5.8**

EM Algorithm for two clusters. Step one uses expectations to compute responsibilities. Step two, maximization, computes the mean and standard deviation of the cluster distributions, along with the mixing parameter.

Instead of working with the probability density functions, the methods define responsibility functions. In the simplest form, **responsibilities** are the relative density functions:  $g_0/(g_0+g_1)$  and  $g_1/(g_0+g_1)$ . Figure 5.7 shows the responsibility functions for the small example. When  $x=4.4$ , the responsibilities are 0.31 and 0.69. If the observed point falls further to the right, the responsibility for cluster 2 increases to 1, and the point would be classified as completely within the second cluster.

The **expectation maximization (EM)** algorithm uses this statistical foundation to determine the means and standard deviations of each cluster and to assign points to the clusters. Figure 5.8 outlines the basic steps for a two-cluster model. Similar to the K-means algorithm, EM begins with starting values of the means and standard deviations of the clusters. It also needs a starting value for the mixture parameter for each point, but this value is commonly set to 0.5 as an unbiased starting point. Once initialized, the responsibilities are computed for every observation at the expectation stage. These values then update the means and standard deviations for the clusters, and define the overall mixture parameter. The process repeats until the estimates stop changing. The mathematics and optimization are more complex with more than two clusters, but the concepts are the same.



**Figure 5.9**

The effect of standard deviation on responsibilities. As standard deviation approaches zero, the separation of the responsibilities increases and each point is associated more closely with a single cluster.

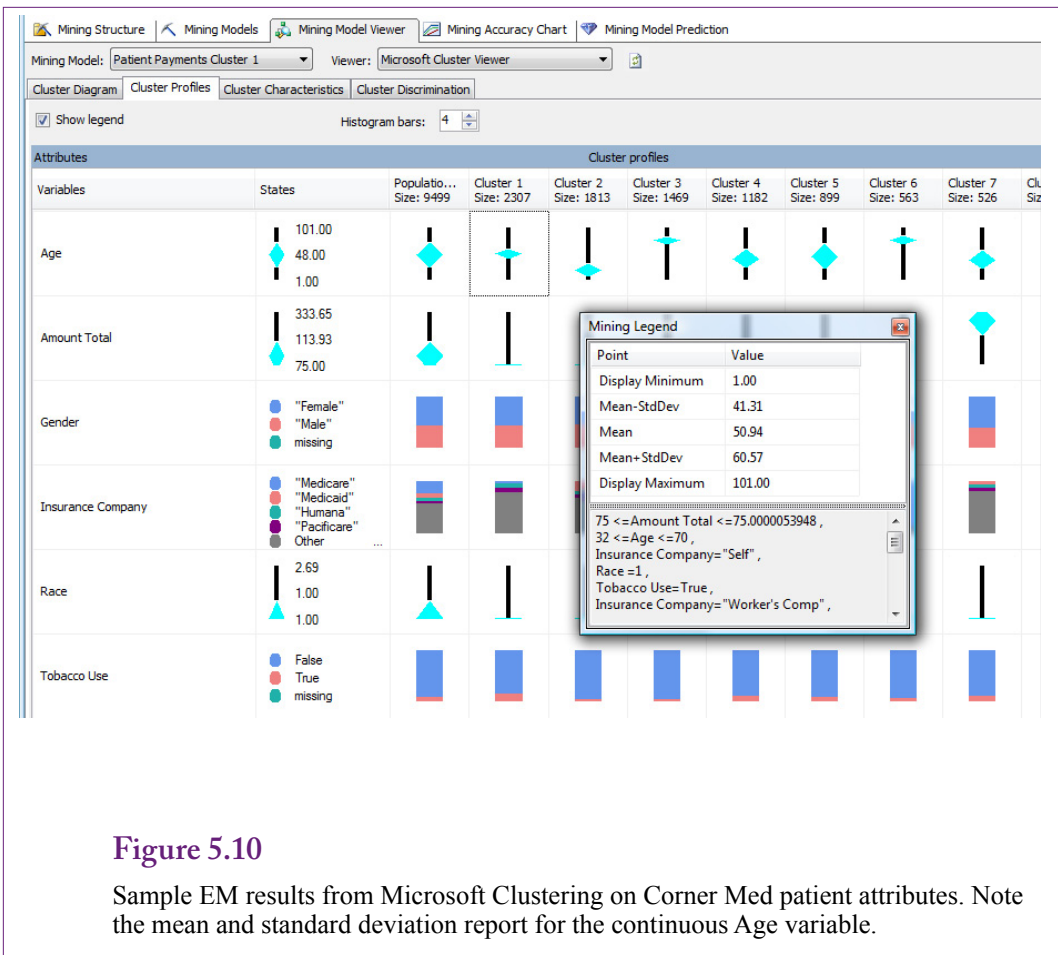
There is no reason to memorize the equations, but they highlight the role of statistics in the process. In particular, note that the clusters are defined by means and standard deviations. The standard deviations play an important role in understanding the results. Figure 5.9 shows the one-dimensional example with three levels of standard deviation, which is set to the same level in both distributions. As the standard deviation approaches zero, the responsibilities become more rectangular and each observation is associated more closely with a single cluster. Conversely, when the standard deviation is high, the responsibility values approach 0.5. At zero, the EM method is essentially equivalent to the K-means approach and each observation is assigned to a single cluster.

When interpreting the results, the primary difference with the EM approach is that observations can be associated with more than one cluster. It is a softer association than with K-means. But, unless you look at individual observations, this effect is somewhat hidden. A more obvious difference is that clusters are defined by means and standard deviations, which are reported for attributes with continuous measures. Figure 5.10 shows a portion of the results from applying Microsoft's Clustering EM tool on the Corner Med patient attributes. One cluster definition is highlighted in terms of the Age attribute. Notice the use of the mean and standard deviation. Also, notice that Gender seems balanced across the clusters, so Gender is probably a weak classifier. This issue is covered in more detail in the Microsoft results section.

The EM algorithm faces the same basic issues as the K-means algorithm. The number of clusters must be determined in advance. The starting means and standard deviations of the clusters must be specified. Because of this second issue, EM converges to a local optimum, which might not be the best global solution. Most algorithms use processes similar to those for K-means for choosing starting points. Testing various starting points can also be used to help find better solutions.

### Hierarchical Clusters

The question of number of clusters or means can be difficult to solve. Each tool uses different methods, so the final results are different depending on the tool. If necessary, the results can be made similar by forcing tools to use a specific number of clusters, but it does not solve the problem of determining the best number

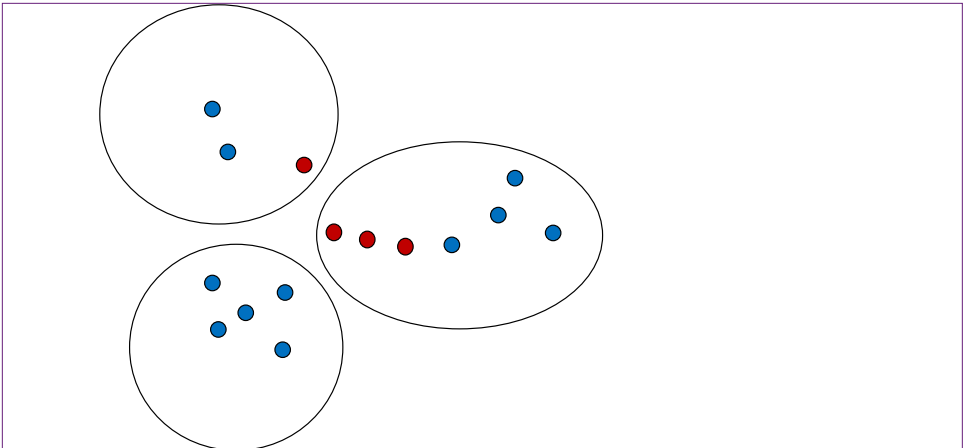


**Figure 5.10**

Sample EM results from Microsoft Clustering on Corner Med patient attributes. Note the mean and standard deviation report for the continuous Age variable.

of clusters. One potential answer is to use hierarchical clustering. With **hierarchical clustering**, the tools start at the top (all data in one cluster) and work down; or they start at the bottom (each observation is a separate cluster) and work up. Either way, the tools create a collection of clusters for all possible values of  $K$ .

Several variations exist of both hierarchical clustering methods. Consider the top-down approach first. Top-down hierarchical clustering is usually called **divisive** because the top cluster contains all of the data observations. At each level, the algorithm examines each existing cluster and divides it into two new clusters. The most interesting approach looks at the cluster to be split and finds the one observation that is the most dissimilar or whose average distance is farthest from the center. This observation becomes the center of the new cluster. The process then looks among the remaining elements to compute the average distance from the original cluster to the new cluster. The element with the greatest net average distance is moved to the new cluster. The process repeats until net gains no longer exist. Divisive hierarchical clustering can continue to the end where each observation is in a separate cluster, or it can be cut off early once a specified number of clusters has been reached. For this reason, divisive clustering is useful when it is important to hold the total number of clusters to a relatively small number. Analysts could choose the maximum number of clusters, or they could examine the re-



**Figure 5.11**

Single linkage problem is large-diameter clusters. Some items within a cluster are not very close to the others. Problems arise because only the smallest distance is used and items can chain their way into an inappropriate cluster.

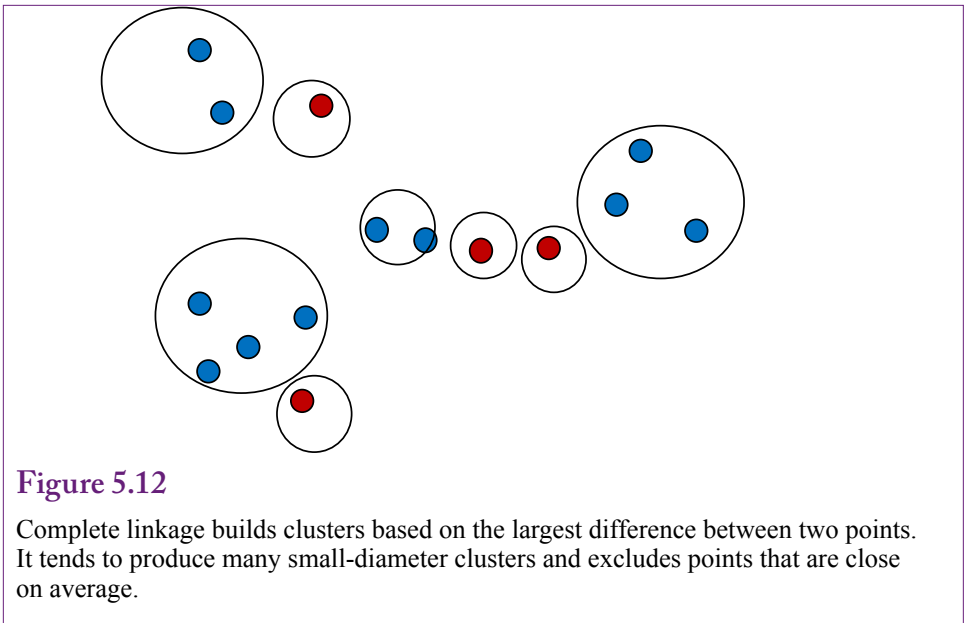
sults at each step and decide if the gains from one more split are important enough to justify a more complex result. The technique is often used in data compression systems, where the goal is to obtain a close approximation to the data with as little storage (fewer clusters) as possible.

Hierarchical clusters can also be built from the bottom—starting with each observation in its own cluster. At each step, the **agglomerative** algorithm finds the two most similar clusters and combines them to form a new cluster at a higher level. The trick is to find the “most similar” clusters. This step requires a distance measure of dissimilarity between clusters. Earlier measures were defined between a single point and a cluster. Agglomeration can use the fundamental distance measures but it needs to apply to each point in the two clusters. Several measures have been defined, including single linkage, complete linkage, and group average. All methods compare each point in the first group (G) to each point in the second group (H), so the distances are pair-wise measures. The average simply computes each pair-wise distance, adds them up, and divides by the total number of pairs ( $N_g N_h$ ).

Single linkage selects the smallest distance between any pairs (nearest neighbor):  $\min d(c_1, c_2)$ . The single linkage, smallest distance, measure tends to create clusters with large diameters. A point needs to be close only to one other point in the cluster. Figure 5.11 shows that points can enter a cluster by chaining onto a near neighbor. Once one point is in the cluster, then another close one enters, and the rest fall as part of the chain. But, on average, the points at the end of the chain can be considerably different from the other points at the far end of the cluster.

Complete linkage selects the largest distance between any pairs (furthest neighbor). If the data clusters are well separated, all three measures should produce about the same results.

Complete linkage tends to produce clusters with small diameters, and can exclude observations that are “close” in average distance. The points within a cluster are all closely related, but the measure tends to exclude points that are also close

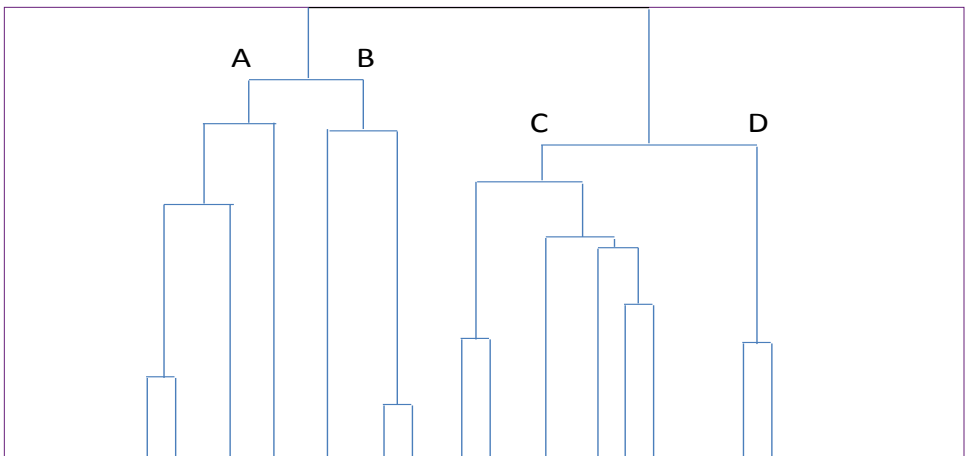


on average and should probably be included. Figure 5.12 shows the problem. The points within the clusters are all close, but some nearby items are being excluded because the distance to one point within the cluster is too large. Ultimately, as the agglomeration continues, the points will be incorporated into the clusters, but the final cluster definitions can be different from those generated by other methods.

On balance, the group average measure presents a balance between the two extremes. It tends to produce relatively compact clusters with relatively good inclusion of close items. It also has nice mathematical properties that relate well to the statistical definition of clusters. But it is not a perfect measure. Because all attributes are numerically averaged, the distance measure is subject to the scale of the numbers. For example, one attribute might be measured in relatively small values from 0 through 100. A second attribute (such as income) could be measured in tens of thousands. This higher scale is going to dominate the effect of the other attributes. And, if the scale is changed (convert the income to thousands), the distance measure changes and the resulting clusters will be different.

Why should you care about the different methods? The answer is that the tools that support these methods have options. As an analyst, you have to select the options that most closely match the problem being investigated.

As shown in Figure 5.13, hierarchical clusters are sometimes displayed on dendrograms. A **dendrogram** is a compressed display of the clusters created at each level of the hierarchical clustering process. The height of each node (cluster) is proportional to the dissimilarity of its children. In the example, the dissimilarity between C and D is larger than it is between clusters A and B. Large dissimilarities are generally good because they indicate the need for different clusters. If dissimilarities are low, there is little need to split into new clusters. The dendrogram presents an interesting picture of how the clustering behaves at each level. However, keep in mind that the actual clusters, and the dissimilarities, are highly dependent on the distance measure chosen. Hence, the dendrogram is not a picture of the underlying data—merely a picture of the clustering choices. Some systems



**Figure 5.13**

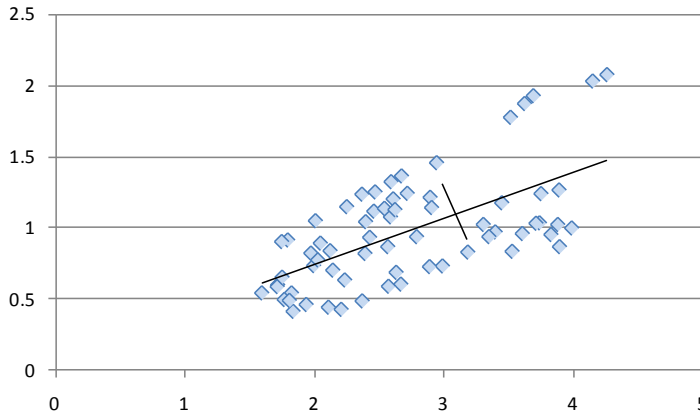
Dendrogram example. The diagram shows all of the clusters created at each level. The node's height is proportional to the dissimilarity between the node's children. So C and D are more dissimilar than A and B.

always draw the ending lines to the bottom of the chart. Some provide color capabilities to highlight higher-level clusters. In the example, everything below cluster C could be one color and below D would be a second color—indicating two of the final clusters chosen.

### Other Statistical Methods

One of the main purposes of clustering is to reduce the number of dimensions or even the number of data points in a problem. Several other tools perform related tasks that can also be used to reduce the dimensions of a large problem. Many of them were designed for specific tasks, such as latent variables, exploratory projection pursuit, self-organizing maps, and multidimensional scaling. These techniques are useful for some data reduction problems but are not covered in this book. However, one older statistical technique is commonly used in data mining tools. **Principal components analysis (PCA)** is a technique used to reduce the number of dimensions or attributes in a problem. The method searches for a smaller number of components that are linear combinations of the existing data that can approximately match the original data. It is particularly useful when the attributes have a high degree of **multicollinearity**. When attributes are closely related, most statistical methods have difficulties separating the effects of the variables and often cannot find a good solution. PCA defines the internal correlations and uses them to create a smaller number of variables that define the same output data.

Figure 5.14 shows a small example with attributes X1 and X2. The first principal component is a vector that identifies the direction that contains the highest amount of variation. It is shown as the longer line in the diagram. The second component is **orthogonal** (perpendicular in two dimensions) to the first vector. It is substantially shorter than the first component because the data points have less variation in that direction. This set of data could possibly be reduced to a single dimension by projecting all of the points onto the first principal component. It



**Figure 5.14**

Sample principal components in two dimensions. The first component identifies the direction of most variance ( $X_1$ ). The second is smaller and orthogonal.

would incorporate variations from both the  $X_1$  and  $X_2$  dimensions (axes). The results would be an imperfect representation of the data. Some of the variation would be lost. The amount of information lost is visually seen by the length of the shorter line. If it is relatively short, less information is lost by projecting the points onto the longer line. With each problem, the analyst needs to choose how much variation should be lost. No fixed answer exists, but the tools do provide information to help make the decision.

The mathematics behind PCA are straightforward but require knowledge of linear algebra (matrices). It is probably unnecessary for business analysts to understand all of the mathematics, but some of the commonly-used terms come from linear algebra. One way to approach the problem is to define the matrix  $\mathbf{X}$  which contains the  $N$  rows of data consisting of  $p$  columns of attributes. These values are centered for each attribute by subtracting the mean from each observation. An efficient method for finding the principal component vectors is to decompose the  $\mathbf{X}$  matrix into products of three new, specially-constructed matrices:

$$\mathbf{X} = \mathbf{U} \mathbf{D} \mathbf{V}'$$

Matrix  $\mathbf{U}$  is an  $N \times p$  matrix such that  $\mathbf{U}' \mathbf{U} = \mathbf{I}_p$ , the identity matrix. Similarly,  $\mathbf{V}$  is a  $p \times p$  matrix where  $\mathbf{V}' \mathbf{V} = \mathbf{I}_p$ .  $\mathbf{D}$  is a diagonal matrix where all elements are zero, except those on the main diagonal, which are sorted in descending order. This factorization of  $\mathbf{X}$  is known as the singular value decomposition (SVD), and it is a commonly-used tool in statistics and numerical analysis. Numerical analysis is the branch of mathematics that deals with using the computer to solve mathematical problems. The point of the equation is that the columns of  $\mathbf{UD}$  are the principal components of  $\mathbf{X}$ . SVD is particularly useful for finding the coefficients in a linear regression model. Consequently, efficient computer algorithms exist for finding the principal components.

Many data mining and most statistical packages have the ability to perform principal components analysis. It is not necessary to know the underlying mathematics. Figure 5.15 shows the results of running PCA on a small sample dataset



eigenvalue	proportion	cumulative	
2.87027	0.57405	0.57405	$0.536X3+0.497X5+0.479X2+0.475X1+0.101X4$
1.35739	0.27148	0.84553	$0.826X4+0.414X5-0.337X1-0.137X3-0.116X2$
0.44212	0.08842	0.93396	$-0.824X2+0.459X3+0.285X5-0.163X4+0.05 X1$
0.29819	0.05964	0.99359	$0.811X1-0.391X3+0.301X4-0.278X2-0.147X5$

**Figure 5.15**

Sample results for PCA with five variables (X1-X5). The eigenvalues are used to choose the number of vectors to keep. Every component with an eigenvalue above one needs to be kept. Other components might be useful. Check the cumulative proportion of variation explained. Keeping all four would account for almost 99.4 percent of the variation, so the reduction of one dimension would yield very little loss of information.

with five attributes (X1-X5). The system has identified four principal components that are evaluated in terms of the **eigenvalues**. Technically, a principal component is an eigenvector, and each eigenvector has an association eigenvalue. The terms come from linear algebra and arise from a way of rewriting the problem. The eigenvalues are the key because they measure the variation explained by the matching eigenvector. In any problem, the eigenvalues sum to the number of dimensions (5 in the example). Dividing each eigenvalue by the total gives the percentage of variation explained. Eigenvalues are listed in descending order and most tools report the cumulative proportion. Kaiser's criterion (Kaiser 1960) states that components with an eigenvalue greater than one should always be included. This rule would include the first two vectors. From the cumulative column, those two dimensions together explain 84.6 percent of the variation. Some people might consider a 15 percent loss of information too great of a price to pay for reducing a problem from five dimensions down to two. Ultimately, the decision depends on the problem, the analyst, and how critical it is to reduce the number of dimensions. Continuing down the table, including all four components saves one dimension and covers almost 99.4 percent of the variation—a reasonable tradeoff. As a side note, the results fairly accurately depict the data. The columns were generated using two independent basis columns, with some additional randomness added to each of the columns.

Once the number of components has been selected, the vectors shown in Figure 5.16 are used to compute the new data columns. For example, the new variable V1 is computed as:  $0.475 X1 + 0.479 X2 + 0.536 X3 + 0.101 X4 + 0.497 X5$ . The computation is performed for each data row in the original **X** matrix. Data mining tools that implement PCA can usually generate the new data columns automatically. The selected analysis is then performed on the modified (V1...V4) data columns. One difficulty with this approach is that it can be harder to evaluate and understand the results of the final tool. The end results will be expressed in terms of these principal components. To understand the role of the original variables, the analyst has to trace the impacts through these multipliers. For instance, X1 has a strong influence (0.81) on V4, a relatively strong effect on V1 (0.48), a tiny effect on V3 (0.05), and a negative effect through V2 (-0.34). Results expressed in terms of V1, V2, V3, and V4 have to be interpreted carefully.

V1	V2	V3	V4	
0.4752	-0.3374	0.0499	0.8110	X1
0.4788	-0.1161	-0.8245	-0.2783	X2
0.5363	-0.1366	0.4586	-0.3907	X3
0.1007	0.8260	-0.1625	0.3012	X4
0.4971	0.4145	0.2846	-0.1468	X5

**Figure 5.16**

Principal component vectors. New data columns are created by multiplying each coefficient by the original matching attribute (X1...X4) and adding the terms.

PCA is typically used only if it is required because of huge datasets and problems with too many dimensions. On the other hand, Ding and He (2004) showed that PCA is equivalent to a continuous solution for K-means. The principal component vectors and K-means centers are essentially defining the same concepts. Although, principal components categorizes points in terms of a mixture or percentage instead of forcing each point to a single cluster. The other interesting result of their work is that PCA provides a method to improve K-means searches to help ensure global optima instead of stopping at local points.

## Data

**What type of data is used in clustering?** Clustering relies on a distance measure to determine whether an observation is closer to one cluster instead of others. Distance is more precise when it evaluates continuous variables. Discrete attributes can be used, but the results can be quite different than those from continuous variables. Huge databases can be a problem for some clustering algorithms. The number of observations (N) is an issue, but the number of attributes and clusters is also a problem.

### Attributes and Observations

Clustering algorithms expect observations of data to be stored in a table or matrix where the columns represent different attributes or dimensions and each row contains one observation. This structure matches database tables and queries, so data mining tools based on a DBMS are easy to configure for clustering. Other tools usually read data from text files where each new line contains one row of observations. The data within a row are separated by a unique character—usually a comma or a tab character. Figure 5.17 shows a few rows of data from the Corner Med database. It contains some basic attributes about patients that might be used to identify possible groups of customers. Tobacco use is a binary (true/false) variable. Race is a categorical attribute but it is coded numerically. The Age was computed from the date of birth to a specific point in time at the end of the year. It could have been defined as the age at the time service was provided, but that approach runs into problems for patients who visit more than once a year. The insurance company is the name of the organization that is paying for visits. Technically, it could change during a year as well, but for most patients, it will be constant. The amount is the total amount of service billed for that patient over the year. Notice that each row represents one patient. The database contains thousands of patients but only a few are shown here.

PatientID	Age	Race	Gender	Tobacco	Insurer	Amount
1	58	1	Female	False	Cigna	75
2	2	1	Male	False	United Health	75
3	1	3	Male	False	Cigna	75
4	45	1	Female	False	UniversalCare	75
5	35	1	Female	False	Assurant	150

**Figure 5.17**

Standard data layout. A column represents a single dimension attribute. Each row is one observation, such as a purchase or a customer. Typically, a key column attribute identifies each row.

Data for tables is often stored in simple text files to make it easier to transfer the data to other systems. The **comma-separated values (CSV)** file is one of the most common formats. Most programs and DBMSs can read this format. Files in this format are simple text files, where data for one observation is stored in a single row. The data for each column are usually separated by commas or tab characters.

### Continuous and Discrete Data

Whenever possible, it is best to use continuous data for attributes used in clustering. Most tools can handle categorical data, but the distance measures are limited. Typically, distance is measured as either zero or one. Zero distance is defined if two categories match exactly; otherwise the value is set to one. A few variations exist, but there is no automated method to define a more precise measure.

If the managers and analysts know that subjective differences exist among the categories, then the variables should be recoded into numerical data that reflect these differences. In the healthcare example, managers might want to weight the insurance companies differently. In particular, the Medicare and Medicaid governmental programs are often perceived quite differently from private insurers. Specifically, the government programs place tighter caps on fees and pay lower rates to physicians than most private firms pay. Perhaps the reason for including the insurance companies in the analysis is to consider these payment issues. If so, SQL can be used to recode the data. For example:

```
SELECT Visit.InsuranceCompany,
       InsV =
       CASE Visit.InsuranceCompany
         WHEN 'Humana' THEN 10
         WHEN 'Self' THEN 15
         WHEN 'Medicaid' THEN 1
         WHEN 'Medicare' THEN 1
         WHEN 'Blue Cross/Blue Shield' THEN 11
         WHEN 'Charity' THEN 0
         ELSE 9
       END
FROM Visit
```

Each insurance company is assigned a different number that somehow represents the payment history and value from the company. Only a few companies

are listed in the example. The default condition (ELSE) covers the others, but it would be better to include each company separately for most cases. This approach can be used when these subjective valuations exist. If no one knows appropriate values, then the neutral distance of one assigned to categorical data will have to suffice. It would also be possible to create a new table holding the names of the insurance companies and the new values. SQL would be used to join the tables together on the insurance name.

## Missing Data

Clustering routines do not work with missing observations. Most tools delete observations (rows) that contain missing data. If the dataset contains only a few isolated missing values, this approach is reasonable. However, if one attribute happens to have many missing values, it would probably be better to drop that attribute from the analysis. Otherwise it will cause many rows of potentially useful data to be discarded.

Some tools provide options to replace missing data with new values. These new values can be a constant, an average, or perhaps an interpolated value computed from the two nearest points. For some problems, replacing missing data with the mean is relatively neutral. But, if a large percentage of the observations are replaced, the results may be altered. Many of the clustering tools automatically fill missing data points with the overall mean.

An interesting situation exists with missing data for categorical variables. In many cases, the missing value can be treated as just another attribute value. For example, Gender might contain: Female, Male, Missing. In some case, this interpretation might be difficult to understand. In those cases, filters or SQL queries can be used to drop the rows with missing data.

## Clustering on Products: Cars

---

**How are clusters identified with multiple dimensions?** The simple example of clusters in two dimensions is relatively easy to see—as long as the clusters are relatively distinct. If clusters are weak, with considerable overlap, they can be difficult to see even in two dimensions. Now, imagine what happens in three, four, or more dimensions. The problem gets worse with hundreds, thousands, or millions of observations. Even when the mathematical algorithms identify clusters, they can be difficult for analysts and managers to understand. Software vendors try multiple methods to display clustering results, and these tools represent the major differences between software products.

### Goals

To illustrate the challenges, this section uses real-world data on automobiles for sale in the U.S. in the 2012 model year. The purpose of this data is to illustrate the basic challenge of interpreting clustering results using a product that has multiple attributes that are likely to be recognizable to most students. Two tools are used to illustrate the process and the results: Microsoft Clustering and the free **Weka** data mining software (<http://www.cs.waikato.ac.nz/ml/weka>).

The goal is to determine which vehicles are similar and which are different. For instance, a marketing manager of an automobile company might want to know which cars are the closest competitors. A potential customer might ask the same question. More complex questions could also be addressed, but they generally require more data. For example, policymakers and long-term planners might want

ID	Year	Make	Model	Sec	Category	MPG	Price	Wt.	Cyl.	HP	Seat
769	2012	Mitsubishi	i-MiEV	11.9	Hatch	126	29125	2579	0	66	4
781	2012	Nissan	Leaf	7.9	Hatch	106	35200	3385	0	107	5
595	2012	Chevrolet	Volt Hatch	8.53	Hatch	95	39145	3781	0	149	4
839	2012	Toyota	Prius Plug	10.9	Hatch	95	32000	3165	4	98	5
615	2012	Fisker	Karma	5.9	Sedan	52	95900	5300	0	403	4
838	2012	Toyota	Prius	9.7	Hatch	51	23015	3042	4	98	5
648	2012	Honda	Civic Hybd	5.7	Sedan	44	24050	2853	4	93	5
705	2012	Lexus	CT 200h	10.4	Hatch	43	29120	3206	4	98	5
832	2012	Toyota	Camry Hybd	7.2	Sedan	43	25900	3435	4	156	5
630	2012	Ford	Fusion Hybd	8.7	Sedan	41	28775	3720	4	156	5

**Figure 5.18**

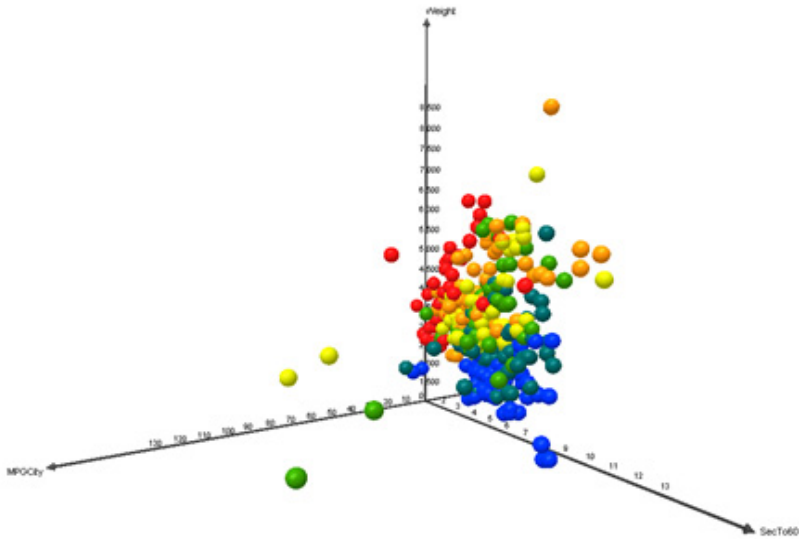
Sample car data. The CarID is random and is used as the key column. Year, Make, Model identify specific vehicles. Category is a human-assigned definition that should not be used in the analysis. The other columns are standard attribute measures. Sec is the number of seconds to reach 60 mph, MPG is miles per gallon in city driving, Price is the list price of the base model, weight is in pounds, Cyl. is the number of engine cylinders, HP is horsepower (bhp), Seats is the number of passenger seats to indicate size. Data was derived from various car Web sites and magazines.

to examine trends in clusters over time—which requires detailed data for multiple years or decades. Similarly, manufacturers would want consumer opinions on similarities and differences, but that information changes the way the question is analyzed and is more suitable for other chapters. From a personal standpoint, you could simply think about the data as an exploration of searching for a new car. The clusters will provide groups of cars that have specific similarities.

## Data

The data consists of a simple CSV text file with 311 data rows. Each car is given a simple ID number to serve as the key. Cars are identified by the Year (2009 for all), Make (Toyota, Chevrolet, Ford, and so on), and Model (Prius, Cobalt, Fusion, etc.). Cars are often assigned to predetermined categories (SUV, Compact, Truck, and so on). This Category variable is included, but it should not be used or it might interfere with other clusters. Measures on the cars include: Seconds to 60 mph—a measure of performance, miles per gallon in city driving—reported by the EPA, price—typically the price of a base vehicle, weight—a measure of size in pounds, number of cylinders—which is likely correlated with performance and mileage, horsepower—a measure of performance, and number of seats—a measure of size.

Figure 5.18 shows a small portion of the data file. The data consists of standard measures on cars and was collected from various car Web sites and car magazines. A few vehicles have missing data. For instance, heavy trucks are not required to report gas mileage. Prices are the least useful because the listed price applies only to the base trim-level with few options. Some vendors use the trim levels to nudge cars into different categories—such as putting a more powerful engine in



**Figure 5.19**

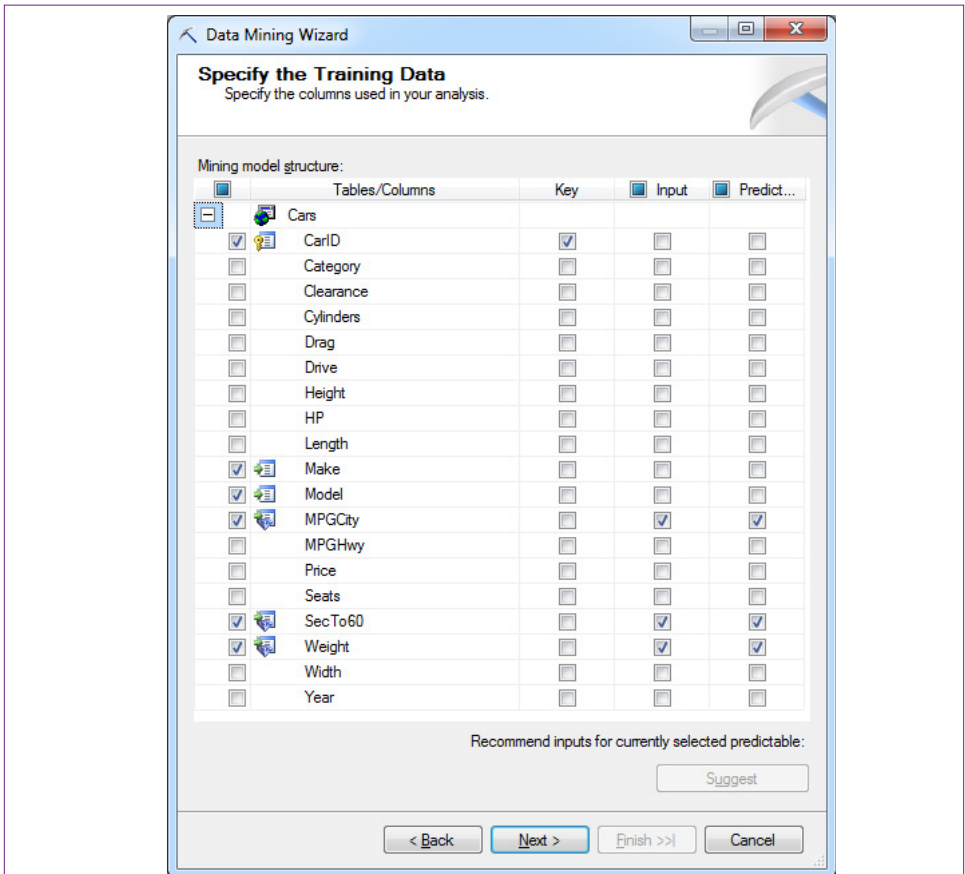
Sample car data in 3D plot. The axes are: MPGCity, SecTo60 (acceleration), and Weight. The color coding is assigned based on list price of the vehicle. Miner3D was used to plot the data, which has nice plots but is a relatively expensive tool. With the tool, the chart can be rotated and zoomed interactively to explore the data relationships.

the higher-trim level to improve performance. But, these changes blur the cluster definitions making the results even harder to interpret so they are ignored here.

Microsoft Clustering requires that the data be imported into a database table. A SQL script is available that creates the table and loads the data from the comma-separated values file. Once the data is in the database, a new Analysis Services project can be created using the Visual Studio Business Intelligence template. Add a **Data Source** by right-clicking the Data Sources entry in the Solution Explorer and choose the New Data Sources option. Follow the prompts to select the correct server and database.

Figure 5.19 shows the car data using a 3D plot which shows relationships across three dimensions. Actually, a fourth dimension (price) is displayed by color-coding the items. The tool (Miner3D) is relatively expensive, but it also supports clustering—in which case the colors can be assigned based on cluster values. In the sample data, the price coding is actually a decent clustering mechanism across the other three attributes (weight, acceleration, and MPG).

A tricky element of the data source is specifying the impersonation option. Obtaining data from the DBMS requires permissions on SQL Server to retrieve the data. Microsoft provides several options to set the appropriate level of security. One option is to set up a specific account on the SQL Server computer, or through Kerberos on a company network. Each person who uses the data source will use that account. A weaker but easier option is to use the built-in service account. This option is the simplest for testing, but weakest security for a production database. The third option is to use the Windows account of the user running the applica-



**Figure 5.20**

Selecting columns for Microsoft Clustering. Include Make and Model but be sure they are not used as Key, Input, or Predictable columns. For now, pick only the three measures: Weight, MPGCity, and SecTo60. Set them as Input and Predictable.

tion. This option enables detailed access control over the data, but requires that the server and client computers be connected through a security system (Windows Active Directory), and that permissions be assigned to each user on the database server. For now, use the service account option but remember to review security if the application is moved into a production environment and shared with multiple users.

The second step is to create a Data Source View with a right-click on the Data Source View entry. A **Data Source View** is typically a collection of tables and named queries. It defines the tables and relationships for data needed for processing. In this case, the data simply consists of the entire Cars table. Note that a Data Source View provides the ability to create named queries, which function similarly to SQL views.

## Microsoft Clustering

Microsoft Clustering defaults to the EM algorithm, but it has an option to run K-means clustering. For the first pass, stick with the default EM algorithm—and



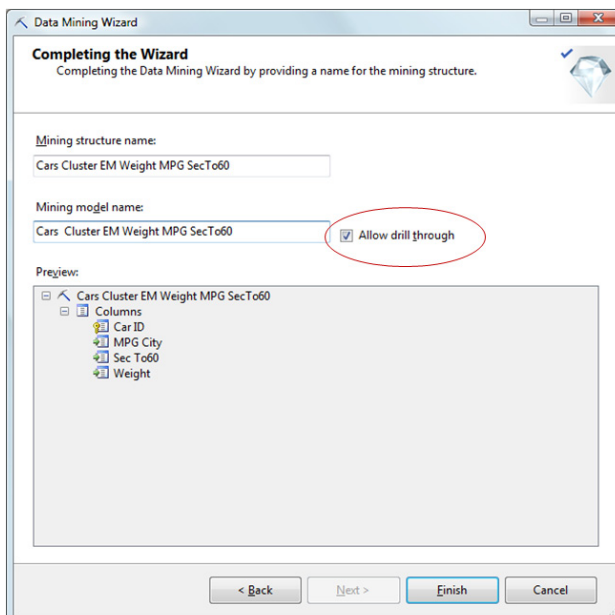
remember that it assigns a mean and standard deviation to each cluster attribute. Points can belong to multiple clusters.

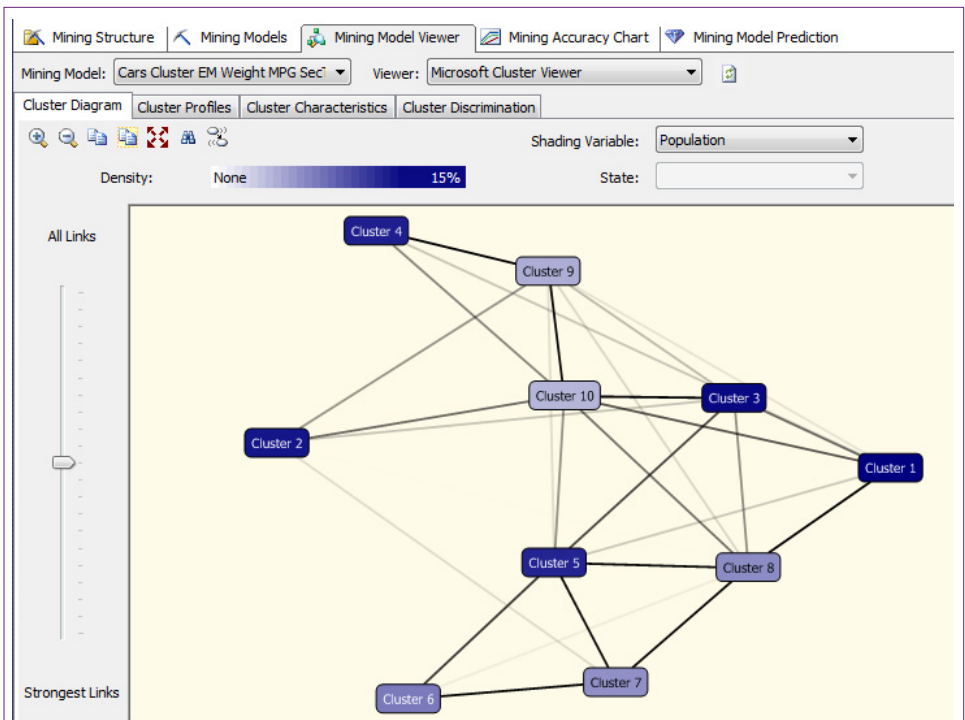
It is straightforward to create the default data mining model. To simplify the initial results, begin with three attributes: MPG, Weight, and SecTo60 time. Right-click the Mining Structures entry in the Solution Explorer and choose the option to add a new mining model. Follow the basic prompts and select Microsoft Clustering as the tool. Use the default data source and Cars table as the Case table. Figure 5.20 shows one of the trickier steps—choosing the columns for the analysis. It should be set by default, but be sure CarID is the only key column. Select the three attribute measures to be used in the clustering as Input and Predictable types: MPGCity, SecTo60, and Weight. Finally, select Make and Model by setting the checkmarks in front of the names. Be careful not to assign them any other roles (Key, Input, or Predictable). They are included here only so that they will be available later for drill-down analysis providing human-recognizable names instead of the meaningless CarID value. For clustering there is often no need to reserve data for testing. With a limited number of observations in this dataset it is best to use all of them. Set the percentage of data for testing to zero.

Figure 5.21 shows the other tricky part of the configuration. Enter a name and description for the analysis that will be unique and recognizable later. Also, be sure to check the box “Allow drill through.” Combined with the Make and Model columns, this option makes it possible to see exactly which vehicles fall into each cluster. Without this check box, the results return only the statistical data. It is much easier to understand results—particularly for products—when the actual

**Figure 5.21**

Setting the drill through option. This option is important for understanding clustering results. It makes it easy to get a list of exactly which products fall into each cluster.





**Figure 5.22**

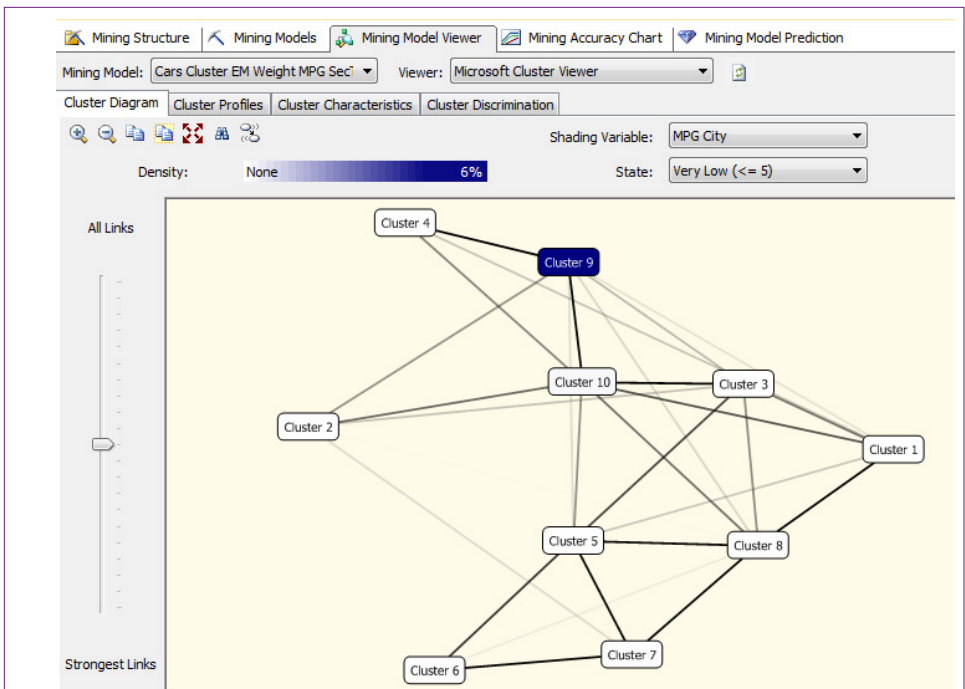
Cluster Diagram. Initially, the clusters are color-coded based on the number of observations—darker implies more data. The links indicate clusters that are close to each other.

values can be displayed for each cluster. Finish the wizard and accept the default values.

Mining models in Microsoft BI need three additional steps: (1) Transfer the model to the Analysis Server, (2) Process the model on the server—run the analysis, and (3) Browse the results. Steps 1 and 2 are often combined. However, for large, time-consuming models, it is possible to transfer everything to the server and then schedule the analysis job to run at a time when the server might be less busy. The models in this chapter are small enough to run in a few seconds. Right-click the new mining model entry in the Solution Explorer list and choose the Process option. Accept the default options on the pop-up screens to upload the data and Run the analysis. When it is finished close the setup forms. Right-click the data mining model entry and choose the Browse option to begin examining the results.

### Results from Microsoft Clustering

Because of the challenges of understanding results in higher dimensions, Microsoft Clustering provides several tools to examine estimated clusters. Figure 5.22 shows the top-level Cluster Diagram. This diagram color codes the clusters based on the number of cases in each group. Darker clusters are larger. The lines attempt to show which clusters are close to the others. Selecting a single node will high-



**Figure 5.23**

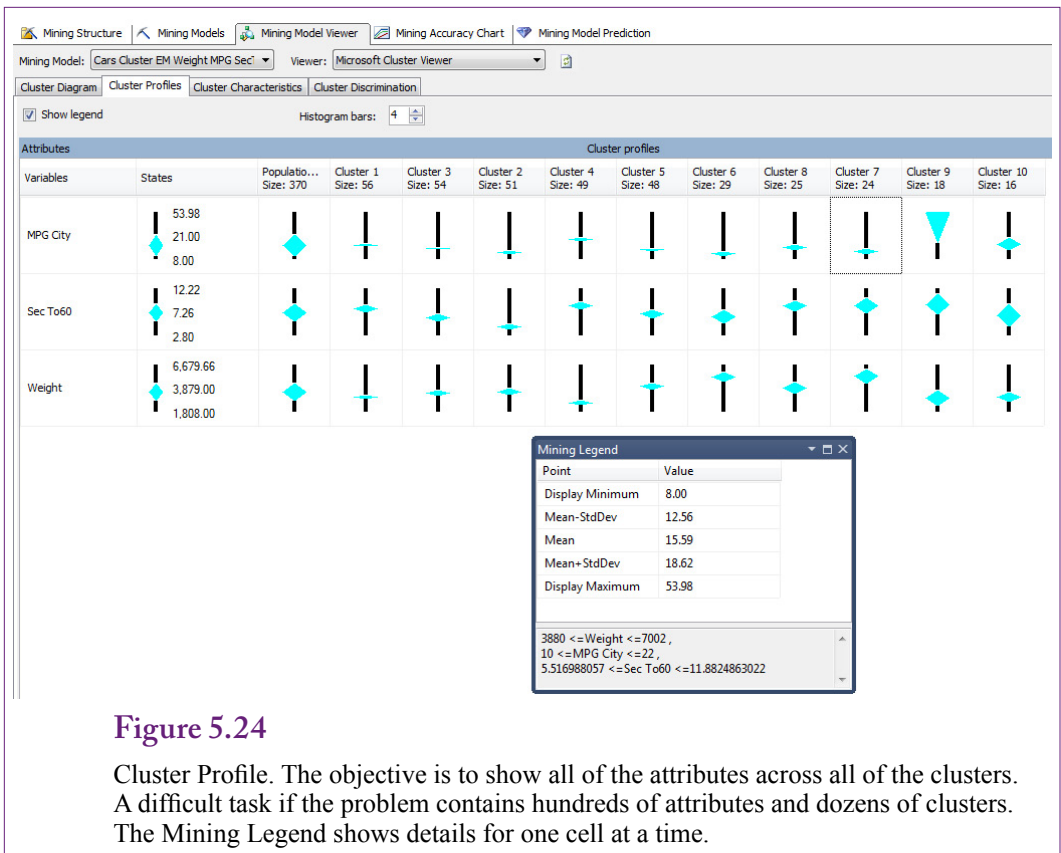
Cluster Diagram with MPG set to low values. Choose an attribute and a range to see which clusters consist of items with that attribute level.

light the nearest clusters to the one selected. The diagram provides minimal information for analysis, but it makes it easy to examine all of them at the same time.

The shading variable can be changed to one of the attributes using the dropdown list. The option also provides a selection box for various levels of the attribute. Figure 5.23 shows the clusters that contain cars with the lowest MPG. Cluster 9 is the strongest. Conversely, Clusters 4 and 10 represent higher MPG cars. This exploratory tool is useful for gaining an initial understanding of the clusters. The goal of the analyst is to understand what each cluster represents. Along those lines, the clusters can be given an explanatory name once the primary characteristics are identified.

One drawback to the Cluster Diagram is that it focuses on a single attribute. The Cluster Profile shown in Figure 5.24 is designed to provide information on all of the attributes across all of the clusters. It is a daunting task for a problem with hundreds of attributes and dozens of clusters. In some ways this tool provides too much detail, but the graphics provide some visual clues. The height of the blue diamond indicator represents the variation within a cluster for the specified attribute. Check the MPG attribute horizontally to see that most of the bars are small—indicating that each cluster represents cars within a narrow range of MPG. The glaring exception is Cluster 9, which appears to be a catch-all category, because the system defaults to 10 clusters.

The Mining Legend shows the details for any specific cluster. In the case of Cluster 7, MPG is between 10 and 22; 0-to-60 times are between 5.5 and 11.9 seconds, and weight ranges between 3880 and 7002 pounds. The mean and standard



**Figure 5.24**

Cluster Profile. The objective is to show all of the attributes across all of the clusters. A difficult task if the problem contains hundreds of attributes and dozens of clusters. The Mining Legend shows details for one cell at a time.

deviation of a specific attribute are also shown in the top of the legend. Compare Cluster 7 to the others and it is clear that it represents heavy cars, with low gas mileage, and medium-low acceleration.

To better understand the cluster, drill through the cluster to see the observations within it. This option was checked during set up. Right-click the cluster and choose the option to drill through including the structure. The structure includes the Make and Model attributes that were specifically included but not set as Input data. Figure 5.25 shows the list of items in Cluster 7. Most of the vehicles are trucks or large SUV models. Think about the results for a minute. The system was able to group these items together based on only three factors: acceleration, weight, and MPG. In some clusters and in other problems, it will be necessary to add more dimensions to separate out some of the vehicles. For example, check out Cluster 6 which has some of the lowest MPG numbers and heaviest vehicles. That list includes several Bentleys and Maybachs, along with a few trucks.

Before moving on, look for clusters similar to Cluster 7. Notice that Cluster 2 vehicles have low MPG, fast acceleration, and at least medium weight. Drill through to see the list of vehicles and note that it consists of performance cars, so yes; the two groups are similar; yet different.

Microsoft provides additional tools to compare one cluster to another. The Cluster Characteristics provide a graphical display of the attribute values assigned to each group. Figure 5.26 shows the rules for Cluster 7. Most of the attributes

Car ID	MPG City	Sec To60	Weight	Structure.Make	Structure.Model
530	17	11.21	5567	Audi	Q7 Diesel
556	16	9.22	4960	BMW	X5
569	20	7.8	5879	Cadillac	Escalade Hybrid
571	15	8.55	5840	Chevrolet	Avalanche
586	20	8.1	5573	Chevrolet	Silverado 1500 Hybrid
588	20	7.1	5775	Chevrolet	Silverado 3500 HD Regular
591	15	8.6	5672	Chevrolet	Suburban
592	15	8.28	5448	Chevrolet	Tahoe
593	20	8.08	5598	Chevrolet	Tahoe Hybrid
616	13	9.57	5208	Ford	E-150
624	15	8	5954	Ford	F-350 Regular Cab
625	8	9	8178	Ford	F-450
640	13	10.25	4919	GMC	GMC Savana 1500
641	15	9.47	4460	GMC	GMC Sierra 1500 Regular Cab
643	15	8.28	5448	GMC	GMC Yukon
644	20	8.09	5598	GMC	GMC Yukon Hybrid
718	14	8.42	5936	Lincoln	Navigator
748	17	11.99	5512	Mercedes-Benz	GL350 BlueTEC
751	18	11.31	5226	Mercedes-Benz	R350 BlueTEC
758	18	12.22	5081	Mercedes-Benz	Sprinter 2500 BlueTEC

**Figure 5.25**

Cluster 8 Drill Through with Structure. Examine the items that appear in the cluster. Most of the vehicles are expensive, and large, with huge engines.

are evenly balanced, which makes it harder to identify the important elements for this cluster. If some bars are much longer than others, those become the primary dimensions. The lowest element in this chart is the MPG City 8-13 range, so this cluster excludes vehicles with very low gas mileage. But, acceleration and weight cover most categories equally.

Particularly in the case of this cluster, it is easier to evaluate the cluster in comparison to other clusters. The Cluster Discrimination tool is an interactive display that compares one cluster to any other. It can also compare a cluster to all others, by choosing the complement as cluster 2. Figure 5.27 shows the comparison between Cluster 7 and Cluster 2. Cluster 7 represents heavier vehicles with medium-to-slow acceleration. Cluster 2 contains slightly lighter-weight cars and really fast acceleration. The differences in MPG are negligible.

## Prediction

Sometimes it is helpful to examine individual cases to see which cluster they would fall into. With EM clustering, it is also useful to look at the probability a specific point falls into a given cluster. Prediction is handled through a set of data mining functions. The Analysis Services provides an easy-to-use link to these functions that can be applied to a table of possibilities or to just one item. Setting up a table of inputs requires several steps, so the tool is easiest to use with just one item at a time.

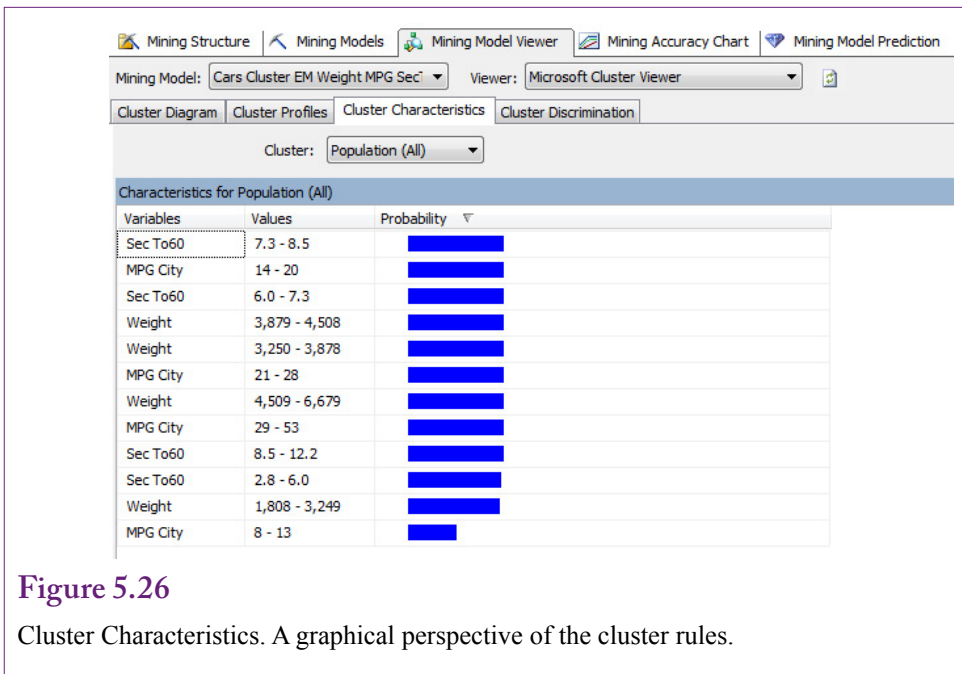
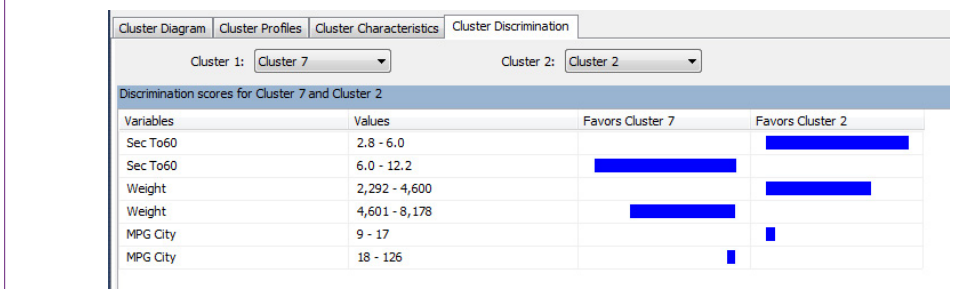
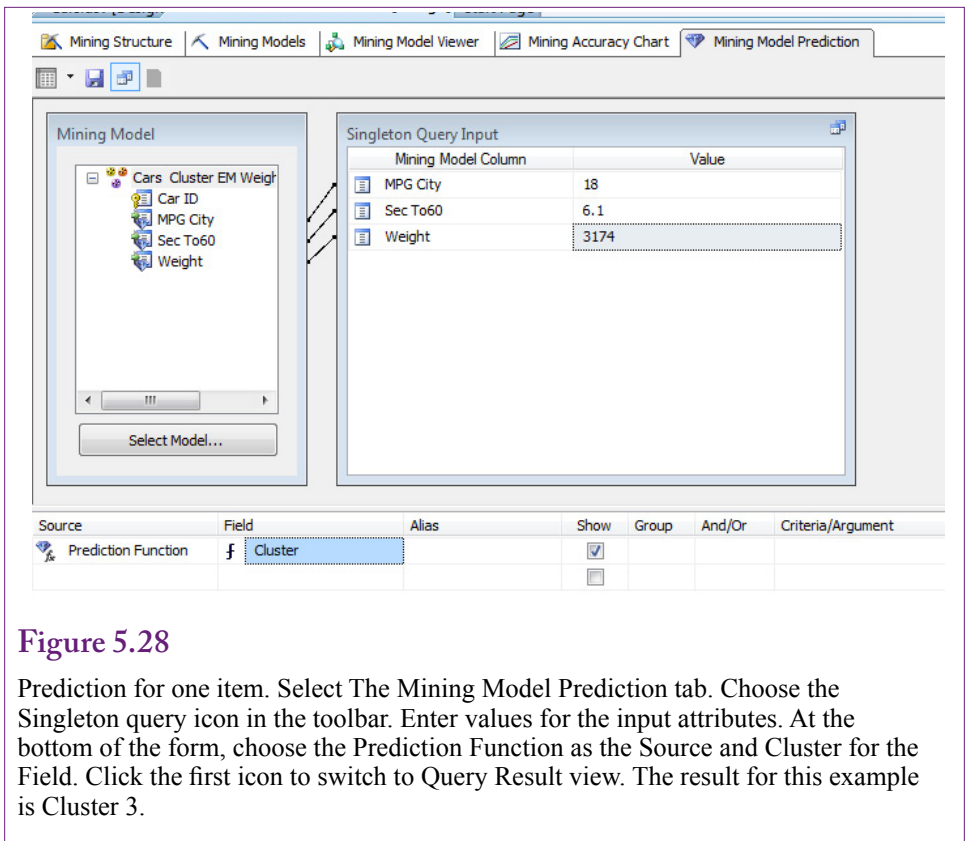


Figure 5.28 shows the basic process for predicting the cluster for a specific car. Begin by selecting the Mining Model Prediction tab. By default, the tool is configured to select input data from a table. Click the (third) icon (or right-click) to switch to a singleton query. Enter the input attributes for a specific car. The example here is: MPG=18, SecTo60=6.1, and Weight=3174. In the grid at the bottom of the form, select Prediction Function as the source and set Cluster as the field. Run the query by clicking the first icon (display results). The answer should be Cluster 3. Return to the design screen and change the Field to ClusterProbability. Run the new query and the result is 0.7946; so this particular car is associated with Cluster 3 with about 80 percent probability. Remember that with EM clustering, an individual point could be associated with other clusters. It is straightforward to obtain these probabilities. On the query design grid, under the column for Criteria/

**Figure 5.27**

Cluster Discrimination. Compare one cluster to (a) any other, or (b) all others. Useful for comparing clusters that have similar but slightly different attributes.





**Figure 5.28**

Prediction for one item. Select The Mining Model Prediction tab. Choose the Singleton query icon in the toolbar. Enter values for the input attributes. At the bottom of the form, choose the Prediction Function as the Source and Cluster for the Field. Click the first icon to switch to Query Result view. The result for this example is Cluster 3.

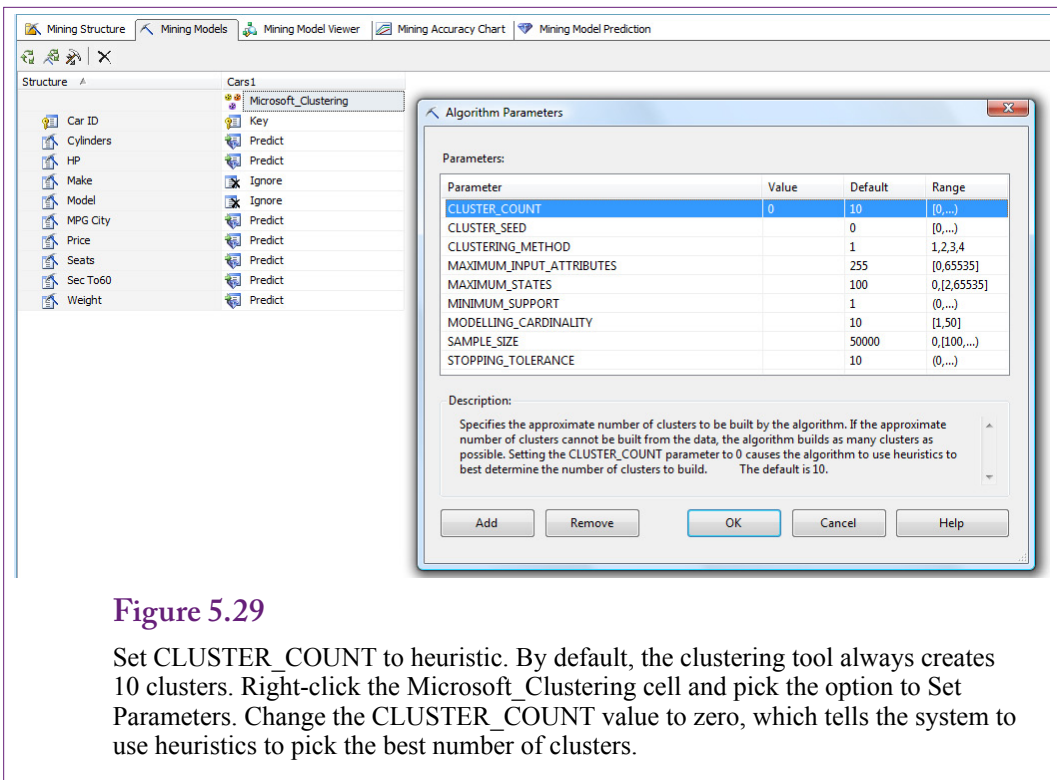
Argument, enter 'Cluster 8' and run the query again. You must include the single quote characters around the cluster name. The probability for Cluster 4 should be 0.0125. Other cluster names can be tested as well, and each cluster can be tested separately. It is probably useful to return to the Cluster Diagram and select Cluster 3 to highlight the clusters most closely associated with Cluster 3 (1, 2, 8, 9, and 10) and test those values first.

### Larger Model and Parameter Changes

It is time to make the problem a little more realistic and more complex. The initial model used only three measurable attributes on the cars, because the tradeoffs among those three are relatively easy to see. The full model contains a few more attributes that might be important differentiators for clusters. Close the existing model.

Because the data source view contains all of the attributes for the cars, a new data mining model can be created using the same data source and view as the earlier model. Create a new model by right-clicking the Mining Structures entry. Follow the wizard and choose the Microsoft Clustering technique. Stick with the Cars table and verify that CarID is set as the Key attribute. As before, select Make and Model by marking the checkboxes in front of the columns; but do not set them as Key, Input, or Predictable. They are only going to be used as lookup values for the drill through option. Select most of the other attributes as Input and Predictable: Cylinders, HP, MPGCity, Price, Seats, SecTo60, and Weight. Year is not necessary





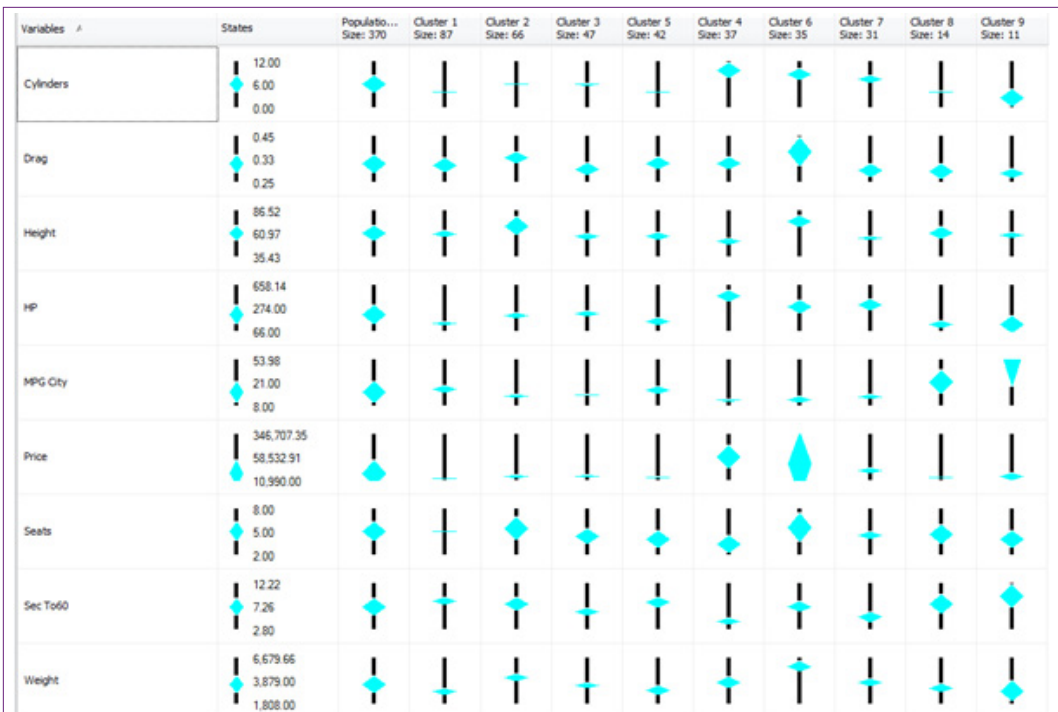
**Figure 5.29**

Set `CLUSTER_COUNT` to heuristic. By default, the clustering tool always creates 10 clusters. Right-click the `Microsoft_Clustering` cell and pick the option to Set Parameters. Change the `CLUSTER_COUNT` value to zero, which tells the system to use heuristics to pick the best number of clusters.

because all of the vehicles are from the same year. In a more extensive database across multiple years, the Year attribute would be useful. Work through the rest of the Wizard screens, enter a unique and memorable name and be sure to check the Drill Through option. Finish the wizard steps but do not process the model yet.

Remember that the initial results contained 10 clusters. This value is the default, and the Clustering technique will always try to fit 10 clusters regardless of the problem. Sometimes the managers and analysts will have a better idea of the number of desired clusters. Other times, the heuristics described in the Model section are useful to help determine the appropriate number of clusters. To set these options, click the Mining Models tab. Right-click the `Microsoft_Clustering` cell and choose the option to Set Algorithm Parameters. As shown in Figure 5.29, change the **CLUSTER\_COUNT** entry to zero (0) to ask the system to heuristically find the appropriate number of clusters.

Take a closer look at some of the parameters that can be set. The **CLUSTERING\_METHOD** is one that might be useful. It accepts four values: 1=Scalable EM, the default; 2=Non-scalable EM, 3=Scalable K-means, and 4=Non-scalable K-means. For most cases, the non-scalable choices (2 and 4) have limited value. Although they are a little more complete at testing various combinations, they can be used only with problems with a relatively small number of observations. The primary choices are the EM method (1) and K-means (3). As explained in the Model section, analysts might choose K-means if it is important to assign observations to exactly one cluster. For now, leave the `CLUSTERING_METHOD` at the default EM value.



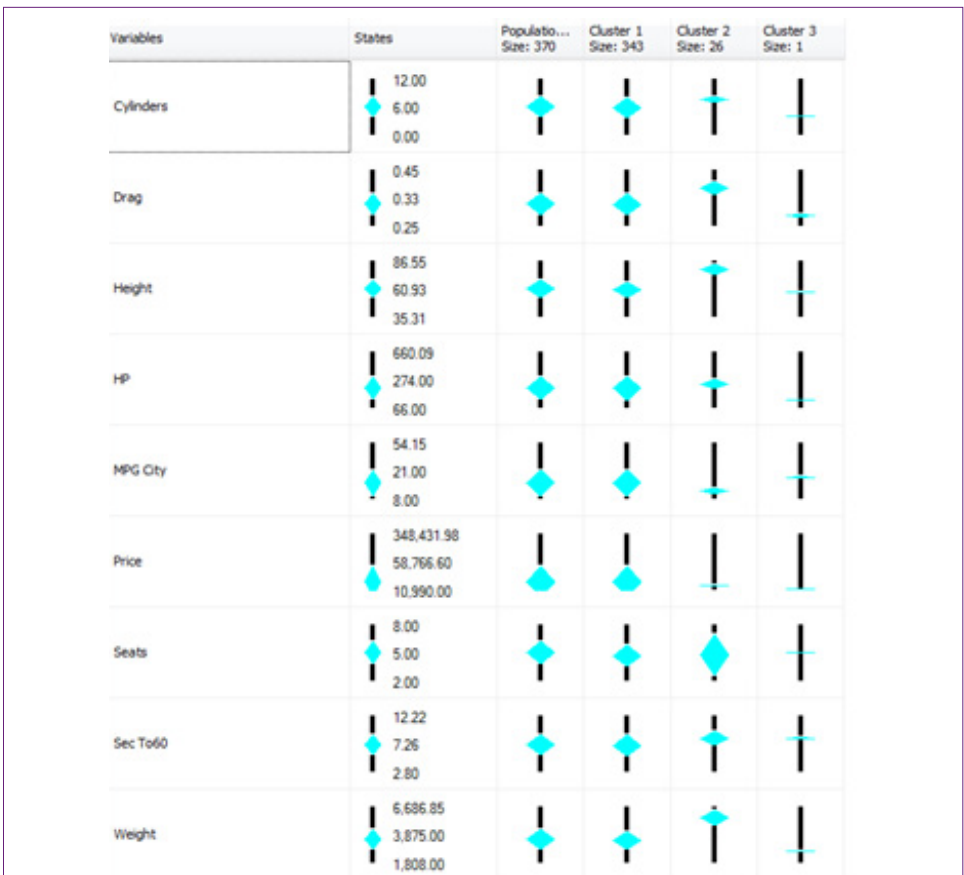
**Figure 5.30**

Complete results. Note the number of clusters (9), and notice that the internal variance is low, indicating tight clusters in almost all cases.

The other parameters can be used to fine-tune results or performance. Maximum values for inputs and states place limits on the size of the problem and are rarely changed. Cluster seed, modelling cardinality, and sample size are used to control the estimation process and should not be changed unless the problem is too large to be solved with the defaults. The only other value analysts might change is minimum support which specifies the smallest size cluster to be allowed. It defaults to one observation, which is not much of a group.

When the cluster count has been set to zero, process the new model and browse it. The first thing to notice is that it contains 9 clusters instead of 10. And the relationships among the clusters are thinner than before. Adding more attributes made it easier to classify the vehicles. Figure 5.30 shows the profile results. The increased number of attributes makes the chart more complex, but it appears to have improved the overall clustering process. The variance within each cluster for most attributes is low—indicating tight clusters. Examine a few of the clusters to see which vehicles fall into each category. The vehicles appear consistent and it should be possible to assign meaningful names to each cluster.

For comparison, the same data was evaluated using K-means clustering (option 3 in the parameters). Figure 5.31 shows the cluster profiles. Note first that the heuristic chose only three clusters to summarize over 300 vehicles. Also, notice the first cluster contains 3443 vehicles, or almost all of them. Drill through to check out the vehicles in each cluster—the second cluster is generally large trucks and SUVs and the third cluster contains one car, which is probably there because of



**Figure 5.31**

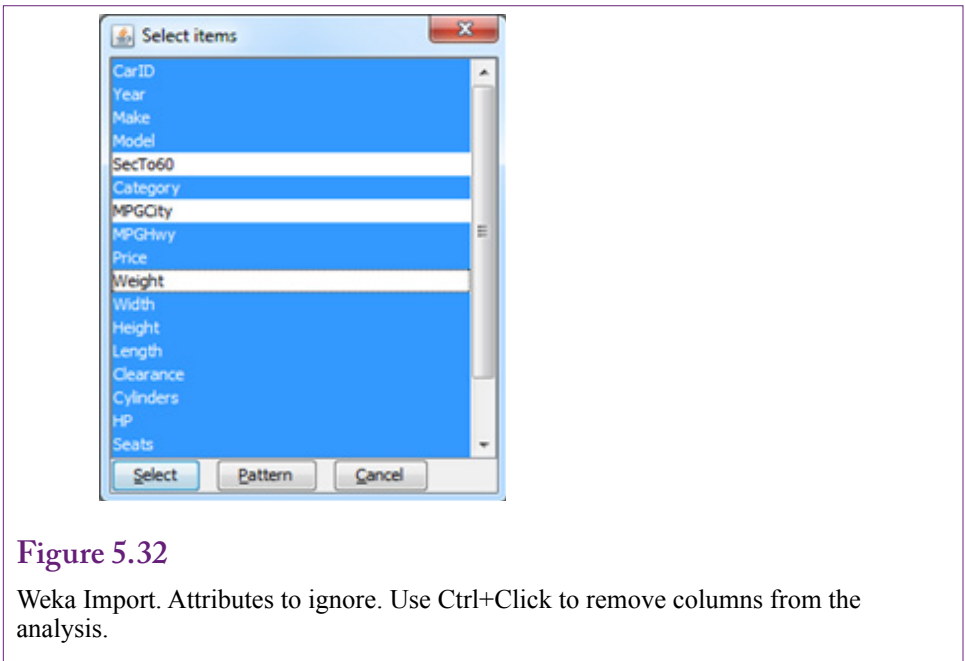
K-means clustering on all attributes. The heuristic resulted in choosing only three clusters. But almost all vehicles are in Cluster 1.

the lower price. If K-means is selected as the clustering method, it will be best to specify the desired number of clusters to override the poor default choices. But even when 9 clusters are manually specified to match the number from the EM results, almost half of the observations (173/370) are lumped into one cluster. Overall, the EM clustering technique seems better suited for this data.

Note that it is possible to assign names to each of the clusters. Whenever possible, clusters should be given meaningful names that describe the attributes represented by points in that cluster. With the car data, names might include “large SUVs,” “Sports cars,” and “Hybrids.”

## Traditional EM Clustering

**How are traditional EM methods different from Microsoft Clustering?** Many of Microsoft’s data mining tools have additions or tweaks that make them different from the pure model. Throw in the fact that clustering is heavily dependent on (1) the distance measure, (2) the selection of the number of clusters, and (3) the way the algorithms seek a global instead of a local optimum.



**Figure 5.32**

Weka Import. Attributes to ignore. Use Ctrl+Click to remove columns from the analysis.

It quickly becomes possible to obtain different results from every tool tested. For comparison, the Weka tool from the University of Waikato in New Zealand (<http://www.cs.waikato.ac.nz/ml/weka>) was used to examine the Cars dataset.

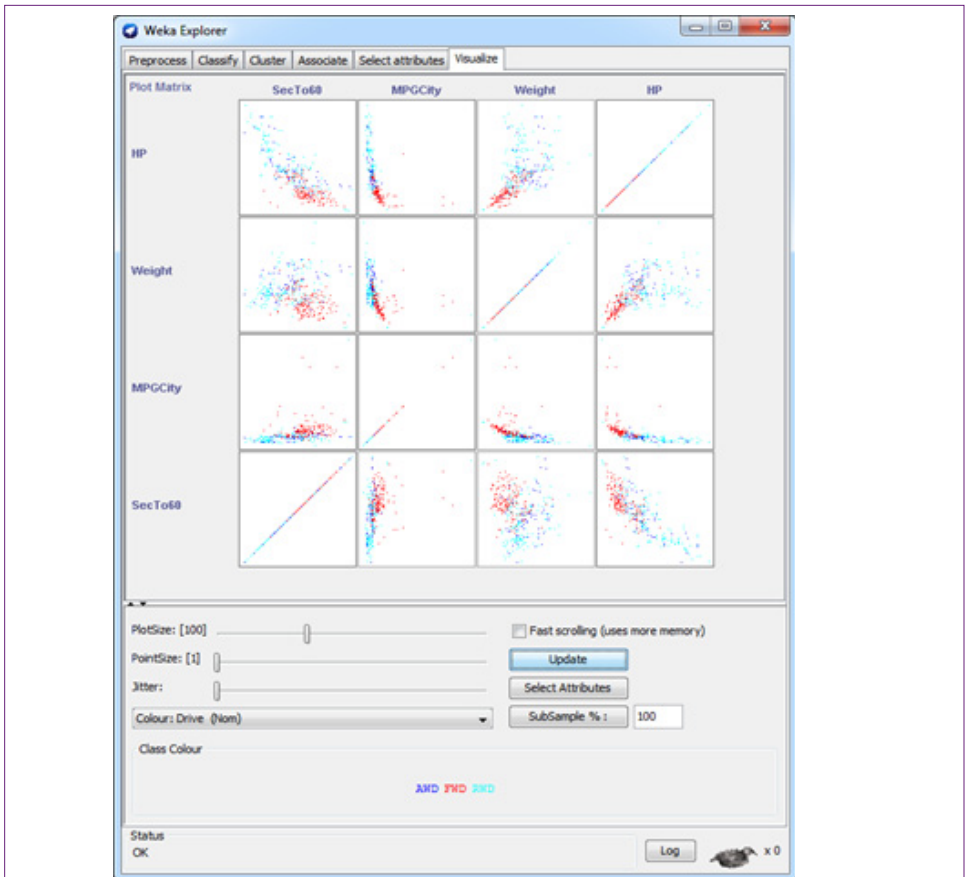
### Goals and Data

The Weka tool is written in Java and runs on most computers. It imports data from CSV files and can save projects in a proprietary format. It has the ability to

**Figure 5.33**

Traditional cluster results from Weka. The content of each cluster is defined by the mean and standard deviation.

	C0	C1	C2	C3	C4	C5	C6
N/Pct	0.15	0.26	0.07	0.06	0.14	0.19	0.13
SecTo60							
mean	8.293	6.629	8.424	6.204	4.458	8.231	8.693
std. dev.	1.446	1.032	2.181	1.342	0.603	0.896	0.899
MPGCity							
Mean	16.490	17.917	46.397	11.601	13.905	21.722	27.284
std. dev.	2.353	1.208	27.227	1.308	1.834	1.350	1.983
Weight							
mean	5048	3929	3317	5722	3849	3362	2717
std. dev.	582.5	421.8	792.6	751.9	406.0	295.3	234.8

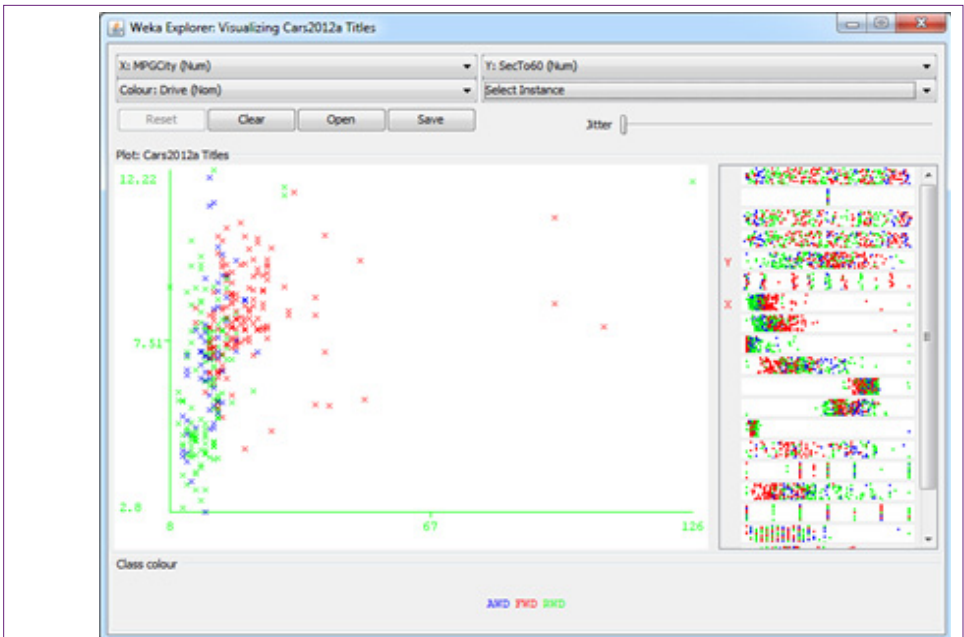


**Figure 5.34**

Weka 2D visualization. The data are plotted in two-dimensional charts for each combination of attributes.

directly connect to a DBMS, but the connection requires the use of Java ODBC. For large problems, it is probably worth the time to configure this connection. For small problems similar to the simple Cars table, it is easier just to import the CSV file. Weka is a tool that contains basic algorithms for several data mining tasks. It is free and relatively easy to use.

The basic Cars.csv file can be imported directly into the Weka Explorer. Start Weka, open the Explorer and click the Open file button. Navigate to the appropriate folder, change the file type list to CSV and input the file that includes the titles. The attributes should be displayed in the main list. Switch to the Cluster tab. By default, the Explorer uses all of the attribute columns in the file. Some of these are unnecessary, and it will be best to start with the smaller model that uses only the SecTo60, CityMPG, and Weight attributes. Click the Ignore attributes button to open the selection window. Figure 5.32 shows the selected items to be ignored. Hold the Ctrl key down and click on all but the three desired entries to remove them from the analysis.



**Figure 5.35**

Weka cluster results shown with color coding. Any pair of attributes can be selected. The clusters are highlighted. Selecting any point with the mouse displays the details for that specific object.

## Results

The basic Cars.csv file can be imported directly into the Weka Explorer. Start Weka, open the Explorer and click the Open file button. Navigate to the appropriate folder, change the file type list to CSV and input the file. Figure 5.33 shows the results of the analysis of the three primary attributes. As a first point of differentiation, note the automatic selection of seven clusters. Based on counts of observations, the clusters seem reasonably balanced. Examine the clusters by looking at the clusters with the fastest acceleration (lowest 0-60 times): Cluster 3 and Cluster 4. At first glance, the clusters appear similar, but compare the MPG and Weight attributes. Vehicles in Cluster 3 are much heavier (larger) with lower MPG—probably SUVs and trucks. The vehicles in Cluster 6 are likely to be sports cars. Although the means and standard deviations provide useful data, it can be difficult to interpret.

Weka provides an integrated visualization tool to make it easier for humans to evaluate the attributes. Figure 5.34 shows the two-dimensional plots of four attributes. The tool shows combinations of all four attributes. Including more attributes in the file leads to a large set of comparison charts. The goal of the visualization is to spot pair-wise correlations and clusters. The attributes can be selected via the button on the form. The purpose of the large number of small charts is to provide a quick overview. Pair-wise correlations are relative easy to spot—as lines in the charts. Clusters appear as concentrations of dots, but the overlap of points makes them harder to spot. Selecting one of the charts provides a larger

Attrib.	All	0	1	2	3	4	5	6	7	8
N	370	21	85	33	44	34	86	32	15	20
Sec To 60	7.26	4.48	6.54	4.67	7.72	9.42	8.36	8.67	8.50	4.92
MPG City	20.7	14.5	18.7	15.0	16.1	20.2	30.3	25.2	15.0	11.7
Price	58,533	127 K	40,632	89,370	42,203	31,426	21,042	25,440	27,856	330 K
Weight	3,879	3,421	3,977	4,020	5,124	3,943	3,030	2,948	5,278	4,947
Height	61.0	49.7	60.0	55.5	72.9	65.3	58.8	57.3	77.9	5.1
Cyls	5.9	7.9	5.7	7.7	7.0	4.5	3.9	3.9	6.8	12.0
HP	274	434	281	456	303	191	153	158	290	539
Seats	4.7	2.0	4.7	4.4	6.9	5.1	5.0	3.3	2.7	4.1
Drag	0.326	0.325	0.323	0.313	0.358	0.341	0.312	0.328	0.334	0.325

**Figure 5.36**

Weka K-means results on all attributes. Values are center points of the clusters.

view of the relationships between the pairs of attributes. A third dimension can be chosen to display as a new color.

Once the clustering analysis has been run, it is possible to display a cluster chart by the paired dimensions. On the main Explorer page, right-click the results line and choose the Visualize option. Figure 5.35 shows the clusters comparing MPG with acceleration. Select a few of the top-right marks and the cars will be hybrids and low-price sedans with small engines and good gas mileage. Select one of the items near the bottom-left of the chart and they will consist of high-performance ultra-luxury cars with huge engines, no gas mileage, and absurd prices. The drop-down lists at the top of the form make it easy to switch the chart to compare different attribute pairs.

Humans perform reasonably well when estimating correlations and clusters in two dimensions. Three dimensions work for some people, but it is difficult to display three dimensional charts without hiding most of the data. Most people cannot visualize relationships beyond three dimensions. Hence, the reason for creating an entire panel of pair-wise two-dimensional charts.

### K-Means Clusters

Weka appears to do a better job with the K-means technique than the Microsoft Clustering tool. The tool is selected near the top of the Explorer window under the Cluster tab—click the Choose button and select the SimpleKMeans technique. Right-click the resulting selection and choose Properties to change the underlying parameters. Specify the numClusters to 9 to see how well the results match the Microsoft EM outcome. In the end, the observations are distributed better across the clusters than they were with the Microsoft K-means technique. The largest cluster holds 24 percent of the observations

Figure 5.36 shows the resulting centers of the K-means. Notice the relatively even distribution of the number of observations. Glancing at the results, Price seems to be a good starting point for evaluating the clusters because most of the



Attrib.	C1	C2	C3	C4	C5	C6	C7	C8	C9
Cyls.	4.0	6.0	5.9	9.6	4.0	8.6	7.2	4.0	2.4
Drag	0.32	0.35	0.30	0.33	0.33	0.38	0.30	0.30	0.29
Height	60.1	69.0	57.3	51.7	57.6	73.7	55.6	61.4	59.0
HP	159	263	282	516	184	375	397	150	150
MPG City	24.4	17.2	18.3	13.1	23.3	13.8	16.8	31.2	64.3
Price	19660	36225	38434	178041	28656	126770	77806	25946	35557
Seats	5.0	5.4	4.4	3.3	4.0	5.5	4.5	4.6	4.0
Sec To 60	8.51	7.93	6.30	4.40	8.23	7.39	5.21	7.97	9.47
Weight	3080	4562	3729	4012	3227	5743	4023	3424	3135
N	87	66	47	37	42	35	31	14	11
Sample	Toyota Camry	Jeep Liberty	Chevy Camaro	Merc. CL63 AMG	Mini Cooper	Cadillac Escal.	Ford Shelby GT500	Toyota Prius	Nissan Leaf
Descrip.	Basic Sedans	Basic SUVs	Mid- perf.	High- price perf.	Mid- price perf.	Big SUVs	High- Perf.	Hybrid	Elec.

**Figure 5.37**

Summary of Microsoft EM clusters. Means are listed. Sample vehicles are pulled from cluster list.

clusters have fairly strong separation based on Price. Weight might be a good secondary differentiator. Compare Clusters 1 and 3 that have similar price points—the weights are quite different. Based on the number of seats and price, Cluster 3 probably represents large SUVs.

## Comparison

**Are the results from the multiple tools and methods really different?** Or, is one tool and method better than the others? This second question is easier to answer: No. Most of the differences across tools arise because of choices in the number of clusters. For the most part, it should be possible to specify the number of clusters to be the same in any tool, and using the same technique, obtain close to the same results. But there are no guarantees. The clustering algorithms are always slightly different. On the other hand, the tools are considerably different in how the data and results are presented and visualized. It is quite possible that analysts will develop a preference for a tool based on the way the results are presented and searched. Most tools automatically include all data by default. For a side-by-side comparison, the holdout size in Microsoft Clustering should be set to zero.

As a test, the Weka EM tool was used to generate the same number of clusters as found by the Microsoft tool (9). Figure 5.37 summarizes the Microsoft clusters. To save space and keep the table legible, only the means of the clusters are listed.

Figure 5.38 lists the clusters found using the Weka EM clustering tool. Using its own heuristics, the Weka EM tool found eight clusters on the full set of attributes.

Attrib.	C0	C1	C2	C3	C4	C5	C6	C7	C8
Cyl.	5.7	5.8	3.8	3.9	8.0	12.0	7.2	6.4	7.9
Drag	0.32	0.34	0.32	0.32	0.38	0.33	0.32	0.35	0.31
Height	59.6	69.3	58.5	59.6	76.3	54.4	50.1	74.2	55.8
HP	278	263	150	156	340	539	425	276	458
MPG City	18.9	17.8	25.4	27.9	14.7	11.7	15.3	15.9	14.8
Price	41,475	32,623	25,304	21,676	48,610	318,756	119,627	36,841	82,745
Seats	4.7	7.1	2.2	4.8	6.4	3.8	2.4	3.8	4.5
Sec To 60	6.69	7.70	8.57	8.54	7.75	4.83	4.46	8.88	4.73
Weight	3959	4489	2812	3114	5689	4791	3373	5152	4122
N	88	22	12	126	26	22	25	23	26
Sample	Volvo XC70	Ford Explor.	Mazda Miata	Toyota Prius	Lincoln Navig.	Bentley	Porsche 911	M-B. R350	Jaguar XK
Desc.	Mid-Level	Cheap. SUVs	Lighter Cars	Basic sedans	Big SUVs	Luxury	High Perf.	Luxury SUVs	Luxury Perf.

**Figure 5.38**

Summary of Weka EM forced to seven clusters. Means are listed. Sample vehicles are pulled from cluster list.

But it is easier to compare clusters if the same number is used for both cases. Try comparing the clusters. Start by looking at prices. Cluster 4 in the Microsoft table probably corresponds to Cluster 5 in the Weka table. But there are differences in the items—look at the counts. The largest clusters in both cases are labeled as “basic sedans,” but the clusters are different. Notably, the Weka cluster includes the hybrids which are separated by the Microsoft algorithm. The Weka algorithm also separated SUVs (and large sedans) differently. Price seems to have played a greater role in the Weka clustering versus weight and MPG in the Microsoft clustering approaches. So, yes, different algorithms give different results. It might be useful to test multiple approaches on data to see if additional insights can be gained. The goal of clustering is not to find a single “perfect” answer—such a thing does not exist. Instead, the objective is to gain insight into the items involved.

These results are not meant to imply that one tool is better than the other. Both tools have configuration options to support multiple distance measures. Although, Weka probably has more options, and Weka does allow people to write custom code to create new models. The main purpose of the exercise is to show that the choice of a model and distance measure can make a big difference in the results. Regardless of the tool used, it is useful to try some of the other options to see if one measure might be more useful for a particular problem.

The choice between K-Means and EM techniques is much clearer. EM uses a soft cluster definition where an observation falls largely into one cluster, but can also be associated with other clusters. When this concept matches reality, probably most of the time, EM does a better job of defining clusters because it can handle these in-between cases better. The key feature of K-means is that it forces each observation into a single cluster. There might be times when this clarity is needed, but forcing data to behave to a strict model might not give the best results.

## Customer Clustering with Categorical Data

---

### How does categorical data change the results and interpretation of clusters?

The automobile dataset was created specifically because most people are familiar with the attributes and all of the attributes are measured with continuous data, which works best for clustering. But, business problems often involve categorical data so it is important to know how to interpret results involving categorical data.

### Data

The Corner Med Patient database contains some relatively common personal attributes of customers, and a few that are unique to the medical world. Begin by examining the Patient table, which is equivalent to a Customer table for typical businesses. It contains attributes on DateOfBirth, Gender, Race, TobaccoUse, and location information. The bulk of the data for Corner Med is generated from government reports on anonymous patient visits that include most of those attributes. However, the location and phone number data (and names) are randomly generated. In a real case, it would be useful to include the location data. In this example, it is unlikely to be important. The Visit table also contains some useful information about patients, such as the InsuranceCompany, and blood pressure data. The blood pressure data is interesting, but it contains a high percentage of missing values so it should not be used. The insurance company data is important because it defines how the business gets paid with various limits on procedures and charges. In particular, the government Medicare and Medicaid programs have strict limits. One additional attribute might be useful to classifying patients—a measure of the patient's involvement with Corner Med. This dimension could be measured either by a count of the number of visits per year or the total amount billed per year by patient. The AmountCharged column in the VisitProcedure table provides that value when it is summed by patient.

To analyze the data with Microsoft Clustering, create a new Analysis Services project in Visual Studio. Add a new data source that connects to the CornerMed database. Create a new Data Source View that contains most of the tables: Patient, Visit, VisitProcedures, ICD9ProcedureCodes, VisitDiagnoses, ICD9DiagnosisCodes, and VisitMedications. Only three of these tables are needed for the clustering problem, but the other tables can be used later for different problems.

Because the attributes needed appear in three tables, it is necessary to build a named query to combine the tables into a single source. Right-click the main data source view screen and choose option for New Named Query. Provide a name for the query that is unique and describes the data, such as PatientCharges. The current data in CornerMed consists of a single year (2010), so no constraints are necessary for the date. Race, Gender, and TobaccoUse are straightforward because they are in the Patient table. The InsuranceCompany attribute could cause problems if people have multiple visits and change insurance companies at each visit—but that rarely happens within a single year, so it can be ignored for now. The total of the AmountCharged can be computed as the Sum of a GROUP BY query. The Age presents a challenge. The database holds date of birth, which is the best way to handle it. An Age column can be computed by subtracting the date of birth from a specific date. But which date should be used? That is, the age of the patient on which day is needed? It would be easy to pick the date on the day of the visit, but the subtotal for Amount runs across multiple visits and dates, so that Age

of Visit value will change. It is better to choose a fixed date so the patient's age is constant within that year. The easy solution is to use the end of the year or start of the next year (01-Jan-2011). The query is:

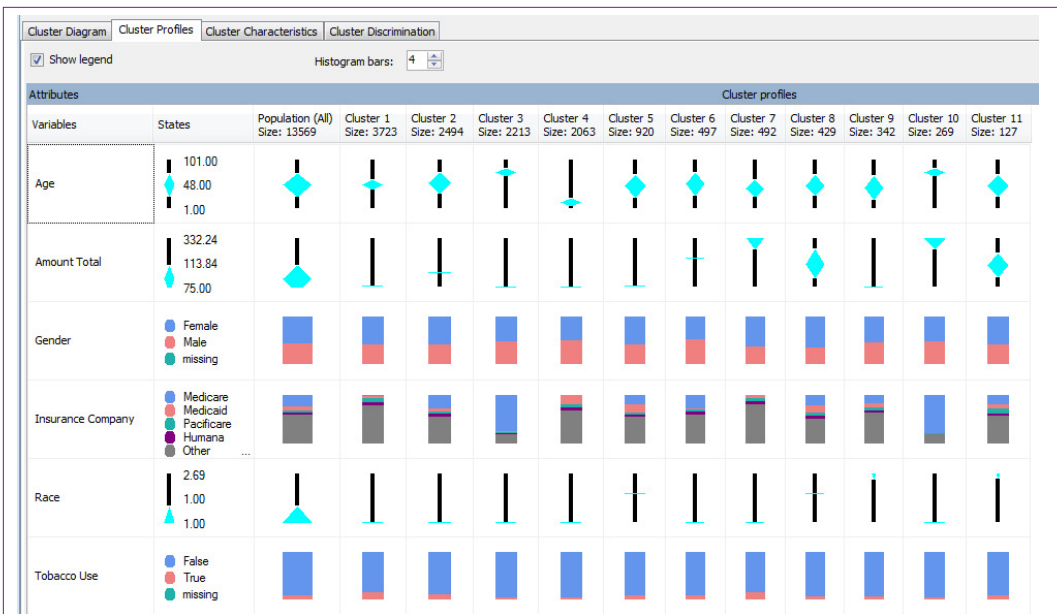
```
SELECT      dbo.Patient.PatientID, DATEDIFF(yyyy, dbo.
Patient.DateOfBirth,
      CONVERT(DATETIME, '2011-01-01 00:00:00', 102)) AS Age,
dbo.Patient.Race,
      dbo.Patient.Gender, dbo.Patient.TobaccoUse, dbo.Visit.
InsuranceCompany,
      SUM(dbo.VisitProcedures.AmountCharged) AS AmountTotal
FROM      dbo.Patient
INNER JOIN dbo.Visit ON dbo.Patient.PatientID = dbo.Visit.
PatientID
INNER JOIN dbo.VisitProcedures ON dbo.Visit.VisitID = dbo.
VisitProcedures.VisitID
GROUP BY  dbo.Patient.PatientID, DATEDIFF(yyyy, dbo.Patient.
DateOfBirth,
      CONVERT(DATETIME, '2011-01-01 00:00:00', 102)), dbo.
Patient.Race,
      dbo.Patient.Gender, dbo.Visit.InsuranceCompany, dbo.
Patient.TobaccoUse
```

To use the same data in an external program, such as Weka, copy the SQL and paste it into a new query window in SQL Server. Run the query to obtain the results. Right-click the Results window and choose the option to Save Results As. Select a location and save the values in a CSV file. Before closing the name query editor, test the query to ensure it returns correct values. Once the query is created, right-click the PatientID and select the option to set it as the **Logical Primary Key**. The analytical tools require that one column uniquely identify each row. By setting it now, the tools automatically pick it up and save steps and reduce errors later.

## Microsoft Clustering Results

Once the named query is created to define the data, the process of building the model and analyzing it is straightforward. Create a new mining model and choose the Clustering method. Set the holdout percentage to zero so that all cases are used in the clustering process. Also, before processing the model, switch to the Mining Models tab and set the algorithm parameters. Set the cluster count to 0 to have the system choose the number of clusters instead of forcing it to fit 10 clusters.

Figure 5.39 shows the results from Microsoft Clustering on the Corner Med patient attributes. Notice the use of the stacked bar charts for the categorical attributes. To evaluate the role of the attribute, look across the clusters for changes in the charts. For instance, tobacco use is slightly higher in some clusters and almost non-existent in others. Notice that Cluster 4 is basically non-smoking, but check the age to see why. So Cluster 4 could be called *Babies*, and it is the only cluster that covers that specific age group. Cluster 7 has a higher percentage of tobacco users than the other clusters. Race is 1 (white), Gender is biased towards female, age is slightly less than average, insurance does not look good (other), and this cluster has one of the highest charges per patient. As another example of categorical data, check out Clusters 3 and 10 which have the highest percentage of Medicare coverage. From a medical and business perspective, those two clus-



**Figure 5.39**

Corner Med patient clusters. Compare the categorical attributes in each cluster and to the population. Tobacco use is slightly higher in some clusters, Gender varies in some, and insurance is a major classifier.

ters deserve more investigation. Most of the attributes are the same, including the highest ages (which fits with the Medicare attribute), yet Cluster 3 has the lowest total spending and Cluster 10 the highest. Some missing factor must differentiate the clusters and it would be nice to track down that factor.

## Weka Clustering Results

Figure 5.40 shows the results from the Weka EM clustering on the same Corner Med patient data. The CSV file was exported directly from the query run inside SQL Server. The results for the categorical attributes have been converted to percentages and highlighted using Excel. The percentages are easier to compare across clusters. Notice that the results are considerably different from the results generated by Microsoft Clustering. However, five of the Weka clusters focus on Medicare and Medicaid, which parallels the Microsoft results. The biggest difference is that five of the clusters here are driven by gender differences. Plus Clusters 2 and 4 represent tobacco users. In fact, the first five clusters are relatively easy to describe because of the dominant features in each cluster. The last two clusters are more difficult to understand, but because the last cluster holds only five observations, it is somewhat irrelevant.

As a side note, Weka can perform hierarchical clustering when the Cobweb clustering method is chosen. However, even with the smaller automobile dataset, the results are difficult to read. It could be useful for identifying the appropriate number of clusters for a problem. The tree view shows the effects of different numbers of clusters.

	0	1	2	3	4	5	6
Count	4309	7723	526	10	820	285	5
Age	80.4	42.5	41.3	74.6	51.5	31.5	51.4
TotalAmount	122.23	111.65	120.57	125.72	115.94	92.14	123.02
Race	1.04	1.20	1.30	1.04	1.08	3.43	1.02
TobaccoUse	0.00	0.00	0.99	0.00	1.00	0.23	0.27
Gender							
Female	0.996	0.626	0.989	0.001	0.416	0.229	0.592
Male	0.004	0.374	0.011	0.999	0.584	0.771	0.408
Insurance							
Cigna	0.002	0.062	0.021	0.050	0.062	0.201	0.455
United Healthcare	0.002	0.065	0.038	0.003	0.055	0.022	0.023
UniversalCare	0.002	0.068	0.062	0.010	0.063	0.028	0.047
Assurant	0.002	0.074	0.068	0.005	0.050	0.049	0.046
Humana	0.002	0.077	0.074	0.007	0.054	0.064	0.050
Blue Cross/Blue Shield	0.002	0.072	0.067	0.005	0.055	0.053	0.043
Pacificare	0.002	0.077	0.082	0.007	0.055	0.029	0.052
Aetna	0.002	0.061	0.115	0.048	0.040	0.162	0.028
Kaiser Permanente	0.002	0.069	0.049	0.008	0.049	0.040	0.045
Medicare	0.974	0.123	0.033	0.838	0.243	0.088	0.025
Medicaid	0.002	0.099	0.255	0.001	0.093	0.161	0.039
Self	0.002	0.060	0.059	0.003	0.079	0.016	0.043
Nationwide	0.002	0.072	0.043	0.008	0.057	0.055	0.045
Worker	0.002	0.015	0.021	0.005	0.036	0.022	0.034
Charity	0.002	0.004	0.013	0.001	0.010	0.008	0.025

**Figure 5.40**

Corner Med patient clusters via Weka. The categorical results were converted to percentages and highlighted to make them easier to read.

In the end, it makes sense to test multiple clustering methods. The results might vary depending on the tools measurement, search, and underlying assumptions. But, data exploration requires thinking about the data in different ways. The goal is rarely to arrive at a precise, repeatable definition. Instead, data mining seeks to shed light on relationships and provide insights. If multiple tools produce different perspectives, it should be considered a useful encouragement of further exploration.

## Summary

---

Clustering is an unsupervised learning method that tries to find groups of items that are close to each other. Close is defined by a distance measure, and different clusters arise depending on the details of the distance measure. Choosing the number of clusters is a second challenge that has several different solutions. Sometimes the number of clusters can be specified within the business problem, other times a heuristic approach is used to estimate the appropriate number.

Three primary clustering methods are in common use: K-means, expectation maximization (EM), and hierarchical. The primary difference between K-means and EM is that K-means creates hard clusters where each observation can fall into only one cluster. EM uses probability to define softer groupings, so that an observation has a probability of belonging to a cluster and could be associated with multiple clusters. Results from EM clustering can be harder to interpret if the associated cluster probabilities are weak. Yet, it is often a more realistic approach because items being clustered often fall into gray definitions and can fit into multiple clusters. Hierarchical clustering attempts to solve the problem of choosing the number of clusters by finding all possible sets of clusters, from one cluster down to N clusters where each observation falls into its own group. The technique only works with a small number of observations.

Principal component analysis (PCA) is a different statistical tool that is sometimes used to simplify attributes. It is slightly different from clustering. Clustering attempts to collect observations into smaller groups. PCA attempts to reduce the number of dimensions or attributes by finding combinations that explain the data almost as well as the original set of attributes. Although the approach and the tools are different, in the end both tools attempt to reduce the complexity of a problem by finding smaller combinations of attributes that can represent the data.

Clustering works best with continuous data because of the reliance on distance measures. Categorical data can be analyzed as nominal measures that assign a distance measure to different attribute values. If the analysts and managers are aware of better measures, better clusters can be created by defining a new variable that contains an expert's assigned measure.

When examining results, analysts and managers should strive to assign names and descriptions to each cluster that accurately reflect the primary focus of the cluster. These groupings are used to examine the impact of decisions on the various groups and explain how each cluster might react to changes. For example, participants in one cluster might be more sensitive to price changes than another cluster. One challenge with analyzing cluster results is that they can vary greatly depending on the technique chosen as well as the specific tool and algorithm used to search the data. It can be easy to get into arguments over which tool or method generates the most accurate clusters. Clearly delineated data can lead to strong clusters—and these are generally consistent across tools and methods. It is the gray areas that lead to diverse results. Still, anything that helps managers and analysts see the data in new ways has the ability to lead to better understanding of the data. Instead of assuming that one cluster definition is better or worse than another, simply embrace the judgments involved and evaluate all potential clusters as knowledge.



## Key Words

---

agglomerative	expectation maximization (EM)
categorical attribute	gap statistic
CLUSTER_COUNT	hierarchical clustering
clustering	K-means
CLUSTERING_METHOD	Logical Primary Key
combinatorial search	mixture model
comma-separated values (CSV)	multicollinearity
correlation coefficient	nominal
Data Source	ordinal measure
Data Source View	orthogonal
dendrogram	principal components analysis (PCA)
distance	responsibilities
divisive	unsupervised learning
eigenvalues	Weka
Euclidean	

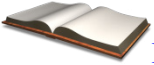
## Review Questions

---

1. What are the common business uses for clustering analysis?
2. What is the best distance measure to use for most clustering problems with continuous data?
3. How are categorical attributes handled in clustering algorithms? Is there a better approach?
4. How is the EM method different from the K-means approach to identifying clusters?
5. How is missing data handled by clustering?
6. How is principal components analysis different from clustering?
7. What features are provided by Microsoft Clustering to analyze results?
8. What features are provided by Weka clustering to analyze results?
9. What is prediction and how is it handled by Microsoft Clustering?

## Exercises

---



### Book

1. Pick a subset of interesting attributes for the car database and set up and run the cluster analysis for them. First use the EM method then run the K-means algorithm. Assign names to the resulting clusters and comment on the differences in the results between the two algorithms.
2. Run the EM clustering analysis for the cars database. Choose five vehicles and run predictions for them. List the clusters predicted for each vehicle, then find the probabilities for each of the other clusters. Comment on the results.
3. Run EM and K-means clustering for the Corner Med patient data. Assign descriptive names to each of the clusters. Comment on differences between the results from the two methods. Hint: Compute `Age=DateDiff(yyyy, DateOfBirth, '12/31/2010')`.
4. Run the Weka clustering tool on the Corner Med patient data using the EM method. Use the cluster visualization tool and comment on the resulting clusters.



### Rolling Thunder Database

5. Run clustering analysis on the Customer data, provide descriptive names to any clusters and comment on the results.
6. Run clustering analysis on the Bicycle data, including at least time to build (`ShipDate-OrderDate`), frame size, model type, year, and construction which represents frame material. Describe the clusters and comment on any results.
7. Run clustering analysis on the components, without the Category attribute. Identify the clusters and comment on how accurately they match the actual categories.
8. Run clustering analysis on the purchase orders. Describe the clusters created and comment on the results and how they might be used to alter purchase decisions.



## Diner

9. Run EM clustering analysis on the diners and describe the resulting clusters. Use prediction tools to determine the most likely cluster and the probabilities of association with other clusters for five different observations.
10. Run K-means clustering analysis on the diners and describe the resulting clusters. Comment on how the clusters might be used to increase sales.



## Corner Med

11. Run EM clustering analysis on the procedures performed for each patient. Describe the resulting clusters and comment on them.
12. Run EM clustering and K-means clustering analysis on the patient diagnoses. Describe the resulting clusters and comment on the differences between the two sets of results.
13. Run EM clustering analysis on the Visit that includes at least the diagnoses, procedure, and drug codes. Describe the resulting clusters and comment on the results.



## Basketball

14. Pick a team and a season and run EM clustering on the players and player statistic averages and describe any resulting clusters.
15. Run clustering analysis against all of the player statistics for one season without including the player's position. Describe the clusters and compare them to the player positions. Comment on any differences.
16. Are some divisions better or worse than others in terms of winning? What about in terms of total points scored? Run clustering by division with won/loss and points scored.
17. Run clustering analysis on the teams and games for one season, without the conference or division, describe the clusters.



## Bakery

18. Run clustering analysis on the products, without using the category. Describe the clusters and compare them to the actual categories.
19. Run clustering analysis on the Sale and SaleItem tables. Convert SaleDate to day of week, month, and split it into time of day: morning, noon, and evening. Describe the clusters.



## Cars

20. Find data on more vehicles and add it to the database. Specifically, find data on at least 20 different trim levels for the existing vehicles. Run EM clustering and compare the results to the clusters found with the original data.
21. Run the Weka hierarchical clustering (Cobweb) on the car data and use the results to comment on the number of clusters that should be chosen.



## Teamwork

22. In the dining database, assign a different month to each team member. Run clustering analysis for each month and compare the results. Comment on any differences.
23. In the basketball database, split the database into teams that made the playoffs and those that did not. Split your team into two groups and assign one set to each group. Run clustering analysis on the player statistics, describe the clusters, and compare the results across the teams.
24. In the bakery database, split the data into years. Assign one year to each team member. Convert SaleDate into month, day of week, and time of day (morning, noon, evening). Run cluster analysis on the sales data, describe the clusters, and compare the results from each team member.
25. In the cars database, add at least one more attribute to the analysis—such as ground clearance or height. Run the clustering analysis and compare the results to the original clusters.

## Additional Reading

---

Ding, Chris and Xiaofeng He, 2004, K-means Clustering via Principal Components Analysis, *Proceedings of the 21st International Conference on Machine Learning, Banff, Canada*. [Highly mathematical but a proof that principal components are related to K-means.]

Kaiser, H. F. 1960, The Application of Electronic Computers to Factor Analysis, *Educational and Psychological Measurement*, 20, 141-151. [The original source of the rule to choose principal components when the eigenvalue is greater than one.]

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman, 2009, *The Elements of Statistical Learning/2e*, Springer: New York. [An outstanding book on data mining, with an emphasis on theory. A graduated-level book that requires a strong mathematics background. Excellent mathematical analysis of the clustering algorithms.]

<http://www.cs.waikato.ac.nz/ml/weka> [Free data mining software from The University of Waikato in New Zealand. The software is written in Java and runs on most computers.]

# Association and Market Baskets

## Chapter Outline

Introduction, 270	<i>Results</i> , 293
Business Situation, 271	Microsoft Association Rules, 294
<i>The Bakery</i> , 272	<i>Goals</i> , 295
<i>Product and Dimension Levels</i> , 273	<i>Data</i> , 295
Model, 274	<i>Results</i> , 296
<i>Goal</i> , 275	<i>Comparing Results</i> , 298
<i>Assigning Values to Rules</i> , 276	Summary, 300
Problems with Dimensions, 282	Key Words, 301
<i>The A Priori Algorithm</i> , 283	Review Questions, 301
<i>Issues in Setting Minimum Support and Confidence</i> , 284	Exercises, 302
Potential Problems, 285	Additional Reading, 305
<i>Simpson's Paradox</i> , 285	
<i>Skewed Support Data</i> , 286	
<i>Continuous Data</i> , 288	
<i>Quantity</i> , 289	
Data, 290	
<i>Database Structure</i> , 291	
<i>Market Basket Structure</i> , 291	
Traditional Tools for Association Rules, 292	
<i>Goals</i> , 293	
<i>Data</i> , 293	

## What You Will Learn in This Chapter

- What products are purchased together?
- What is association or market basket analysis?
- Does association analysis really find useful rules?
- Why do association routines have parameters and what values should be selected?
- Do association rule systems return all of the interesting rules?
- How does data need to be organized for association analysis?
- How difficult is it to create and interpret association rules?
- How does the Microsoft Association Rules tool differ from traditional tools?

### Netflix

Amazon.com demonstrated to other vendors the importance of its recommendation engine. By showing books and other items purchased by other customers, sales can be increased by bringing new items to the attention of customers. Netflix relies on a recommendation system to help customers find movies to rent. Movie rental is an interesting business: millions of people want to rent the same movie, when it is released. Then demand for a specific movie quickly drops off to low levels. Companies that rely on physical media (DVD and Blu-Ray) face the problem of spending money for a large number of initial copies to keep customers happy during the initial rush. (U.S. rental companies are required by law to rent original discs, not copies.) Netflix attempts to manage the demand for new releases through control lists and by encouraging customers to rent other movies. A key step to encouraging the adoption of other movies is the recommendation engine which suggests movies that the viewer might like—based on ratings of previous movies. In late 2006, Netflix created a contest to encourage people to create a new recommendation engine. Anyone who could beat the existing approach by at least 10 percent would win \$1 million. In the end, in 2009, a team of researchers won the prize—by combining a variety of approaches. Most of the researchers began the contest separately, and each achieved small improvements. By combining several approaches and working together, the BellKor Pragmatic Chaos seven-man multinational team, largely led by AT&T Research found a way to beat the older models. In the last 30 days, the other competitors all joined together to form a second team (Ensemble)—which also beat the 10 percent requirement and almost won the prize. [Copeland 2009] It is difficult to determine the ultimate impact of the algorithm on Netflix because many factors have changed over time, but there is little doubt that Netflix is the current leader in movie rentals, and total revenue increased from \$1.7 million in 2009 to \$3.2 million in 2011. [Income statements]

Researchers continue to develop new data mining techniques and better methods for analyzing data. The association problem and recommendation engines are key tools for many businesses.

Michael V. Copeland, “Box Office Boffo for Brainiacs: The Netflix Prize,” *Fortune*, September 21, 2009. <http://tech.fortune.cnn.com/2009/09/21/box-office-boffo-for-brainiacs-the-netflix-prize/>



## Introduction

---

**What products are purchased together?** This classic business question seems easy to answer. It certainly has many applications in business—from determining product placement in a store to making recommendations for additional purchases. Amazon was one of the first vendors to highlight its strength by recommending additional books to customers. Netflix pushed the curve with a million-dollar contest to anyone who created a substantially better recommendation engine. Better answers to the question can improve ordering and stocking decisions, reduce inventory costs, increase sales, and improve advertising targets.

The one catch is that the problem is relatively hard to answer—particularly with giant datasets available to retail merchants. The best computational algorithms are amazingly fast, but they gain speed by making simplifying assumptions. Still, the market basket algorithms are useful. Plus, once the data is organized correctly, the routines are almost completely automatic. The tools in this chapter can be run as **unsupervised learning** systems. Additionally, the output is relatively easy to understand, where associations are described as **rules**, such as customers who purchased Pies are also likely to purchase Cakes, and the strength of the association is given a number expressing the degree of confidence in the rule.

The basic output of association analysis consists of a set of rules of the form  $A \rightarrow B$  which can be read as “A implies B” or in market terms: If a customer buys A, then they are likely to buy B. The rule also contains one or two numbers describing the strength of the relationship. Often the numbers are conditional probabilities, describing the frequency in the database where B followed A.

Dimensionality is the biggest issue faced with association tools. A sales database could have billions of rows of sales, with hundreds of thousands of products. The number of rows is less of a problem than the number of products (dimensions). Rows generally increase processing time linearly. Dimensions increase processing time exponentially. Even if an algorithm and system are fast enough to compare and identify relationships among so many products and sales, how can a human ever evaluate or understand the resulting rules? Hence, a key step in most association analyses is to find a way to reduce the complexity of the data. For instance, when examining sales in a hardware store, does it really matter if a customer buys 3-inch nails versus 2-inch nails? Or, is it sufficient to know that the customer purchased nails (with that hammer)? In other words, the analyst has to determine the appropriate product level for the research. If the size of the nails is unimportant, the analyses can be conducted at the level of product category. If the size is critical, the problem can be reduced by looking at fewer items, instead of the entire inventory.

The point is that analysis often has to be performed at multiple levels. In the hardware example, with tens of thousands of product variations, a high-level analysis might examine the categories of items purchased (hammer, nail, board, and so on). Then, departmental managers might want to examine sales at more detailed levels. For instance, did people buy the right hammers for the nails purchased? In this case, a lower-level analysis can look at the detailed items (SKUs) by limiting the items to those within a specific department.

Most of the examples in this chapter are oriented towards traditional market basket analyses—because the results are easy to understand. However, keep in mind that the tools can be used for more complex problems. Think in terms of association—identifying which events tend to happen together. In this context,

the tools could be used to examine things such as customer attributes, stock price events, medical treatments, and even crime events.

## Business Situation

---

**What is association or market basket analysis? Data associations** can be defined loosely as events or situations that tend to happen together. In business, these events are often captured as transactions. In an accounting context, transactions typically consist of sales and purchases. **Market basket** analysis is a subset of association analysis that focuses specifically on products that were purchased at the same time—in the same market basket. In a broader context, the events could include almost anything—such as responses to marketing campaigns (marketing), product failures in different environments (operations), sales of stock to various groups (finance), security attacks or technical support requests (MIS), and evaluations and promotions (HRM). Sales and purchases (costs) are the most commonly studied business events—because the data is readily available. Bar code scanners automatically capture the list of items purchased by each customer in one sale. The detailed data is generally available in giant databases organized by store, date, region, and sometimes by customer—when customers use frequent-buyer cards. An increasingly popular example is to use association rules as the foundation of a **recommendation engine** such as those used by Amazon to suggest similar books and Netflix to recommend movies.

The classic example of market basket data mining might not be true (no one seems to know who actually ran the study). Nonetheless, it represents an interesting example of the potential results. Supposedly a chain of stores (e.g., 7-eleven) analyzed sales on weekend nights and the association tool found a strong rule that baby diapers and beer were commonly purchased together. Presumably father went out for diapers and picked up beer at the same time. The story sounds plausible, and lacking real data to test it (or a *Mythbusters* evaluation), it is likely to remain as the classic example. The beauty of the story is that (1) it could be true, (2) the results are something that could be found through market basket analysis, and (3) everyone can understand the implications. So, even if it is not true, it makes a good example. If you run a market basket association analysis and see a rule of the form (diapers => beer), the manager needs to think about what the rule means in business terms. For example, moving beer and diapers to the same aisle might increase sales of both. Or, perhaps putting snack items between the two products would lead to increased sales of snacks because of the improved visibility to these customers. The key point is that identifying potential rules is a good first step, but background knowledge of the topic is required to understand the value of the rule and put it to good use.

One of the issues quickly faced by anyone running association analysis is that the tools tend to turn out dozens, hundreds, or even thousands of rules. Some of the rules are interesting, some are not. Some can be used to increase sales or reduce costs, but others are simply curiosities. It is important for managers to read the rules and determine which relationships make sense for the particular situation. Fortunately, the tools are easy enough to set up and run that a manager with little formal training in data mining can make use of the tool. Today, some specialized firms provide data mining services where they take copies of the sales data and run the results on their customized high-speed servers. So managers only deal with the results. Of course, these firms charge fees, so most smaller problems could be handled without them—as long as the amount of data is reasonable.

Bread	Muffin
Cake	Pastry
Candy	Pie
Cookie	Rolls
Crepes	Scone
Cupcake	Sweet Rolls
Miscellaneous	

**Figure 6.1**

Categories for the bakery. Each category has many products. For example, pies can be chocolate, berry, lemon, coconut, and so on. Categories and important method for reducing dimensionality.

Market basket rules for shopping need a fairly large amount of data to estimate and the products need to be reasonably consistent over the time frame of the analysis. The Rolling Thunder Bicycle case has plenty of data, but the actual products change every couple of years as the vendors put out new versions of components. Additionally, the bicycle component data is usually constrained by the groups—customers almost always buy all of the products within one of the primary Shimano or Campagnolo groups. Consequently, the strongest associations are for products within the groups—which is a dull result.

## The Bakery

To provide a more interesting set of results—particularly for association—the bakery case was created with simulated sales over many years. A bakery is similar to the classic market basket problem: Many different products, but they fall into defined categories. Figure 6.1 lists the categories for the bakery—they include the common items encountered at most bakeries. Each individual product is classified into one of the categories. For example, the bakery produces 22 varieties of cakes including White, Angel, Sponge, Carrot, and German Chocolate. The initial bakery data contains over 140 individual products, but they are grouped into these 13 categories. The categories are important in determining association rules. As explained in the Model section, having too many dimensions makes the problem exceedingly difficult to solve—even with high-speed computers. Grouping products into a smaller number of categories speeds the analysis.

In reality, 140 items is a relatively small number and can be handled by the systems in a short time. More importantly, it is unlikely that important relationships exist at the detailed product level. And if they do exist, is it possible to determine the meaning? What does it mean if a rule states that customers who buy Spice cake often buy chocolate chip cookies? And, how would managers deal with the potentially hundreds or thousands of similar rules? For books or movies these detailed associations would be useful. If a customer buys a specific book or movie, the customer will want to see specific other books. But at the bakery, if a customer buys a Spice cake, it makes more sense to suggest cookies in general instead of just one variety of cookies.

Cookie -> Muffin	(13.3, 30.8)
Rolls -> Muffin	(10.5, 24.4)
Cake -> Pie	( 8.1, 19.7)

**Figure 6.2**

Sample rules for bakery. The rule is read from left-to-right, so that customers who purchased a cake tended to also purchase a pie. This rule lists support of 19.5 percent with a confidence of 41.5 percent.

Figure 6.2 shows sample results based on a limited set of data for the bakery categories. The **association rules** are read in the direction of the implication arrow (->). A few tools write the rules from right-to-left instead, but left-to-right is easier to read. For example, the first rule states that customers who bought a cookie were likely to purchase a muffin. The numbers in parentheses are the support and confidence values which are explained in detail in the Model section. Essentially, the support is the frequency of that combination appearing in the sample. So 13.3 percent of the transactions included purchases of both cookies and muffins. The confidence is an estimate of the conditional probability of the outcome given that the first event has already occurred. For instance, the probability that someone will purchase a muffin given that a cookie has already been purchased is estimated to be 30.8 percent. Of course, the numbers apply only to the sample and there is no guarantee they will apply to future purchases. This point is particularly critical if other factors might be influencing the choices.

These sample rules are relatively simple in that they represent pairs of item categories. More complex rules can include multiple categories on the left side, such as

Scone, Sweet Rolls -> Muffin

This sample rule reads: customers who purchased both a scone and sweet rolls were also likely to purchase muffins. In theory, any combination of the dimensions could form the **antecedent** (left side) of the rule. Multiple dimensions quickly lead to a huge number of possible combinations. Beyond the computational problems, the rules are also difficult to understand. Picture the challenges of examining the bakery data with 141 products if the results led to rules with 100 items in the antecedent list. Now think about a modern superstore with hundreds of thousands of detailed products and imagine the potentially convoluted results.

## Product and Dimension Levels

The analyst must choose the level at which to conduct the analysis to match the business problem. This decision can be based on business factors with input from managers. Of course, in many situations it is possible to conduct the analysis at several levels. Start at the top (category) level and work down to more detail. For huge stores, detailed analyses can be conducted within groupings. For instance, in a discount store it might make sense to do market basket analysis just within the toy department. Customers who purchased a specific doll might also purchase specific clothing or accessories. Working within the department would miss rules that cross boundaries. Perhaps those customers who bought that doll and clothing

<u>1132</u> , 1053, 893, 757, ...	Store	Are sales at one store associated with sales at another store?
<u>Toys</u> , Garden, Auto, Shoes, ...	Department	Do sales in one department increase sales in others? Good for loss leaders.
<u>Dolls</u> , Balls, Trucks, Stuffed, ...	Category	Are categories of items commonly purchased together? Product display.
<u>Girls</u> , Boys, Baby, Fashion, ...	Subcategory	Do subcategories affect each other? Handled within departments.
<u>Barbie</u> , Princess, Corolle, ...	Product	Which items are purchased together? Usually within department or category.
Thumbelina, Supergirl, ...	Size, Color	Which color items are associated? Useful for clothing.

**Figure 6.3**

Sample levels of analysis. Start at the top and work down to more detail. Detail levels might have to be run within a sublevel instead of across the entire data set. For example, product sales can be examined within a department, but probably not at the store level.

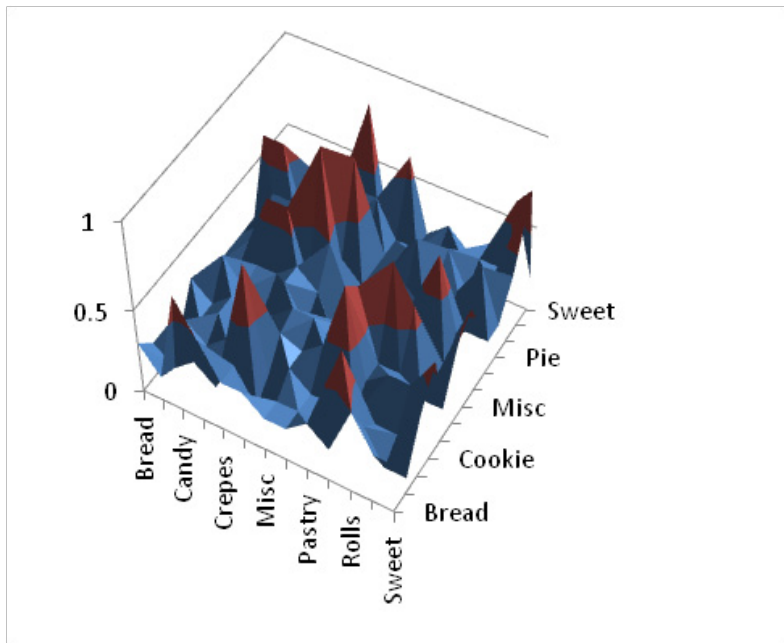
also purchased cookie dough at the same time. But, because the store is organized in departments, even if some unusual cross-departmental patterns exist, it would be difficult to act on them. Figure 6.3 presents some examples of levels for a department or discount store. The questions to be answered vary by level or grouping. The Figure provides some examples to highlight the differences, but the list is endless.

Product hierarchies are important in most businesses. Simply to keep the data easier to understand, most organizations section the data into groups. Products are sold by country, region, store, and department. Products are grouped into departments, manufacturers, and subcategories. Individual items often have multiple sizes or colors. Sometimes the variations get out of hand. Does any store really need to carry 30 or more varieties of Oreo cookies? All of these groupings are useful candidates for aggregating sales before attempting to run the association analysis. The results for each grouping can provide different interpretations. Are customer choices of vendors related? Do some departments bring in customers who then buy profitable items from other areas?

Hierarchies exist in most other organizations—because humans routinely use them to reduce complexity. Workers are organized into groups—do some groups affect outcomes (e.g., quality) with other groups? Are some managers more closely associated with other managers in terms of assigning raises? Similarly, finance divides investments into categories, including broad categories of stocks, bonds, and derivatives. Each major grouping can be divided into sectors, duration, rating, or other categories. Starting with the top categories can provide guidance on where to look within sublevels.

## Model

**Does association analysis really find useful rules?** This particular question is probably the most important one regarding association analysis, but it is extremely difficult to answer. One approach might be to say that based on ex-



**Figure 6.4**

Frequency for pairs. The peaks or bumps show points where items are often purchased together. The chart shows the correlations between categories. The goal of association analysis is to find the peaks and identify interesting pairs.

perience, yes, association analysis has found useful rules in many situations. But, at heart, association analysis searches for patterns that arise often in the sample data set. There is no assurance that high frequency is associated with usefulness or even interestingness. For example, a sales analysis of a fast food chain would most likely reveal that people who purchase burgers (or almost any other entrée) also purchase fries. Having the sales clerks always ask “Would you like fries with that?” is going to skew the results. But, that is the whole reason for continually asking customers—to increase the sales of profitable products. Is there usefulness or interestingness or surprise in having a data mining tool find that relationship? Perhaps confirmation is important. But, what if that single rule is so dominant that it hides other associations?

The fry association is a fairly blatant example, but similar outcomes can arise in other industries. More importantly, the question and the example highlight the key aspect of association rules: The tools need to assign numbers to the relationships that have value to managers, are easy (fast) to compute, and can sort and differentiate the rules. These questions are covered in the model section, and each tool has methods of trying to handle the complex interactions.

## Goal

Ultimately, the goal of association data mining is to find “bumps” in the frequency distribution of the transactions data. The objective is to find those points or combination of items that stand out from the others. Figure 6.4 illustrates the concept by

Measure	Definition
Support	$P(A \text{ and } B)$
Confidence	$P(B   A)$ $P(A \text{ and } B) / P(A)$
Lift	$P(A \text{ and } B) / [P(A)P(B)]$
Relative Risk	$P(B   A) / P(B   \sim A)$

**Figure 6.5**

Traditional measures and common definitions. A and B represent events or itemsets of products. Support is the number of times the events occur in the sample divided by the total number of transactions (frequency). Confidence is an estimate of the conditional probability. Lift is an estimate of the gain that might be expected for B if event A occurs.

comparing sample sales for pairs of categories. The peaks are likely to be the most interesting points to business managers. They represent combinations of product categories that are often sold together. Note that this figure represents the simplest combination of categories—pairs. And it still requires combining every category with every other category and finding the resulting correlation data for each point.

Even if it were possible to create a multidimensional picture similar to Figure 6.4 for every combination of categories, it would still be difficult to interpret. The visualization of the peaks is interesting, but if hundreds or thousands of peaks (rules) appear, it would be difficult to examine or understand them. Instead, the data mining tools assign numbers to the combinations which enable the analyst to sort and compare the various rules. The numbers are also commonly used to cut off the display lists to keep them to manageable size. Hence, it is important to understand what the numbers mean because they control which rules are displayed.

### Assigning Values to Rules

At heart, association rule algorithms examine combinations of attributes, assign a value, and then display the combinations with the highest values. It is then up to the analyst and managers to decide if the resulting rules make any sense, and can inspire changes that will increase sales and profits. Yet, the two main steps of the algorithms leave plenty of room for differences among tools. They differ based on how the search is performed and on the measures used to value each combination. The measurement issue is the most important, because it affects the ultimate question of whether the rules are meaningful. The objective is to find an easy-to-compute value that identifies interesting and meaningful rules for managers.

Several measures are used for identifying frequent combinations. The confusing part is that data mining research has created names for them, but some systems use the same names to represent different calculations. Hence, it is helpful to look at the mathematical definitions because they are straightforward and easier to understand when comparing tools. Figure 6.5 shows the most common measures used by association tools. The first two were defined in the original association analysis paper by Agrawal (1995). The relative risk term is less common, but it is a key element in Microsoft's association model. This list is a small portion of the concepts that have been proposed. Geng and Hamilton (2006) provide a



good introduction to the concept of **interestingness** measures and list almost 30 variations.

Combinations of attributes or products are called **itemsets** and the letters A and B are commonly used to represent different itemsets. In market basket terms, an itemset consists of combinations of different products. Typically, the A itemset is the antecedent of a proposed rule, and the B itemset (often a single item) is the proposed result. So, a rule might be evaluated that proposes the presence of apples (A) leads to the purchase of bananas (B).

### *Support*

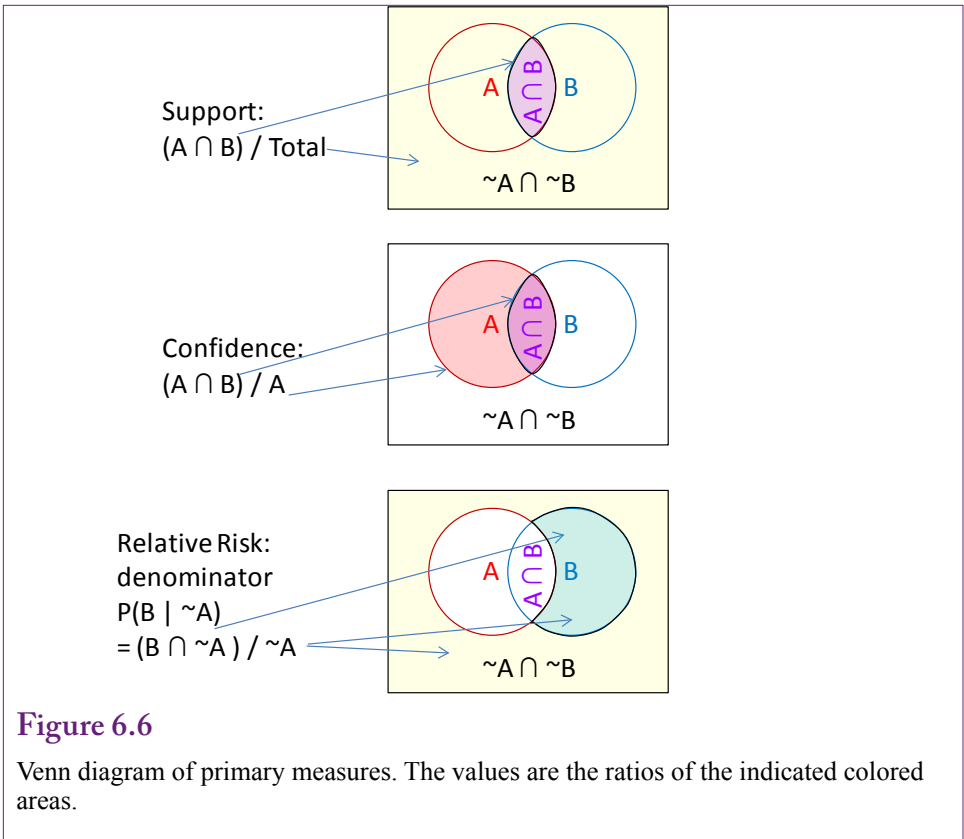
**Support** is a critical measure. It is the frequency of the itemset within the sample: The number of times the specified set occurs divided by the total number of observations. Support can be computed by counting the number of transactions containing all of the itemsets specified in the parameter. Support is sometimes denoted with a sigma function, such as  $\sigma(A)$ , but because it represents an estimate of the probability, it is most commonly written as a probability measure  $P(A)$ . In association analysis, the most important support measure is  $P(A \cap B)$  which is the probability or frequency of both A and B itemsets appearing in transactions together. In words, support represents the relative number of times the proposed rule actually occurs in the sample. So, high support values indicate that the combined itemsets often appear together. Support is a critical measure in most of the data mining algorithms. It is one of the primary tuning parameters used to reduce the number of rules to examine. Potential rules with low support are discarded from the analysis.

### *Confidence*

Remembering probability rules, the two itemsets can be interchanged and the support level will be the same:  $P(A \cap B) = P(B \cap A)$ . So how can a rule claim that one product leads to the other? By itself, support cannot. Support simply indicates that the two itemsets occur together, it says nothing about which might lead to the other. On the other hand, **confidence** is an estimate of  $P(B | A)$  and equals  $\text{support}(A \text{ and } B) / \text{support}(A)$ . Because of the divisor,  $P(B | A)$  is different from  $P(A | B)$ . The conditional value is interpreted as the probability that itemset B will arise given that itemset A has already happened. Hence the name (conditional). High values of the conditional probability provide confidence that the combination of events will happen in future transactions. Because of its one-way nature, confidence is a better measure of interestingness than support.

### *Lift*

It is possible to build an association data mining tool using just support and confidence measures. The results are ranked by confidence values and they are relatively easy to interpret. However, most tools include more measures. Lift is a common term that attempts to measure the impact of the rule. **Lift** is best defined as  $P(B | A) / P(B)$ . This ratio measures the probability of item B being purchased with the rule (A already chosen) versus without the rule—B by itself. Values greater than one indicate a positive effect, and values less than one reveal that the presence of itemset A reduces the probability of purchasing itemset B. The ratio is read as a measure of the probability gain. For example, if  $\text{lift} = 1.2$ , then the presence of the rule and itemset A leads to a 20 percent increase in the probability that B will be purchased, compared to a customer who does not already have itemset A in the basket. This estimate can be used to decide how much value might be gained by



convincing customers to purchase itemset A. The accompanying 20 percent probability increase can be used to estimate the expected gain in sales of B, hence the expected gain in profit. A portion of this expected gain could be spent on marketing A, reorganizing the store or Web site to increase sales, or decreasing the price of A.

### Relative Risk

**Relative risk** is another of many interestingness measures. It is included in this chapter (versus the 20 or 30 other measures not included) because it is a major element in Microsoft association analysis. It is conceptually similar to lift because it is trying to measure the gain in probability of purchasing B when itemset A is present. The difference is in the comparator. Lift compares the gain to the total probability that B is purchased. Relative risk computes the gain from the probability that B is included when A is not purchased. The formula for relative risk is:  $P(B | A) / P(B | \sim A)$ . Recall that lift just used  $P(B)$  in the denominator. Here,  $P(B | \sim A)$  is the probability that B is purchased given that itemset A is not ( $\sim$ ) present. The interpretation is similar to that for lift. The ratio measures the relative difference in the probability of purchasing B with and without A present. If the ratio is larger than one, then the effect of A is positive, otherwise it is negative. The ratio measures the percentage gain in the probability of purchasing itemset B when itemset A is present.

For the most part, lift and relative risk are comparable. However, relative risk does not work well if A and B consist of all possible items. If it seems unlikely

Pie	Sweet Rolls	Cake	Sweet Rolls
Sweet Rolls	Cookie	Cookie	Muffin
Pie	Pastry	Cookie	Sweet Rolls
Pastry	Cookie	Cookie	Muffin
Cupcake	Cake	Bread	Pie
Cookie	Sweet Rolls	Sweet Rolls	
Muffin	Cake	Cake	Pie
Cake	Cupcake	Pastry	
Rolls	Cake	Pie	Pie
Pastry	Cake	Rolls	

**Figure 6.7**

Small sample of 10 transactions and 6 categories. Sample single support: Muffin shows up in 3 of the 10 transactions for a support of 0.3. As an example of pairs, Pie and Cake appear together 4 times for a support of 0.4.

that someone would conduct data mining on only two items, remember that A and B are itemsets, and itemset A could consist of everything except B. A little probability manipulation is needed. Look at the denominator:

$$P(B | \sim A) = P(B \cap \sim A) / P(\sim A)$$

$$P(\sim A) = P(B \cap \sim A) + P(\sim B \cap \sim A)$$

If A and B are all the items, then  $P(\sim B \cap \sim A) = 0$ , so  $P(B | \sim A) = 1$

Reducing relative risk to  $P(B | A)$  or confidence.

Ending up with relative risk equal to confidence is not a major disaster, but it does change the interpretation of the concept. Just keep in mind that if the itemsets A and B are large and represent a substantial portion of all transactions, relative risk is less useful.

### *Venn Diagram*

Sometimes visual depictions of the concepts are easier to understand and remember. Besides, if the SAT was the last time you saw Venn diagrams you should know that they are relevant and occasionally useful. Figure 6.6 uses three Venn diagrams to show how the measures are calculated and how they are different from each other. Note one key aspect of the diagrams is the explicit inclusion of the  $\sim A \cap \sim B$  component—the itemsets that are in the universe of transactions but not in either A or B sets.

Support is the percentage of transactions that include both A and B (the intersection of the circles), divided by the total number of all transactions. The comparison with confidence is clear because although the numerator is the same, the denominator for confidence is all of circle A and nothing else. The computation for lift is not shown because it is a different type of ratio (not a percentage). It is the confidence value divided by the B over everything ratio.

Likewise, only the denominator for relative risk is shown in the diagram. The numerator is the part of B that is not included in A. The denominator is everything

Support	Cake	Pie	Sweet Rolls	Cookie	Muffin	Pastry
Cake	0.1	0.4	0.1	0.0	0.1	0.2
Pie	0.4	0.1	0.2	0.1	0.1	0.1
Sweet Rolls	0.1	0.2	0.2	0.3	0.1	0.1
Cookie	0.0	0.1	0.3	0.2	0.2	0.2
Muffin	0.1	0.1	0.1	0.2	0.0	0.1
Pastry	0.2	0.1	0.1	0.2	0.1	0.0

**Figure 6.8**

Support values for pairs of categories. The values are derived from simple counts of the transactions containing both items. The highlighted values are greater than 0.2 and would be the initial cut off for search algorithms.

not in  $A$ , so it includes the numerator part plus all of the transactions outside both  $A$  and  $B$ . It is clear from the diagram that if nothing exists outside of  $A$  and  $B$ , then the denominator for relative risk reduces to one (the green part of  $B$  divided by the green part of  $B$ ).

### Examples

With a small number of dimensions, all of the measures presented in this section are relatively easy to calculate. Actually, computability is an important aspect of the measures. The search algorithms spend a huge amount of time retrieving, sorting, and comparing combinations of dimensions. If the value computations are extensive as well, performance will fall dramatically. Figure 6.7 shows a tiny sample of 10 transactions with 6 categories. Each row contains data for a single transaction or basket. The items purchased in each transaction are listed in columns. As an example of support for a single item, the Muffin entries are highlighted to yield a support of 0.3 for muffins (3 of 10 rows contain muffins). The support values for the individual categories are: Cake=0.6, Pie=0.5, Sweet Rolls=0.4, Cookie=0.4, Muffin=0.3, and Pastry=0.4. The simplest rules consist of pairs, so the transactions that contain both Pie and Cake are highlighted, showing a support of  $P(\text{Pie} \cap \text{Cake})$  of 0.4 or 40 percent.

Figure 6.8 shows the support values for all of the pairs of categories. It is created from simple counts of the transactions that include each pair. Even that simple table requires some effort to create by hand. The highlighted values are greater than 0.2 and would be the ones that would be selected for further analysis. Most algorithms have an option to set the cut off level. Only items above this threshold are analyzed further.

Figure 6.9 shows the computed confidence values. The easiest computation method is to take the support values for each pair and divide by the individual support value for the row item. For example,  $P(\text{Cake} \cap \text{Pie})/P(\text{Cake}) = 0.4/0.6 = 0.67$  for the row=Cake, column=Pie entry. The values greater than 0.4 are highlighted. Again, the analyst sets this cutoff value to keep the number of displayed rules to a manageable level so that the most valuable ones are easier to find. However, most systems apply the support rule first and only compute confidence (or perhaps lift) for the items that pass the first test. Hence, only the four rules highlighted in Figure 6.8 would be evaluated and displayed.

Confidence	Cake	Pie	Sweet Rolls	Cookie	Muffin	Pastry
Cake	0.17	0.67	0.17	0.00	0.17	0.33
Pie	0.80	0.20	0.40	0.20	0.20	0.20
Sweet Rolls	0.25	0.50	0.50	0.75	0.25	0.25
Cookie	0.00	0.25	0.75	0.50	0.50	0.50
Muffin	0.33	0.33	0.33	0.67	0.00	0.33
Pastry	0.50	0.25	0.25	0.50	0.25	0.00

**Figure 6.9**

Confidence values for pairs where row  $\rightarrow$  column. Values greater than 0.4 are highlighted. But only the items that first passed the support criteria would be evaluated.

Figure 6.10 shows the computed lift values for all of the pairs. The lift can be computed as the confidence divided by the support for each column. For example,  $\text{Confidence}(\text{Pie}|\text{Cake})/P(\text{Pie}) = 0.67/0.6 = 1.33$ , which is the lift for the Cake  $\rightarrow$  Pie rule. The four highest lift values are highlighted. Notice that only the Cookie and Sweet Rolls rules are also highlighted in the support table. This difference is important because many tools use the support rule to cut off further analysis. Hence, the Cookie  $\rightarrow$  Muffin and Muffin  $\rightarrow$  Cookie rules with their relatively high lift will probably not be displayed unless the support cutoff is dropped to a low enough level.

The point of the examples is to illustrate how the standard numbers of support, confidence, and lift are computed. For pairs of items, the numbers are relatively easy to compute. Keep in mind that itemsets can consist of any combination of categories. The example also highlights the difficulties in finding the important relationships within the data. The values enable automated search systems to work, but they still require interpretation and some guidance by the analysts and managers.

### *Summary of Values*

It is important to understand why different measures exist. The goal is to find a measure that can find the interesting itemsets in the data. The challenge is that interestingness is hard to define. It might be highly subjective, and several researchers are trying to find useful ways to automatically incorporate subjective measures from managers. Interestingness might also require surprise value—finding patterns that indicate a rule is unexpected or different from typical rules. Researchers are working in all of these areas and more. But, for now, the tools generally rely on the primary measures: (1) support, (2) confidence, and (3) lift. Variations exist among these three measures, but the conceptual goals are similar. Just be careful when evaluating results from different tools—be sure to verify the exact definition of the measures used. These concepts will become clear in the examples later in this chapter.

Lift	Cake	Pie	Sweet Rolls	Cookie	Muffin	Pastry
Cake	0.28	1.33	0.42	0.00	0.56	0.83
Pie	1.33	0.40	1.00	0.50	0.67	0.50
Sweet Rolls	0.42	1.00	1.25	1.88	0.83	0.63
Cookie	0.00	0.50	1.88	1.25	1.67	1.25
Muffin	0.56	0.67	0.83	1.67	0.00	0.83
Pastry	0.83	0.50	0.63	1.25	0.83	0.00

**Figure 6.10**

Lift values for rules Row  $\rightarrow$  Column. The four highest values are highlighted. Only the Cookie and Sweet Rolls rules are also highlighted in the support table.

## Problems with Dimensions

**Why do association rule routines have parameters and what values should be selected?** Association or market basket analysis is one of the data mining tools that is highly sensitive to the number of dimensions. Remember that rules consist of two itemsets  $A \rightarrow B$ , where the set of items on the antecedent or left (A) implies the set of items on the **consequent** or right side (B). With  $d$  total dimensions (or items), these itemsets can consist of any combination of 1, 2, 3, ...  $d-1$  attributes. In a full comparison, each possible itemset on the left should be compared with every possible itemset on the right. As Tan (2005) notes, some algebra reveals that this full comparison using brute force amounts to  $3^d - 2^{d+1} + 1$  comparisons. To reduce the number of computations, most systems assume problems are somewhat simpler; so most tools assume that the predicted side (B) contains only one element. Still, the left side (A) consists of every combination of 1, 2, 3, ...  $d$  items; and the sum of all of these combinations is  $2^d$ . The point is that the number of rules to be searched is exponential in the number of dimensions. For example, the six categories in the example problem would require  $2^6$  or 64 comparisons, which is relatively small. A problem with only 20 dimensions quickly jumps to  $2^{20}$  or one million comparisons. How many problems have more than 20 dimensions? Even if the problem is reduced by using departments and categories, there could be hundreds of dimensions. Even if  $d$  is 100,  $2^{100}$  is a huge number ( $1.27 \times 10^{30}$ ). Forget about trying to search all of those combinations. The problem is sometimes known as the **curse of dimensionality**. Now think about trying to compare every combination of products in a huge discount store. With existing computing systems, these problems are not feasible if the computer has to evaluate every possible combination of dimensions.

Two solutions exist to this problem. The first is efficient search algorithms as explained in the following paragraphs. The second solution is to reduce the number of dimensions. No matter how efficient the algorithm or how fast the computer, at some point it becomes necessary to reduce the number of dimensions being compared. Typically, that means switching from product-level comparisons to category comparisons. Similarly, analysts often restrict the number of products to those that fall within a specific set—such as comparing detailed items only within a department. In cases where detailed product comparisons are needed (recom-

mendations for books and movies), the algorithm needs to be restricted to examining the data just as single pairs, which requires in  $d^2$  comparisons.

## The A Priori Algorithm

It is helpful to understand a few concepts of the most common association algorithm. The most critical aspect is that parameter options are used to control the level of the search. The tools have default values for these parameters that try to strike a balance between finding useful rules and running within a reasonable time. These parameters are related to the issue of too many dimensions, but the parameters are based on the value computations that attempt to measure interestingness. In practice, if the model runs within short time frames, it is possible to experiment and try different parameter values. However, for large problems, it is important to begin with conservative values to first see how many rules are returned and to determine how long the search takes.

Agrawal (2005) first developed the **a priori algorithm** which opened the door to large-scale association analysis and remains one of the most popular algorithms in use. Yet, it is not perfect, so most tools modify it to improve performance. Also, many variations exist largely because of the number of possible measures that can be used to evaluate interestingness. The basic original model is briefly described here. From that point, it is easy to see the variations.

The basic process of the a priori algorithm is to start with a seed set of candidate itemsets that occur frequently in the dataset on the assumption these might lead to strong rules. The computer then goes through all of the transactions and computes the value (typically support) for each rule and keeps the ones that are high enough. A new set of seed itemsets is generated from the ones that pass the threshold test and the process repeats until the possible itemsets have been searched. The critical feature of the algorithm is that new candidate itemsets are based only on the prior itemsets that are sufficiently strong.

The algorithm begins its search by looking only at single-item sets. It keeps the set only if its support exceeds some threshold value. Consequently, this cutoff value is the most critical parameter of the method. Any potential rules that occur less often than the specified value are discarded from further analysis. Remember that support is the frequency or percent of times that the itemset appears in the transactions. So, if the minimum support level is 0.4, and some category or dimension does not occur in at least 40 percent of the transactions it is tossed.

After the single-item sets have been evaluated, the algorithm moves to the combination of two-level sets. But, if category A (say apples) appears less than 40 percent of the time, then it is clear that searching for A with any other set will also be less than 40 percent. Subsets are never greater than the original set, so adding B (bananas) to the condition will reduce the number of potential transactions. In the example, if apples appear in less than 40 percent of the sales, then the combination “apples and bananas” cannot appear in 40 percent either. Consequently, any single-level itemset thrown out at the first step will not be used at other levels. This decision substantially prunes the list of itemsets to be considered. The process continues, so that only two-level itemsets that are significant are examined for three-level items and so on.

The algorithm still needs to iterate through all of the data at each step to compute the support values for the candidate itemsets. If the transaction dataset consists of millions or billions of observations, this step can be time consuming. The



a priori algorithm still requires several passes through the data. This is one area where other algorithms have found substantial improvements in performance.

Once all of the itemsets have been found that have support values above the minimum level, the algorithm loops through the data set one more time to compute the confidence value for each of the remaining rules. These values are often sorted and the results displayed in terms of rules in descending order of confidence. Most algorithms also specify a cutoff value for the confidence level.

## Issues in Setting Minimum Support and Confidence

All tools that function similar to the a priori algorithm use two critical parameters: (1) **minimum support**, and (2) **minimum confidence**. Some systems use different terms, sometimes different definitions, and occasionally different values; but the overall concepts and tradeoffs are the same. At the start, the analyst must specify the cutoff values. Any potential rule falling below the threshold will be discarded. Because of the way the algorithm works to reduce the number of itemsets, all subsets of that itemset will also be discarded.

Choosing a low value for minimum support will keep many more itemsets and potential rules in the analysis. But, particularly with huge transaction sets, a low support threshold exponentially increases the number of itemsets to be considered and it might be impossible to examine all of the requested possibilities in a reasonable time.

Selecting a low value for the second parameter (usually confidence, but sometimes lift) is less drastic. It is used to limit the number of rules displayed. If a low value is selected, the system might attempt to display hundreds or thousands of rules. But most systems sort the list in descending order. Many tools also export the rules to a database and provide options to search and sort the rules on various criteria. Overall, the effect on performance is small and it is just a question of analyst time to deal with the extra potential rules.

So what values of support should be selected? Will some values work in all cases? The answer to these questions is that each problem is unique. Values that work in some cases will be unhelpful in others. A better way to approach the issue is to determine a process for finding good values of support and confidence with each problem. The first step in the process is to recognize that for small problems, the impact of the choices is relatively minor, but large problems can result in long computation times unless parameters are chosen carefully.

Small problems can be rerun quickly, so analysts can experiment with different values until they are happy with the resulting rules. But how do you know if a problem is small? The number of dimensions is one important measure, but there is no specific rule defining small and large. Plus, computational time depends on the hardware as well. So, in most cases, it is best to begin by assuming any problem could be big.

Large problems typically have many dimensions and huge transaction data sets. Always remember that the support cutoff value is a critical element used to reduce the number of itemsets to be searched. Setting a value that is too low can result in huge performance penalties. Consequently, the process for large problems is to start with a relatively high value for the minimum support and then slowly reduce it to add more rules to the results. The confidence parameter can start relatively low since it primarily controls the output display. Some algorithms begin with minimum support values of 40 or even 60 percent. With a large number of dimensions and evenly distributed sales, it is possible that no rules will exceed this ini-

tial threshold. But, the analysis will run quickly. At that point, the minimum support can be reduced and the analysis rerun. But, be careful to leave the confidence minimum relatively low. If confidence is set too high, the support parameter might be at a reasonable level, but the confidence minimum is blocking the display. The first goal is to find a reasonable value for minimum support, the confidence minimum can be found as a second step.

There is no solid rule for decrementing the support cutoff, but larger problems require more cautious changes. Values of ten percentage points might be reasonable at the high end. However, support is rarely evenly distributed. Typically a few dimensions will have high support values, then when support drops below a certain point, almost all of the dimensions will come into play at the same time. For example, a dataset might contain a handful of dimensions with 40 percent support, a few more at 30 percent, and then almost half the dimensions at 20 percent support. Even small changes in the minimum support could suddenly result in an intractable problem. For large problems, it would be useful to compute and display the support values for each of the individual items before proceeding with the analysis. A fairly straightforward query can be used to count the number of baskets (rows) containing each item. With this information, the analyst can begin with a cutoff value that will include a few of the top-most items.

Once the support cutoff returns a reasonable number of rules, it is possible to increase the confidence threshold to reduce the display to those that are potentially more important and more interesting. Some systems provide interactive queries to make it easier to explore the rules that pass the support test. The key is to find rules that are interesting and provide insight into the underlying process. Because no single measure conveys all of this information, the process becomes subjective and requires analysts and managers to evaluate the rules individually. Typically the rules with higher importance values make the best starting points.

## Potential Problems

---

### Do association rule systems return all of the interesting rules?

The obvious answer is that it is not possible to completely automate the search process, so some rules are going to be missed. Random errors are always going to occur. The bigger issue is to identify specific types of problems that might arise. Two of the biggest problems arise from Simpson's paradox and skewed data. Continuous data is also an issue that needs to be considered for many problems. All of these issues commonly arise in practice and you need to be able to recognize the potential problems.

### Simpson's Paradox

Simpson's Paradox is named after E.H. Simpson (1951), but the concept was described earlier by Yule (1903) and it is not exactly a paradox, so it is sometimes named the Yule-Simpson effect. **Simpson's paradox** states that comparisons or associations for groups can be reversed when the groups are combined. A classic example consists of men and women applying for jobs, or for admission to programs at Berkeley as revealed in a lawsuit explained by Bickel (1975). Figure 6.11 shows a simple example with hypothetical data. In total, men were hired 73.9 percent of the time versus 65.8 percent for women. But, looking at the detailed departments, in every case, women were more likely to be hired than men. How could this result arise? The difference exists because the majority of the women applied for Department A, which had a much lower overall acceptance rate.

	Men		Women	
	Hired	Applied	Hired	Applied
Dept A	1	3	18	30
Dept B	16	20	7	8
<b>Total</b>	17	23	25	38

	Men	Women
Dept A	33.3%	60.0%
Dept B	80.0%	87.5%
<b>Total</b>	73.9%	65.8%

**Figure 6.11**

Example of Simpson's paradox. In each detail department, women were accepted at a higher rate than men. But the rate for the aggregate total is reversed.

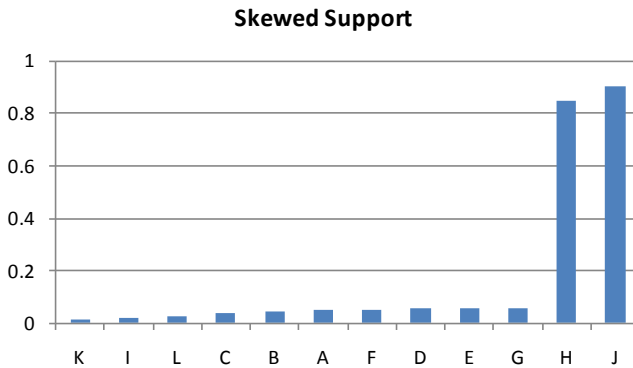
Think of the percentages as conditional probabilities:  $P(\text{Hired} \mid \text{Man})$  and  $P(\text{Hired} \mid \text{Woman})$ . These probabilities or confidence values reverse from the detail to the aggregate grouping.

How can the analyst spot potential conflicts caused by Simpson's paradox? Clearly, if the data rules are tested at both the detail and aggregate levels, it is straightforward to compare the two and see if reversals exist. If they do arise, the key is to recognize that they do not represent an error in computations but arise from Simpson's paradox. Then a decision has to be made whether the detail or aggregate results more accurately answer the questions being asked. A bigger problem arises when data is only analyzed at the aggregate level. Conclusions based on aggregate results might lead to incorrect decisions if there is a hidden factor affecting the results that could lead to Simpson's paradox. Hence, it is important to test association rules at detail/subgroup levels if these groups exist. Any time information exists, it should be incorporated into the analysis. The results will either confirm the conclusions from the aggregate rules or Simpson's paradox will arise, leading the analyst to consider more complex interaction effects.

### Skewed Support Data

Skewed support is likely to be relatively common in business—particularly in sales. Skew is a statistical term that refers to the asymmetry of the distribution (biased towards one end). **Skewed support** appears when the bulk of the items have few sales and a handful of them are sold in every basket. Think about a huge discount store—it is not possible for all 50,000 plus items to sell equally. Among other reasons, expensive items are not likely to sell as quickly as cheaper items. Think about a corner gas station/convenience store. Gas is probably sold in almost every transaction. A few items within the store probably sell reasonably well: perhaps soft drinks, beer (if available), or milk. The rest of the items sell occasionally, but as a percentage of the total number of transactions the support is low. Figure 6.12 illustrates the skew. Compare the approximate distribution to a normal “bell-shaped” distribution.

The effects of skewed support are interesting. The first effect is straightforward. It is going to be difficult to choose a value for minimum support. In the example, any number between about 0.1 and 0.8 will return only the few items that appear



**Figure 6.12**

Skewed support. A few items are purchased by almost every customer. Other items are bought occasionally, but the total support is low. Think about what items might be sold at a local gas station convenience store. Gas and what else?

in most transactions. Remember that once an item is dropped for falling below the minimum support, it is not considered in any other itemsets. The analysis includes only the few items that appear regularly. But, there might be interesting and useful relationships among the other items. Despite the fact that the support level is low compared to the high-volume items, they might still be important to the business. The only solution to this problem is to drop the support level to almost zero to bring in the other values. Of course, that will bring in almost all of the items—which could adversely affect performance if thousands of items exist.

A more challenging problem arises because of the mathematics. **Cross-support** consists of an itemset that contains one item from the high-volume group and one (or more) from the low-support group. Figure 6.13 shows the problem, by displaying values for a typical relationship and comparing them to values generated from cross-support. When support for A and B are roughly equal, the computation of  $P(B|A)$  represents confidence in a prediction and explains a relationship and perhaps causality between the two items. In cross-support, where B occurs in almost every transaction, this relationship no longer holds. Because B arises almost all the time, the few times that A shows up, it is likely to already have B in the basket. If 95 percent of the customers buy gas, then if anyone buys milk it appears that the two items were purchased together. But there is no correlation and certainly no causality. Simply by random chance, if almost everyone buys item B (gas), anything else that is purchased will appear alongside the gas item. This **spurious correlation** arose because of the extreme popularity of the B item. In fact, the results would actually be more interesting if the conditional probability were low. Since random chance indicates B should almost always appear with A, when they do not appear together, it would indicate some underlying effect keeping them apart is more important.

The cross-support effect of skewed distributions is difficult to correct. A few tools have options to check for and filter out these effects. In other cases, the analyst must simply be aware of the problem—keep an eye on support values for each item. When a rule appears with large disparities in support for the items, discount

	A	B	A & B	P(B A)	P(A B)
Typical	0.4	0.5	0.3	0.75	0.6
Cross-support	0.1	0.9	0.09	0.9	0.1

**Figure 6.13**

Cross-support with skewed data. When support for A and B are roughly the same,  $P(B|A)$  represents confidence in the prediction or potential causality. If  $P(B)$  is high, it will appear in almost any transaction. Consequently, by random chance, A and B can show up together almost every time A appears simply because B will always be there. The value for  $P(B|A)$  is spurious correlation and cannot be interpreted as causation.

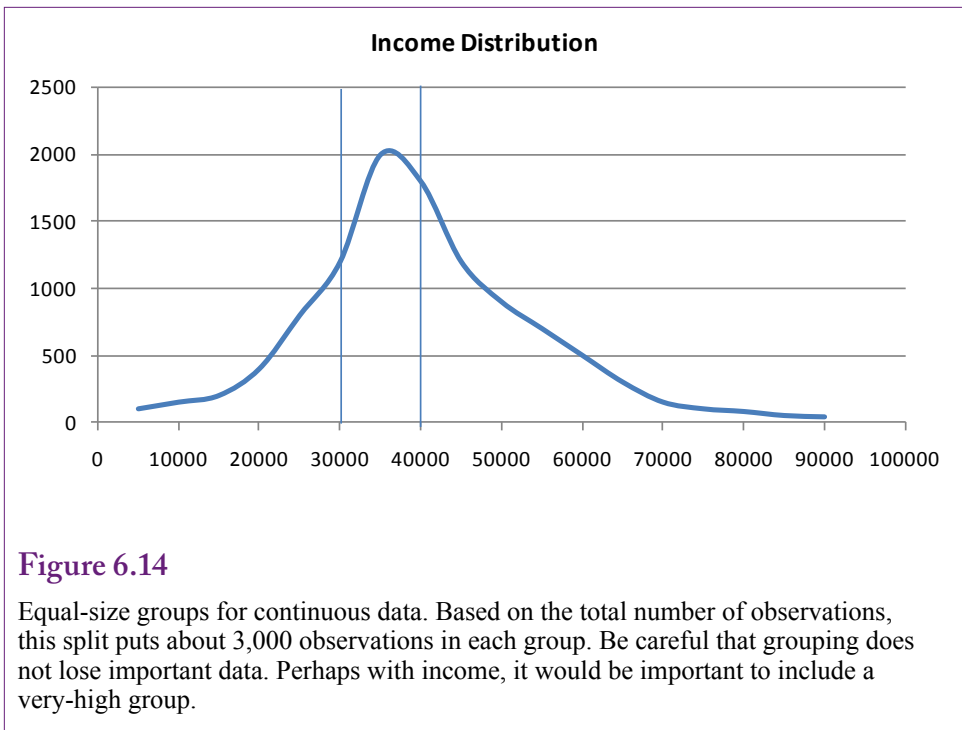
the value of the rule. Also, take a look at the lift. Because lift is confidence/ $P(B)$ , the value of lift will be close to 1.0 if A and B occur randomly in the cross-support example. In most cases of cross support, lift cannot get much higher than 1.0.

### Continuous Data

Association rule and market basket analysis can be applied only to discrete data. It functions by counting the number of transactions that contain each itemset. With continuous data (such as income), each value would represent a separate item. This approach could potentially create millions of dimensions, making the problem virtually unsolvable. Additionally, would a rule using  $\text{income}=50,101$  really be different from a rule using  $\text{income}=50,103$ ? Even if the tool found differences, the business interpretation would be meaningless.

If it truly is important to use continuous data in association analysis, the data series must be converted to discrete groups. For instance, income could be split into Low (Income < 30,000), medium (30,000-80,000), and high (Income>80,000). Each group would be treated as a separate dimension, which raises the critical question of the number of categories to create. Create too many categories and the dimensionality gets out of control. Plus, the results would be difficult to interpret. Create too few, or create them at the wrong split points, and the analysis will miss useful effects and rules.

If some existing business definition exists, it can be used to define split points. For example, the size of a firm is often defined by the number of employees, and definitions exist for small businesses (less than 500 employees), with the high-end defined by Fortune 500 level companies with tens of thousands of employees. Similar examples might be used to define splits for income levels, education, city population, and so on. In other cases, many choices exist for discretization. One approach is to determine the number of categories and find the split points that assign the observations equally to each group. Figure 6.14 shows an example of splitting observations on income into equal-sized groups. Based on the hypothetical data, each group contains about 3,000 observations. The split points are probably reasonable, but think about potential loss of data from the grouping. With income data, it might be important to include a very-high income group. Although few people would fall into the group, it could be important to the company and might generate useful rules for a group that could substantially improve profitability.



Clustering is another common approach to defining groups. If the data fall into natural groupings that can be identified by clustering algorithms, these clusters will be more likely to yield useful results. Once the clusters are identified and interpreted, the results will also be easier to understand and explain to managers.

Yet another approach might be to use standard deviation. Any observations that are one or two standard deviations below (or above) the mean might be considered as extremes. This partitioning could be useful to compare the “average” group against the lower and higher extremes. Ultimately, the method of discretizing the data depends on the problem and the types of questions to be addressed by the analyst.

## Quantity

True market basket analysis examines sales of items or categories of items. For example, it compares cupcakes to bananas. In most cases, each item is treated equally in a binary fashion—the item is either present or absent from the basket. The analytical tools do not look at the quantity of items purchased. So, someone who buys 20 bananas is considered the same as a person who buys one. For grocery items, this distinction is probably reasonable. It is likely that most people fall within similar limits in terms of quantities. However, other types of businesses might care greatly whether a person buys one item or 20 at a time. Although the tools have no options to handle quantity, it could be added as a new dimension. But, remember that quantity is a continuous variable so it needs to be grouped into categories.

For each product that might have different interpretations due to quantity purchases, define new categories and use SQL or the data mining tool to assign the new values. For instance, separate categories could be defined for 1 jacket, 2-5

## Sale

SaleID	SaleDate	CustomerID	EmployeeID
1101	7/8/2010	393	115
1102	7/9/2010	559	089
1103	7/9/2010	198	115

## SaleItem

SaleID	ItemID	Quantity	SalePrice
1101	22902	1	\$3.95
1101	11983	5	\$10.00
1102	22902	2	\$3.75
1103	83932	3	\$2.00

## Item

ItemID	Description	Category	ListPrice
11983	Blackberry	Pie	\$10.95
22902	Raisin	Bread	3.95
83932	Blueberry	Muffin	2.50

**Figure 6.15**

Typical database tables for Sale, SaleItem, and Item. Market basket analysis needs the transaction items from the SaleItem table, but also generally needs the category from the Item table.

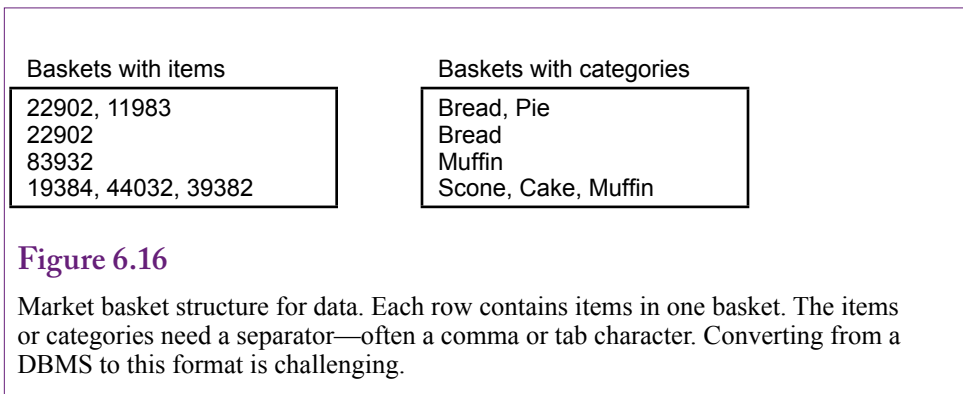
jackets, and 6 or more jackets. Each category becomes a new dimension and the analysis runs normally. Again, the challenge lies in identifying the products for which quantity might be interesting, and then identifying useful split points for the categories. All of these operations require more passes through the data. But, once the categories are defined, the association analysis and interpretation is straightforward.

## Data

### How does data need to be organized for association analysis?

Unfortunately, two different methods are used to organize data for association rule analysis. One method is easy; the other can be difficult when starting with traditional database transactions. When data begins in a DBMS, the transactions are stored in a normalized format. This data lists each item purchased on a separate row. Some tools, particularly those within a DBMS such as the SQL Server association method, use the data in this form. However, some tools that started with the market basket approach require the data to be organized by basket or transaction—where one row represents a transaction and all items are listed on that row. Note that missing data is not supported by the association rules techniques. But, there should never be missing data—either a sale lists the item or category being sold or it does not. If the item is not listed, it must not have been important enough to keep.





## Database Structure

Databases are carefully designed to efficiently handle transaction data. A specific set of rules defines the best way to organize data. Following these rules, the common business data for a Sale is broken into two tables: Sale and SaleItem. The Item data is stored in a third table, and all of them are connected through the primary keys. The SaleItem table has two columns in the primary key: SaleID and ItemID. Figure 6.15 shows sample data with a couple of rows of data. The SaleItem table is the key table for market basket analysis. Each SaleID is the transaction or basket. Each item in the basket is listed in a separate row. However, to perform the analysis by category instead of by each item, a query is built that links the Item table to the SaleItem table, making Category available to the analysis.

This format is the standard method of collecting transaction data in any DBMS or enterprise system. Data mining tools that are integrated with a DBMS generally use this format directly, or have tools to automatically convert data from this format. Sometimes queries are used to limit the data to specific stores, departments, regions and so on.

## Market Basket Structure

Early market basket tools were not designed to pull data from a DBMS. Many of them assumed data would be stored in simple text files—with all items in one basket listed on a single row in the text file. Figure 6.16 shows small examples for listings by item and category. Because one row contains multiple items, the items must be separated by a standard character—usually a comma or tab. This character cannot appear within the text of the items or categories.

Converting from the DBMS transaction format into this market basket format is a challenge. Standard database queries are not designed to repeat items across a single row of output. Custom programming code is needed to read items one row at a time and display them onto a single row of output, adding the separate character along the way. Figure 6.17 provides sample code that runs within SQL Server to create the basket listing for the Bakery case. Because the code uses an nvarchar variable to construct each line, the line is limited to a maximum of 4,000 Unicode characters. With a limited number of items sold on each sale and relatively short category names this limit is not a problem for the bakery case. Perhaps a bigger problem is the number of output rows. With almost two million rows, it is going to be a challenge to save the output. It would be better to use the SQL Server pro-

```

CREATE PROCEDURE BuildBasket
AS
BEGIN
    DECLARE db_cursor CURSOR FOR
    SELECT SaleID, Category
    FROM SaleItem
    INNER JOIN Product ON SaleItem.ProductID = Product.ProductID
    INNER JOIN ProductCategory ON Product.CategoryID=ProductCategory.CategoryID
    ORDER BY SaleID, Seq;
    DECLARE @SaleID integer, @Category nvarchar(250), @PriorSaleID integer;
    DECLARE @iCount integer, @strLine nvarchar(4000);
    OPEN db_cursor;
    FETCH NEXT FROM db_cursor INTO @SaleID, @Category;
    SELECT @iCount = 0, @strLine = "";
    WHILE @@FETCH_STATUS = 0
    BEGIN
        IF (@iCount > 0) SELECT @strLine = @strLine + ',';
        SELECT @strLine=@strLine + @Category;

        SELECT @PriorSaleID = @SaleID;
        select @iCount = @iCount + 1;
        FETCH NEXT FROM db_cursor INTO @SaleID, @Category;
        IF (@@FETCH_STATUS = 0)
        BEGIN
            IF (@PriorSaleID <> @SaleID)
            BEGIN
                PRINT @strLine
                SELECT @iCount = 0, @strLine = "";
            END
        END
    END
    CLOSE db_cursor;
    DEALLOCATE db_cursor;
END

```

**Figure 6.17**

Sample code to create market basket layout for the Bakery. The output line is limited to 4,000 Unicode characters which is sufficient for the Bakery case with a limited number of items sold at each sale.

cess to create a text file and write the lines to that text file. However, the code to handle files is too long to list here. It can be found on the Web. In most cases, big problems would be easier to handle with an external, compiled language such as C# or Visual Basic. The logic is similar, but file handling is simpler and it would be easier to create forms to let users pick the tables and columns interactively.

## Traditional Tools for Association Rules

**How difficult is it to create and interpret association rules?** The best way to understand market basket analysis and association rules is to work with several examples. Experiment with different settings of the support and confidence parameters and see how the results change. Although every problem is different, it takes some practice to understand how to use the parameters to control the results. The sensitivity of each data set is different, but the concepts are the

same. One catch is that many variations of the tools exist and each is somewhat different. One of the earliest tools is available on the Internet. Christian Borgelt provides several tools on his Web site (<http://www.borgelt.net/software.html>). His apriori tool searches for association rules and was the foundation for the SPSS Clementine routine. He even provides the source code, so programmers can modify or improve the routines.

## Goals

The a priori tool is a relatively straightforward implementation of the a priori algorithm. It takes input from a simple text file and writes rules to an output text file. It uses a measure of support as the initial filtering mechanism. The standard confidence measure is used to determine which rules to display. The one catch with this version of the tool is that it uses a version of support that is different from the traditional algorithm. In the common approach, support is computed as the percentage of transactions containing all of items in the rule: the antecedent and the consequent. The default in Borgelt's system is to compute the support only for the antecedent or left side of the rule. The routine does have a flag (-o) to tell it to use the original definition, but it does not work in at least some versions. It is not a critical problem, but it requires caution in setting the minimum support level and when interpreting the results.

## Data

Many of the early association algorithms require data to be stored as baskets—where each row in the text file contains a list of items in that market basket. Figure 6.16 provides an example. Each item must be separated by some character. Spaces and tabs are the default separators in the apriori code. These values can be changed with the -b flag. Use caution if spaces are used as separators—particularly when using text entries such as categories. If the text contains a space character, it needs to be converted to something else, typically an underscore ( \_ ). In the bakery case, the ProductCategory table should be edited and Sweet Rolls changed to Sweet\_Rolls.

As described in the Model section, when transaction data is stored in a DBMS, it is relatively difficult to convert it to the one line equals one basket data file. Usually, the easiest method is to write code that can read all of items for one sale and print the results to a file as a single row. The file can be large. The text file containing all of the Bakery data exceeds 50 megabytes.

## Results

Remember that it is best to start with low values for the minimum confidence, then decrease the support level until enough rules appear to provide results. In the end, a support level of 10 percent and confidence of 20 percent yields 115 rules. The list undoubtedly contains some rules that are weak, but it is short enough for an analyst to read through. The list can be imported into an Excel file and sorted. Alternatively, it could be imported into a new database table and SQL could be used to search the results.

Figure 6.18 shows the top results from the apriori tool sorted by confidence. Because the support definition includes only the antecedent, it provides minimal information. Also, note the inclusion of the empty set (null) as an antecedent. Because the empty set exists in every case, its support is always 100. Although its inclusion might appear strange at first, it is an interesting way to determine

which item categories are most popular on their own. In the example, Muffin has the highest confidence (40.9), followed by Pie (36.2), Bread (33.4), and Cupcake (32.8). These are the items that initially draw customers to the bakery.

The top paired rules also provide useful business information. Check the top three rules that show customers who buy Bread tend to buy Muffins 47.6 percent of the time, Cake implies Pie 42.9 percent of the time, and buying Rolls leads to Muffins 41.1 percent of the time. Managers could use these rules to arrange the store to entice additional sales, or perhaps change the pricing and profit margins. For instance, Bread was a relatively high standalone item and Bread leads to Muffin purchases. Perhaps a discount on Bread prices would attract more customers who would then purchase Muffins with a higher profit margin. In the end, it is up to managers to understand the rules and determine how to apply them to increase sales and profits.

Working down the list of rules, you eventually encounter a few rules with multiple items in the antecedent. The portion of the results shown in the table display an interesting relationship among Cupcakes, Muffins, and Pies. A third rule that is not shown also states that a basket containing Pie, and Muffins leads to the purchase of Cupcakes 28.3 percent of the time. These three rules imply that many customers come in to purchase these three items together. If business slows down for any reason, these three would be a good target for an advertised package.

These rules are only the beginning of the analysis, but they illustrate the basic process. Many other analyses should be run, such as comparing sales by day of week or perhaps certain seasons. These are left as exercises for the reader.

## Microsoft Association Rules

**How does the Microsoft Association Rules tool differ from traditional tools?** One of the main differences with Microsoft's Association tool is that it uses confidence instead of support as the primary filter. In terms of processing and eventual results, the difference is not huge, but it means the analyst has to think more carefully about how to set the minimum value. Microsoft also uses the name probability instead of confidence, which is reasonable since confidence is defined as  $P(B|A)$ . Microsoft then defines Importance as the measure to evaluate the returned rules. Microsoft's definition of Importance is hard to find but it is similar to lift with a few twists. The relative risk measures forms the foundation of Importance:  $P(B|A) / P(B|\sim A)$ . That is, the ratio of the probability of B given that A exists to the probability of B given that A does not exist. The formula translates to the probability of selecting B when A is present versus selecting B when A is not present. For mostly technical reasons, Microsoft adds a couple of tweaks. First, to avoid some specific situations with zero entries, the tool adds 1 to the count of each category. This tweak is usually invisible but required if trying to hand calculate the value from a small dataset. The other change is to take the log of the ratio—to convert the ratio into a number that can be plotted:

$$\text{Importance} = \log_{10}(P(B|A) / P(B|\sim A))$$

Values greater than zero correspond to ratios greater than one—which are positive effects on lift. Log values less than zero would indicate lift ratios less than one, or negative impacts on sales.

Muffin	<-	Bread	33.4	47.6
Pie	<-	Cake	17.5	42.9
Muffin	<-	Rolls	21.6	41.1
Muffin	<-		100.0	40.9
Bread	<-	Rolls	21.6	40.2
Bread	<-	Muffin	40.9	38.9
Muffin	<-	Pie	36.2	38.3
Muffin	<-	Cookie	28.3	38.0
Pie	<-	Cupcake	32.8	37.1
Muffin	<-	Cupcake	32.8	37.0
Pie	<-		100	36.2
Muffin	<-	Pastry	25.6	35.4
Scone	<-	Sweet_Rolls	26.3	34.9
Pastry	<-	Crepes	25.4	34.9
Crepes	<-	Pastry	25.6	34.7
Muffin	<-	Candy	21.3	34.7
Bread	<-	Candy	21.3	34.5
Pie	<-	Muffin	40.9	33.9
Cupcake	<-	Pie	36.2	33.6
Bread	<-		100.0	33.4
Cupcake	<-		100.0	32.8
Cookie	<-	Pastry	25.6	32.8
Pie	<-	Cupcake	12.2	32.3
Muffin	<-	Cupcake	12.2	32.2

**Figure 6.18**

Top results from apriori routine sorted by confidence. The Support definition uses only the antecedent.

## Goals

The basic objective of the Association tool is to find the rules or items that appear together most frequently. The tool uses the a priori algorithm, but does the initial cutoff based on minimum confidence instead of minimum support. The resulting rules are displayed with the confidence (Probability) estimate and the Importance estimate. By using a log transformation, the Importance values are charted to make it easier to see the rules with the highest lift.

## Data

Because the data mining tools are integrated with the SQL Server database, the association tool can read market baskets directly from the database transaction tables. It is important to specify the key columns. Typical columns consist of items such as SaleID for the transaction identifier and ProductID or Category depending on whether the analysis uses individual items or categories.

```

SELECT      dbo.SaleItem.SaleID, dbo.SaleItem.ProductID,
            dbo.SaleItem.Quantity, dbo.SaleItem.SalePrice,
            dbo.Product.CategoryID, dbo.ProductCategory.Category,
            dbo.Product.ProductName
FROM        dbo.SaleItem
INNER JOIN  dbo.Product
           ON dbo.SaleItem.ProductID = dbo.Product.ProductID
INNER JOIN  dbo.ProductCategory
           ON dbo.Product.CategoryID = dbo.ProductCategory.CategoryID

```

**Figure 6.19**

Named query to combine SaleItem, Product, and ProductCategory tables. Name it SaleItemProductCategory.

The Bakery case was designed to illustrate the uses of the market basket tool. Create a new data source that connects to the Bakery database. Because the database has few tables, create a new dataset view that contains all four tables: Sale, SaleItem, Product, and ProductCategory.

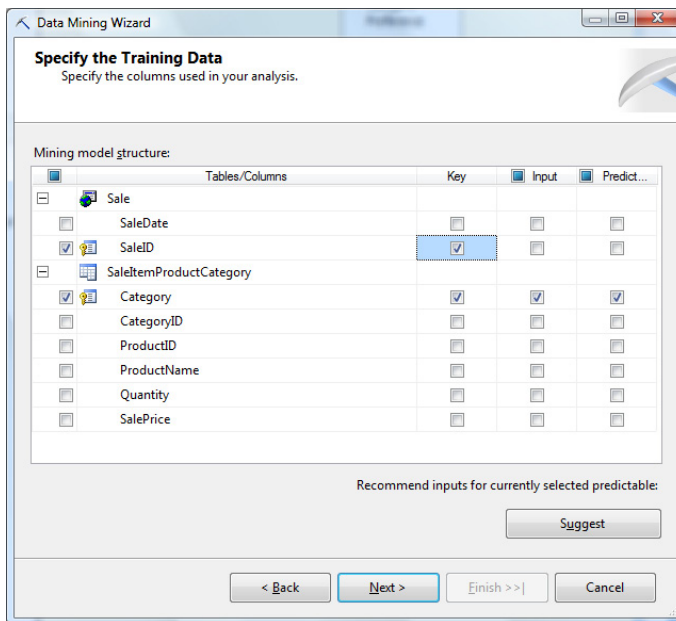
The analysis is easier to configure if all of the main columns are in one named query instead of scattered across the SaleItem, Product, and ProductCategory tables. Create a named query (SaleItemProductCategory) that links SaleItem with the Product and ProductCategory tables. It simply needs to display the SaleID, ProductID, and Category columns, but the others can be included as well. Figure 6.19 shows the simple query. In some cases, the combined data might already exist in a data cube.

Once the named query is created, assign logical primary keys to it in the data source view: SaleID + ProductID. Then create a relationship from the query to the Sale table based on the SaleID. Remember if the analysis is performed on a different server than your development computer (localhost), use the Deployment properties of the project to set the name of the server.

Microsoft's Association Rules model has an additional feature that is useful in some problems. So far, all of the examples in this chapter have pulled dimension attributes from a single column, such as Category or ProductID. Microsoft's configuration wizard makes it easy to use multiple columns. For instance, the Dining case has one table that includes day of week, gender of the customers, and the meal time (breakfast, lunch, dinner). All of these attributes can be combined into itemsets. With the Microsoft wizard, simply specify each attribute as both an input and predictable column. For traditional tools, it would be necessary to create a query that defines a new column to combine the attributes. For example, Meal = DOW + '\_' + MealTime + '\_' + Gender. On the flip side, the transaction key needs to consist of a single column and the tool is easiest to configure if all of the data reside in a single table or query.

## Results

The tool is straightforward to run. Create a new mining structure using the Microsoft Association Rules. To ensure the SaleID key is unique, select the Sale table as the Case table and the new named query (SaleItemProductCategory) as a Nested table. Figure 6.20 shows the selection of the columns for the analysis. The SaleID must be set as the Key value because it defines the transaction or market basket.



**Figure 6.20**

Selecting columns for association rules mining. Set the SaleID as the key to define the transaction or market basket. Select the Category column as the prediction data and the input. Check drill through on next screen.

Every item purchased with the same SaleID is purchased at the same time. The Category is specified as the predictable and input column. The Category column contains names so it is more useful than the CategoryID column which consists of simple numbers. This choice performs the analysis with categories as the dimensions. Selecting ProductID instead of Category would perform the analysis on all of the individual items. Avoid starting with ProductID because it is a much bigger problem and could take a long time to run. Finish the wizard to complete the model definition. Right-click the model in the Solution explorer and choose the option to Process it and run the processor. When processing is complete, right-click the model and choose the option to Browse the results.

Figure 6.21 shows the initial results from the Association Rules model. Clicking on the heading causes the rules to be resorted. Selecting minimum probability and importance in the filter boxes reduces the number of rules displayed. These tools provide a filter for searching for rules—particularly useful when hundreds or thousands of rules appear. The charts for importance make it easier to see which rules have the greatest lift. Note that negative values will appear in red—they represent lift values below 1—or negative correlations.

Notice that the cutoff value for probability can only be increased on this form. Changing it here only changes the display results—it does not rerun the analysis. Decreasing the numbers requires changing the parameters and re-computing the results. Figure 6.22 shows the basic process for changing the underlying parameters. On the Mining Models tab, right-click the Microsoft\_Association\_Rules model and choose the option to Set Algorithm Parameters. Remember to be cautious when making changes. The minimum probability (confidence) is used to



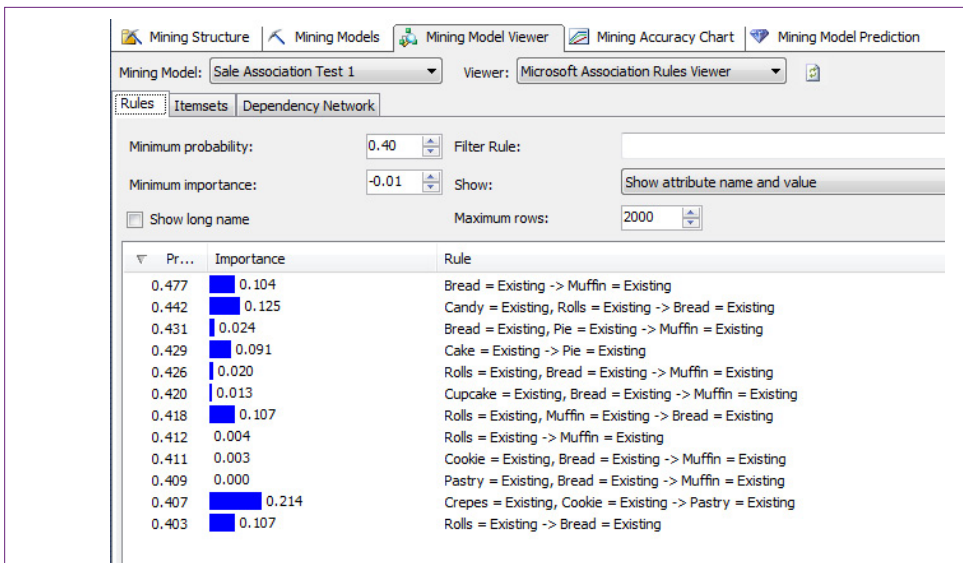


Figure 6.21

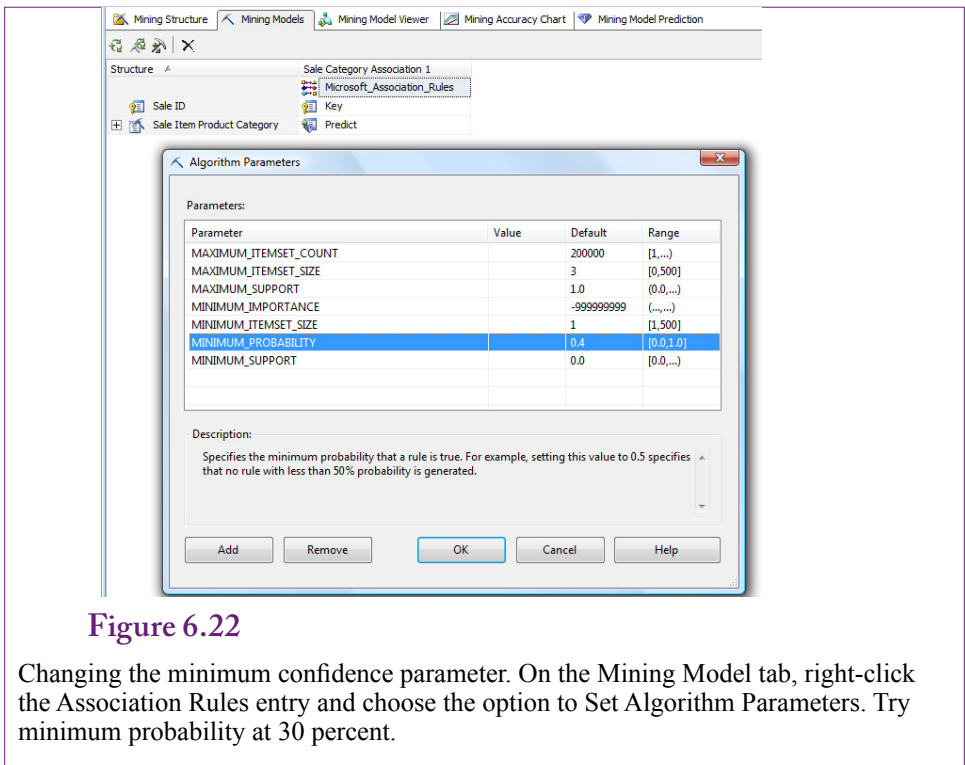
Initial results from Microsoft Association Rules. Note the minimum confidence can be increased to reduce the number of rules displayed. To decrease it, the model needs to be re-processed with new parameters.

limit the number of itemsets that are evaluated. Setting a low value causes the result set and processing time to increase exponentially. For the full collection of data in the Bakery problem, the default values appear to produce a reasonable number of results. Judging from the confidence values in the traditional analysis, some rules might be missed, so it could be useful to rerun the analysis at 30 percent.

As shown in Figure 6.23, the Microsoft tool also generates a dependency diagram. The arrows show the dependency of the nodes. For instance, purchase of a Cake leads to a purchase of Pie. The slider on the left is used to cut off weaker links. Clicking a single node causes the others to be color coded to show the direction of dependency. For instance, select the Muffin node to see that purchases of both Rolls and Bread have an effect on the sale of Muffins.

## Comparing Results

At first glance, the results from the traditional and Microsoft tools might appear to be different. To see similarities, sort the results from both tools by confidence (probability). Both tools list Bread -> Muffin at the highest confidence of 47.6 percent, Cake->Pie at about 43 percent, Rolls -> Muffin around 41 percent, and Rolls -> Bread around 40 percent. The probability values might differ slightly from the traditional results because the Microsoft tool holds out 30 percent of the observations to be used for testing. If this holdout set is decreased to zero, the numbers should match those from other tools. Also, if the Microsoft results are reprocessed with a lower minimum confidence, the results should continue to match those from the traditional tool. However, note that the Microsoft tool has found additional results with relatively high confidence that were not found by the



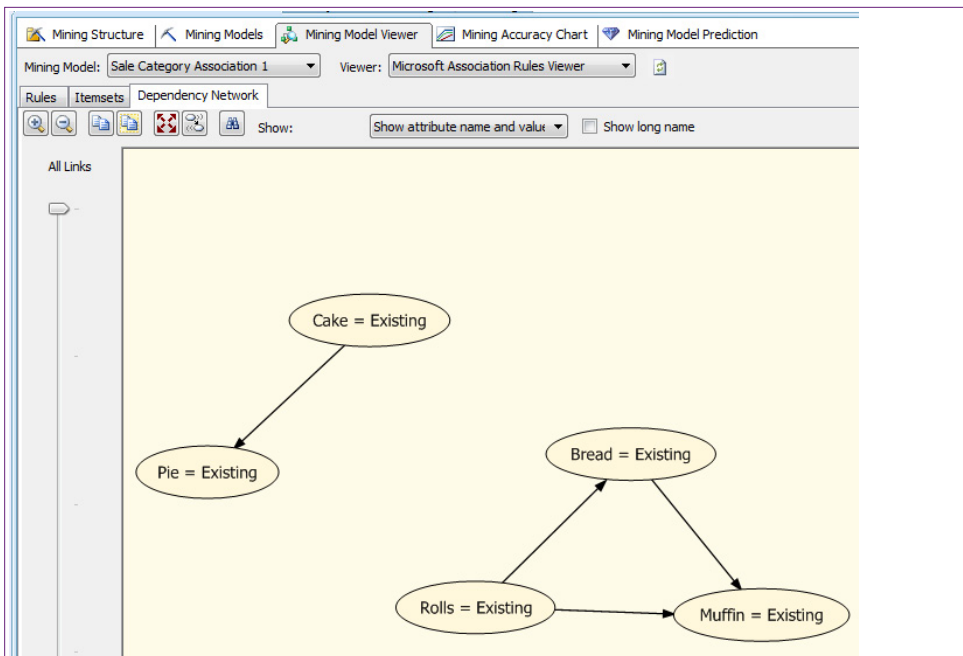
**Figure 6.22**

Changing the minimum confidence parameter. On the Mining Model tab, right-click the Association Rules entry and choose the option to Set Algorithm Parameters. Try minimum probability at 30 percent.

traditional tool. In particular, Crepes, Cookie -> Pastry (0.407) and Candy, Rolls -> Bread (0.442) have the two highest Importance measures. Neither rule appears in the list of items from the traditional tool.

How can the Microsoft tool include (useful) results that are not in the results from the traditional tool? Because the traditional tool uses support as a filter and Microsoft uses confidence as a filter, the most likely cause is that the support for the items is too low to meet the cutoff value in the traditional tool. An SQL query could be used to count the number of cases, but Microsoft output provides an Itemset search tool. To check the rule for Crepes, Cookie, Pastry, set the minimum itemset size to 3 and filter on the word Crepes. The resulting list shows that the three items appear together in 23,728 market baskets, out of about 2 million total transactions. Because the traditional tool used in the example defines support only on the first two items, change the itemset size to 2 and find the entry for Crepes and Cookie. These two categories appear together in 58,250 baskets. Divide by 1,925,819 transactions to get a support percent of slightly over 3 percent. The cutoff used in the traditional tool was 10 percent—and that created 115 potential rules. Dropping to the 3 percent level would bring in the missing rule, but would also bring in hundreds of additional rules to sort through.

The point of the comparison example is to emphasize that any association tool can miss rules that might be useful. It is the reason there are so many different measures of interestingness and importance. It is a good idea to look at large datasets from several different perspectives. Bring in extra rules, compute multiple measures of interestingness and sort the results. Compare them and think about the potential interpretations. Remember that the purpose of data mining is exploration.



**Figure 6.23**

Dependency Network. The arrows show dependency. For example, purchasing cakes leads to purchases of pies. Select one of the nodes to see a color-coded reference of how the nodes are linked.

## Summary

Association rules or market basket analysis is one of the methods that started the data mining industry. The underlying concept is straightforward: Find the items or events that occur together. In market basket settings, the goal is to find items that are commonly purchased together. The tool can be used for many tasks in business, including store layout, pricing, and cross-selling. The tool is the foundation of many recommendation engines, such as those used by Amazon and Netflix.

In practice, the model is challenging to implement because of the number of comparisons required. If  $d$  is the number of items or dimensions, the potential number of comparisons assuming a single output item is 2 raised to the  $d$ -power. The a priori algorithm significantly simplifies the search and makes the method feasible even for relatively large datasets. Still, problems with a large number of items often need to be reduced by using item categories instead of detailed products. In many cases, these aggregate comparisons are more useful because the interpretation is more rational. For instance, if you are looking for a comparison between Cakes and Pies, is it really useful to search for differences between types of cakes and pies?

The a priori algorithm gains performance by cutting off potential rules early. This pruning is based on a parameter—typically the support or frequency of the initial items in the basket. Once an item or category is dropped from the analy-

sis, all combinations of that item are also dropped. Choosing the threshold value for this parameter is a critical step in running market basket analysis. Setting the value too high blocks any potentially interesting rules from entering the analysis. Setting the value too low brings in too many itemsets to be solved in a reasonable time.

Categories are often a useful compromise when the number of products is too large. Yet, Simpson's paradox can result in unexpected shifts in the results. Aggregate results do not necessarily match the results from the detailed subgroups. Consequently, it is important to perform analyses at multiple levels and to evaluate the results of subgroups as well as the total. Different tools use different measures of interestingness, so results sometimes vary across tools. The problem is that no perfect measure of interestingness exists, so any tool can miss rules that might be useful. In the end, analysts often need to perform many different searches for rules, sorting results by various measures and evaluating rules for potential value. Market basket analysis is an exploratory tool that provides new perspectives on combinations of items.

## Key Words

---

a priori algorithm	market basket
antecedent	minimum confidence
association rules	minimum support
confidence	recommendation engine
consequent	relative risk
cross-support	rules
curse of dimensionality	Simpson's paradox
data associations	skewed support
interestingness	spurious correlation
itemsets	support
lift	unsupervised learning

## Review Questions

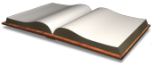
---

1. What business tasks are suited to association analysis?
2. What are the statistical definitions of support, confidence, and lift?
3. Why are different measures needed (and others created) for evaluating association rules?
4. How does the a priori algorithm solve the problem of the curse of dimensions? What problems does its solution create?
5. What steps can be taken to reduce problems with dimensionality on huge datasets?

6. How can Simpson's paradox affect market basket analysis and what steps should be taken to reduce its impact?
7. What problems are created when a few items or categories appear in most of the transactions?
8. Why do the algorithms only use discrete data and how can continuous data be used in an association rules problem?
9. Why is transaction data from a DBMS often a problem for tools that estimate association rules?
10. How is the Microsoft Association Rules tool different from traditional tools?

## Exercises

---



### Book

1. Perform the traditional association rules analysis on the Bakery dataset by category. Comment on the results.
2. Perform the Microsoft Association Rules analysis on the Bakery dataset. Comment on the results.
3. Reduce the minimum confidence level in the Microsoft Association Rules and comment on the number of rules added to the result set and the processing time.
4. Run the Microsoft Association Rules analysis at the product level instead of category. Comment on the results and the processing time. If no rules appear on the initial run, reduce the minimum confidence level.



### Rolling Thunder Database

5. Run an association analysis on the Rolling Thunder component sales. Do the results match the component groups?
6. Create a query that lists ModelType purchased by customer—at any time. Run an association analysis to see if some model types are more commonly purchased together.
7. Create a query that lists ModelType purchased within a state for a specific month. For example, create `StateYearMonth = SaleState + '_' + Year(OrderDate)*100 + Month(OrderDate)`. Run an association analysis using `StateYearMonth` as the transaction key and `ModelType` as the dimensions. Comment on any results. How would these results be interpreted differently from those in the prior exercise?

8. Create a query that defines a new column to combine SaleState and ModelType. For example, create StateModel = SaleState + '\_' + ModelType. Also define a column that includes the year and month of the sale as the transaction key: YearMonth = Year(OrderDate)\*100 + Month(OrderDate). Run an association analysis and comment on the results. What is the goal of this specific model?



## Diner

9. Are there associations between type of meal, day of week, and gender? With Microsoft Association, you can simply add all three as predictable and input columns. With traditional tools, use DinerID as the key and define a new column to use for the analysis as: DOW + MealTime + Gender. Explain how at least one of the rules could be useful to the business. What other data might you want to include to make the results more meaningful?
10. Are there any useful associations between type of meal, gender, and the total bill? See the hint in the prior question, but also recognize that the BillTotal is continuous. Compute the average to discretize the bill into low and high.



## Corner Med

11. What diagnoses codes commonly occur together? Hint: Diagnoses codes are hierarchical. The first (left) character is the primary level. The first three are the main category.
12. What procedures commonly occur together? Hint: Procedure codes are hierarchical. The first (left) character is the primary level. The first three are the main category.
13. What procedures have commonly been done on patients at any time?
14. What drugs are commonly prescribed in the same visit?
15. What drugs are commonly taken by the same patients at any time?
16. Are there procedure associations among employees? That is, using employees as the transaction key, are there association rules? What do they mean?



## Basketball

17. Using Player and Game as the transaction ID, discretize the data and determine if there are associations between field goals attempted, three-points attempted, free throws attempted, and assists (in one season).
18. Using the team as the transaction key, are there associations among the colleges attended by the players on the teams.
19. Using GameID and TeamID as the transaction, are there any association rules between IsHomeTeam, IsPlayoff, and WonLoss?



## Bakery

20. Which products are commonly purchased together?
21. Split the data into three sets based on time of day (breakfast, lunch, and afternoon). Determine which products are purchased together in each of the three sets. Comment on the results.
22. Split the data into two sets based on day of week: Weekend (including Friday) versus the rest of the week. Determine which items are commonly purchased together in both of the data sets. Comment on the results.



## Cars

23. What associations exist between attributes in the cars database?



## Teamwork

24. Use queries to filter the Bakery data to days of the week, or at least certain groups such as work days versus the weekend. Assign one day to each team member and run the association analysis. Comment on any differences in rules among the team's results and the overall results.
25. Use queries to filter the Bakery data by seasons—particularly holidays and summer. Assign team members to each set of data and run the association analysis. Comment on any differences in rules among the team's results and the overall results.



26. Use queries to filter the Corner Med data by day of week. Assign each day to a team member and see if the associations among procedures vary by day.
27. Using the basketball database, use queries to assign one basketball team to each team member. Discretize the data and determine if there are associations between minutes played and field goal percentage. Compare the results for each team.

## Additional Reading

---

Agrawal, R., H. Mannila, R. Srikant, H. Toivonen and A.I. Verkamo, 1995, Fast Discovery of Association Rules, *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, Cambridge, MA. [The article that first described the a priori algorithm for fast association searching.]

Bickel P., E. Hammel, J. O'Connell, 1975, Sex Bias in Graduate Admissions: Data from Berkeley, *Science* (187), 398-404. [Classic example: Berkeley lawsuit on gender discrimination.]

Geng, Liqiang and Howard J. Hamilton, 2006, Interestingness Measures for Data Mining: A Survey, *ACM Computing Surveys*, 38(3),

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman, 2009, *The Elements of Statistical Learning/2e*, Springer: New York. [An outstanding book on data mining, with an emphasis on theory. A graduated-level book that requires a strong mathematics background.]

E.H. Simpson, 1951, The Interpretation of Interaction in Contingency Tables, *Journal of the Royal Statistical Society, Series B*, (13), 238-241. [Original description of Simpson's paradox. Many Web sites provide analysis and examples.]

Tan, Pang-Ning, Michael Steinbach, Vipin Kumar, 2005, *Introduction to Data Mining*, Addison Wesley: New York. [A computer-science approach to data mining with algorithms and computational analysis.]

G.H. Yule, 1903, Notes on the Theory of Association of Attributes in Statistics, *Biometrika*, (2), 121-134. [The earliest version of the Yule-Simpson effect or Simpson's paradox.]

# Evaluation of Dimensions

## Chapter Outline

Introduction, 308	
Business Situation, 309	
Model, 309	
Data, 311	
<i>Attributes and Observations, 312</i>	
<i>Continuous and Discrete Data, 312</i>	
<i>Missing Data, 313</i>	
Linear Regression, 313	
<i>Goals, 314</i>	
<i>Data, 316</i>	
<i>Tools, 318</i>	
<i>Results, 322</i>	
<i>Attribute Evaluation, 327</i>	
<i>Prediction, 328</i>	
Logistic Regression, 330	
<i>Goals, 330</i>	
<i>Data, 332</i>	
<i>Tools, 333</i>	
<i>Results, 334</i>	
<i>Attribute Evaluation, 338</i>	
<i>Prediction, 339</i>	
Naïve Bayes, 340	
<i>Goals, 341</i>	
<i>Data, 345</i>	
<i>Tools, 345</i>	
<i>Results, 346</i>	
	<i>Attribute Evaluation, 346</i>
	<i>Prediction, 348</i>
Decision Trees, 349	
<i>Goals, 350</i>	
<i>Data, 352</i>	
<i>Tools, 353</i>	
<i>Results, 353</i>	
<i>Attribute Evaluation, 355</i>	
<i>Prediction, 355</i>	
Neural Network, 358	
<i>Goals, 359</i>	
<i>Data, 361</i>	
<i>Tools, 361</i>	
<i>Results, 361</i>	
<i>Attribute Evaluation, 362</i>	
<i>Prediction, 363</i>	
Model Comparisons, 365	
<i>Prediction, 366</i>	
<i>Attribute Evaluation, 367</i>	
<i>Nonlinear Complications, 368</i>	
Summary, 369	
Key Words, 370	
Review Questions, 370	
Exercises, 371	
Additional Reading, 373	

## What You Will Learn in This Chapter

- How do some attributes or variables influence other attributes?
- When do you need to evaluate dimensions?
- What type of data can be used and how should it be structured?
- How does linear regression work and why would you use it?
- How can attributes for discrete Y-values be estimated?
- How do you begin an analysis when you know little about the data?
- Is there a way to organize the attributes to see how they explain the decision?
- How can the modeling process be automated to handle nonlinear relationships?
- How do the model results compare to each other?

### Union Pacific

Union Pacific is the nation's largest railroad, operating 900 trains a day. Although many people are no longer familiar with railroad technology, a key element in safety and performance is derailments, which can cost millions in damages and lost time. In the early 2000s, Union Pacific installed acoustic and visual sensors on the undersides of its rail cars. The sensors have focused on imminent problems with the track and wheels—notably the status of wheel bearings; and have reduced bearing-related derailments by 75 percent. The acoustic sensors identify an average of three cars a day and send the warning information to engineers who move the cars offline for repair at the earliest opportunity. Around 2010, the company installed visual sensors (cameras) as well—to help identify flattening of wheels and other problems. More recently, the company developed predictive software that analyzes the data from acoustic and visual sensors for patterns. Lynden Tennison, Union Pacific CIO, noted that the system can analyze 40 million patterns a day and alert engineers of a potential problem within five minutes. [Hickins 2012].

Predicting events is better than waiting for bad things to happen.

Michael Hickins, “Union Pacific Using Predictive Software to Reduce Train Derailments,” *The Wall Street Journal*, March 30, 2012. <http://blogs.wsj.com/cio/2012/03/30/union-pacific-using-predictive-software-to-reduce-train-derailments/>

## Introduction

---

### How do some attributes or variables influence other attributes?

Data mining ultimately comes down to this question. In fact, the question is the heart of any discipline from business to medicine to science and philosophy. Ideally, people can study data, build models, and develop theories that explain how changes in some attributes will cause changes in other variables. But, **causality** is a tricky subject. It is difficult to use statistics to prove causal changes. Most of the time, statistics can evaluate **correlation**—two things that move together. When the sun comes up in the morning and the temperature increases, does that mean the sun caused the temperature increase? Probably, and we have a physics model to explain it. When the sun comes up in the morning and traffic in the city increases, does that mean the sun causes traffic jams? Not exactly but there is a chain of events that can explain the relationship. If the sun comes up in the morning and the number of deaths per hour decreases at a local hospital, did the sun cause it? This last correlation presents a much harder problem to answer. The point is that if data items are correlated, the relationship raises questions that point analysts in a direction to look for explanations. Items that are not correlated are less interesting—so statistical data mining that identifies relationships and shows where they do not exist can be a powerful tool for exploration.

This chapter examines several data mining tools that can be used to evaluate how attributes affect other dimensions. Once relationships are identified, the results can be used for classification and prediction. Classification in the sense that values for the attributes can divide an outcome variable into groups. For instance, certain characteristics of age, income, and education might divide customers into heavy purchasers and weak purchasers. For instance, perhaps older, higher income, people with more education tend to be heavier purchasers of items. Prediction is used when managers and analysts want to generalize results and plug in values of attributes to see what might happen. For example, if the firm increases marketing to attract higher-income customers, how much might sales increase? Specifically, will sales and profits increase enough to compensate for the added marketing costs?

The primary tools for this type of study are: (1) linear regression, (2) logistic regression, (3) naïve Bayes, (4) decision trees, and (5) neural networks. With all of these tools, the analyst identifies an outcome or predictable Y-attribute that managers want to observe or forecast. The goal is to examine a set of X-attributes that might conceivably explain movements in the predictable variable. The differences in the tools are primarily in the techniques used to estimate correlations and create predictions. There is one main exception. Linear regression requires that the dependent variable consist of continuous data, not discrete or categorical observations. The other techniques are geared towards categorical outcomes, although in some cases they can handle continuous data as well.

This chapter looks at each of the five tools and identifies the goal of the tool with a brief look at the methodology. The methods are described in mathematical terms, but the coverage is light and designed to highlight the challenges and applicability of the tool. Deeper mathematical and programmatic discussions of the tools are found in many other books and Web sites. The goal of this chapter is help you understand the tool so you can use it effectively. Each section also describes the data needed. The last four tools are all demonstrated using the same data set from the Rolling Thunder Bicycle Company. The methods are demonstrated us-

ing the versions in Microsoft BI. In the case of linear and logistic regression, the methods are also demonstrated using more traditional tools. These two methods have a long history, and the Microsoft BI results are slightly different from the traditional approaches. It is good to know both methods. The basic results are presented from the application of the methods and they are examined for insight into attribute evaluation and prediction. At the end, the four major discrete-evaluation tools are compared to see what information can be learned for the sample problem.

## Business Situation

---

**When do you need to evaluate dimensions?** The essence of data mining is to focus on an outcome or fact variable. Common business examples include sales, profit, whether a loan will be repaid. As a manager, you want to know how other variables or dimensions affect the outcome. How sensitive are customers to price? If you change the color of a product or its packaging, how many more units can you expect to sell? Or, what happens if you change a production process—such as switching to organic ingredients—will sales increase? Or, perhaps the cost function is complex and no one completely understands what happens to costs as production increases.

Another classic example from the banking industry involves the question of which customers will be able to repay loans. What characteristics or dimensions are the most important: salary, job tenure, savings balances? Are there tradeoffs among the dimensions: If someone has a low salary, how large does the savings account need to be to compensate?

These types of questions involve **classification** and **prediction**. Classification in the sense that you want to know which customers fit into defined categories (successful loans, big spenders, and so on). Prediction is used when you encounter new data you want to be able to predict the eventual outcome. Essentially, you are estimating a **model** of the relationships between the dimensions and the outcome variable. In many cases, linear relationships are useful because they are easy to estimate and easy to understand. These are often estimated using linear regression. More complex systems require nonlinear relationships. These can be estimated with neural networks, but the results can be more difficult to interpret.

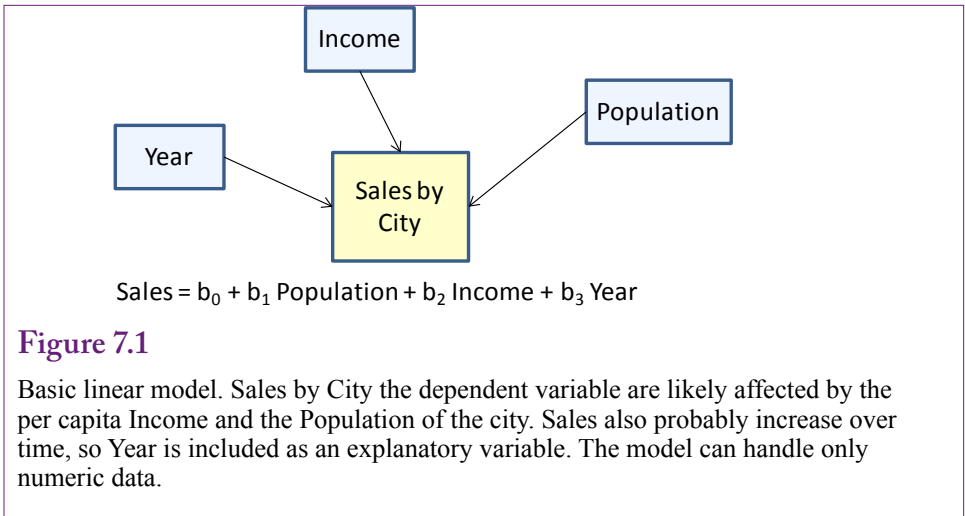
## Model

---

**How do you know which model to use? How do you know which attributes to choose?** The analyses in this chapter are guided by the analyst. In the basic case, you must select the dependent variable and the list of independent variables to be examined. You also select a tool to estimate the model, and the choice of the tool often imposes a structure on the model. For example, using linear regression restricts the model to simple linear relationships. It is possible to minimize the restrictions—particularly by choosing a neural network tool to estimate the relationships—but your role in determining the model is important.

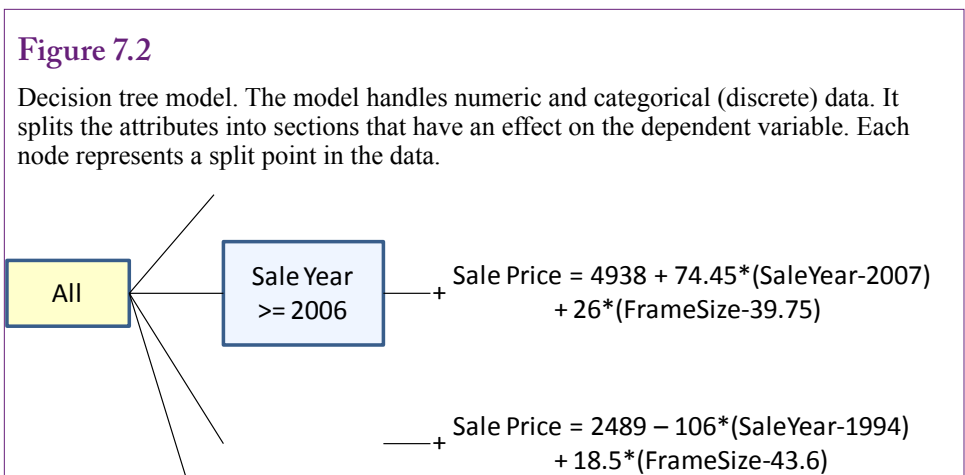
Figure 7.1 shows two ways to present a simple model: (1) graphically showing the explanatory attributes affecting the dependent variable (Sales), and (2) as a linear equation, where the goal is to estimate the coefficients (b-terms). The equation approach is compact and it is sometimes easier to understand. The equation is typically estimated with a **linear regression** tool.

Figure 7.2 shows a portion of a different type of model. A **decision tree** is sometimes used to create a model to analyze the data. A decision tree handles discrete (categorical) data as well as continuous data. Even the dependent variable



can be discrete data. Only a tiny portion of the model is shown in the figure. Decision trees often grow quite large. Each node represents a split point where values for an independent variable have a different effect on the outcome. In this small example, the Sale Price is affected by the Sale Year and the Frame Size. This portion of the tree shows that prices are affected differently for years before 1996 and after 2006. Other year segments are also different but not shown. Marked by the plus sign (+) the tree can be expanded to the right to see the effect of other attributes—notably Frame Size. Decision trees can be complex, but they provide a visual way to help you explore the impact of various attributes on the chosen dependent variable. They are also good for finding interactions among the attributes. In the example, the effect of Frame Size varies depending on the year.

One of the goals of data mining is to reduce the need of the analyst to specify models. This approach is both good and bad. It makes it easier to explore many possible effects without knowing too much about the underlying data. But, without specifying a model in advance, you impair the ability to make statistical tests. The results you obtain might not be statistically useful. Explorations can be useful when you are beginning your study of the data.



If you do need to specify a model ahead of time—and in most cases you at least need to come up with potentially important attributes—you need to look to experts who have studied the problem. In many cases, business disciplines including economics, accounting, finance, and marketing have developed models that can be used to examine problems. For instance, economics identifies several factors that affect the demand for products. So sales and price should be influenced by attributes such as customer income, prices of other products, and expectations about the future. Economics and accounting have developed models to estimate costs, including fixed costs and per-unit or marginal costs. Marketing models identify many attributes that affect consumer choices. Finance has developed complex models that evaluate financial products and risk. In other areas, you need to turn to different scientists, including biological, growth, and DNA models. If you can find a model that someone has already developed and validated, it will be much easier to identify the important attributes and test their importance with your data.

The main point to remember is that to evaluate and understand how various attributes affect an outcome variable, you need to know enough about the subject to choose the appropriate attributes. Once you know the types of data, you then need to choose the appropriate modeling or mining technique that matches the data. These concepts are covered in the next sections.

## Data

### What type of data can be used and how should it be structured?

The data needed for evaluating dimensions is similar to that used in the cube browser: A fact or outcome measure and a set of dimensions that you want to examine. It is helpful to think in terms of the traditional regression model, where the data consists of observations or rows of data. One observation is essentially a row of data consisting of values for each of the dimensions and the corresponding attribute you wish to predict. The prediction attribute is sometimes called the **dependent** or endogenous variable; the other factors are **independent** or exogenous variables.

**Figure 7.3**

Standard data layout. A column represents a single dimension attribute. Each row is one observation, such as a purchase or a customer. Typically, you will need a key column attribute that identifies each row.

CityID	Pop	Income	Year	Sales
4650	48950	34004	1994	2700
14438	2049	23020	1996	2430
4943	13064	31282	2001	2680
17664	2662	45101	2004	2420
342	16844	23153	2007	3910
4586	8491	39605	1997	1790
2625	7173	32016	1997	2630



## Attributes and Observations

For all of the tools in this chapter, the data is most often viewed as a table—where the columns represent the attributes and each row consists of one observation. Figure 7.3 shows sample data for a problem with independent attributes on each city (population, per-capita income, and the sale year). Total sales value to customers in that city for a given year is in the Sales column. Some tools allow the use of multiple dependent variables. For example, you might add sales volume (count the number of items) to this example. However, most standard data mining tools treat multiple predicted columns as separate problems. Statistically, you gain little by combining them into one problem. It might be slightly more convenient to list multiple dependent variables, but you need to be careful in the estimation process. You must also be careful when you explain the results and remember that even if the two variables are logically related, the process treats them separately. In general, it is safer to use a single dependent variable and run separate analyses. However, most advanced econometric problems have multiple dependent variables—and multiple equations. These simultaneous equation models require special estimation methods that are not handled by typical data mining tools.

Microsoft BI requires that every dataset must include a key column that uniquely identifies each row. This column does not need to be used within the analysis, but it ensures the rows are identifiable. An important catch is that only one column can be used—not composite keys consisting of multiple columns. If you do not have a single-column key, you can create a new column in the dataset that combines the values from the columns you do have. For example, if you want analyze sales by customer by year you start with two key columns: CustomerID and Year. In the data view, you need to create a new column:

```
Cast(CustomerID as nvarchar) + '-' + Cast(Year as nvarchar)
```

This expression creates a new column that contains both the CustomerID and the Year, separated by a hyphen. For example: 192-2009. The *Cast* function is needed because the data must be converted to text (nvarchar). Without the cast, SQL Server would simply add the two numbers numerically.

The structure of the data helps you focus on the goals of this type of analysis. You choose the dependent variable that you want to understand or predict. Then you select various attributes that you think might affect that variable. You collect observations of cases and they become the rows of the table. The data mining tools examine the rows and use them to estimate weights or values that have an impact on the dependent attribute.

## Continuous and Discrete Data

As you will see in the following sections, one of the key issues affecting your choice of tools is the type of data available. In particular, it is important to know if each attribute contains numeric continuous data, discrete numbers, or even categorical data. For example, regression tools require numeric data, but decision trees can handle categorical values. Figure 7.4 summarizes the data types that can be handled by the tools covered in this chapter. Notice that the regression tools are the most constrained. Decision trees are flexible, but it can be harder to interpret the results. Neural networks are also flexible, but some features are only available when you use continuous data. Also, the specific requirements can vary depending on the specific version of the tool. Some vendors are less flexible than others. The requirements listed here apply to the Microsoft BI tools.

Tool/Method	Dependent	Independent
Linear Regression	Numeric: continuous	Numeric: continuous Numeric: discrete
Logistic Regression	Numeric: discrete	Numeric: continuous Numeric: discrete
Decision Tree	Numeric: continuous Numeric: discrete Categorical	Numeric: continuous Numeric: discrete Categorical
Naïve Bayes		
Neural Network	Numeric: continuous* Numeric: discrete Categorical	Numeric: continuous* Numeric: discrete Categorical

**Figure 7.4**

Data requirements for tools. Some tools require numeric data. The dependent variable is often the most critical—particularly with regression tools. Decision trees are generally the most flexible. \*Neural networks typically support most data types but some options require continuous numeric data.

Because each tool has slightly different results and interpretations, you need to consider the results you want to see as you select the data. If you find that a specific tool should be used for its interpretation, you might need to recode the data. For instance, you could recode categorical data into numbers (e.g., Female=1, Male=2) if you want to use regression, which requires numeric data. Some tools can automatically recode categorical data into discrete numbers. Conversely, many tools convert continuous data into discrete groups, sometimes called **discretized data**. However, the tools in this chapter generally work better by leaving continuous data in its original form.

### Missing Data

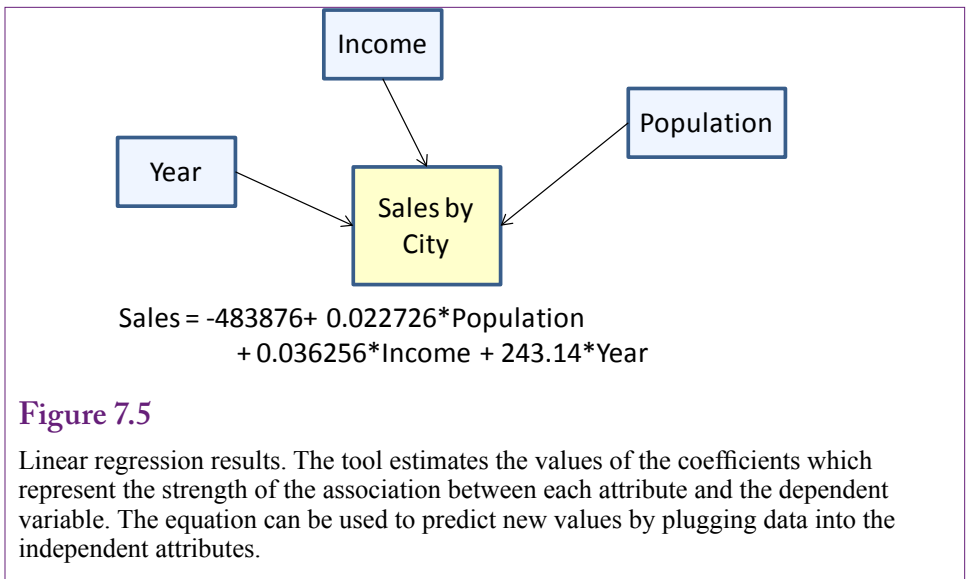
Missing data or null values can cause problems with some methods. In particular, tools that rely on continuous data do not support missing values. Also, all the attributes must have the same number of rows. Regression (linear and logistic) are the most restrictive of the tools in this chapter. If possible, the data should not contain missing values. If so, the most common approach is to remove those rows from the analysis. An alternative is to substitute values for the missing entries, such as computing the mean value of an attribute and using it to replace missing values. But, replacing values can alter the results, so be cautious.

Tools that support discrete values can tolerate null values. The missing value simply becomes another discrete case. For example, a gender attribute might contain *Female*, *Male*, and null values. Hence, the tool will evaluate three cases instead of two. So, if your data has a large number of null values, you should consider estimating decision trees or neural networks instead of regression equations.

## Linear Regression

### How does linear regression work and why would you use it?

Linear regression is one of the oldest data analysis tools. Its theory, properties, complications, and applications have been studied for many years. It is a powerful statistical tool often used in research. When applied statistically, it is a fundamen-



tal tool of science that is used to prove or disprove hypotheses and to test models. However, when used for exploratory purposes, such as most data mining projects, the statistical validity often disappears. Although the tool is the same, the selection of the data and the way it is handled often violate the theoretical foundations of the regression statistics. In particular, for statistically valid results, the data must meet certain conditions—which are typically obtained through random samples. Also, you would have to start with an underlying model that can be tested. Repeatedly trying combinations of attributes and multiple models on the same set of data changes the statistical value of the results.

Because this book focuses on data mining and exploration, the details of the underlying statistics are not covered here. The reference section at the end of the chapter lists a couple of classic econometrics textbooks that provide details of how to use regression techniques appropriately in a scientific setting. But, you are probably not searching for scientific truths if you are using data mining. Instead, you are trying to see how various attributes might influence an outcome variable. Later, you can determine if you need to build a complete model, collect more data, and conduct formal tests.

If you know ahead of time that you will want to conduct a more scientific study, you should conduct your data mining on only part of the available data. Hold some of the data for use later—a random selection of rows might be useful.

## Goals

Regression is commonly used to determine how various attributes influence a selected dependent attribute variable. Linear regression estimates the coefficients of a linear model where each attribute has an independent effect on the dependent variable. These coefficients reveal the impact of each attribute. They are also easy to use when predicting the outcome of new data. Figure 7.5 shows the basic result using three independent attributes. The objective of linear regression is to find the coefficients that best fit a linear relationship. Linear relationships are useful because they are easy to understand. Also, they are relatively robust in the sense that small changes in the data usually have only a small impact on the coefficients.

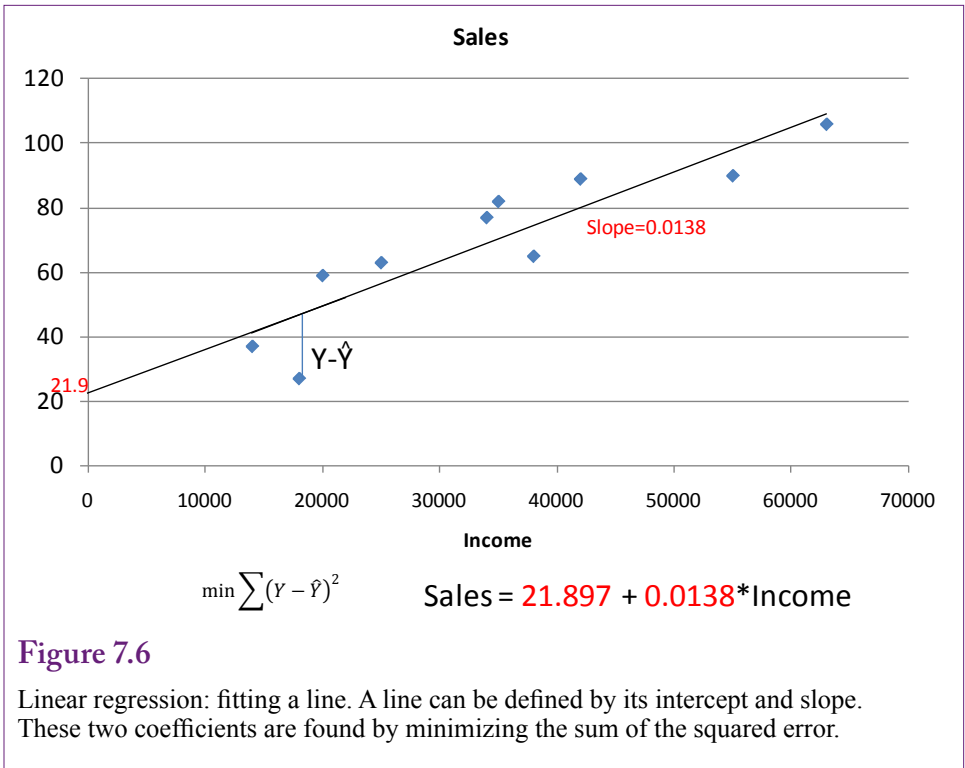


Figure 7.6 shows the regression process in more detail—using only one independent attribute. The objective of linear regression is to find the line that best fits the observed points. Best fit is usually defined as finding the line that minimizes the sum of the squared error—or the distance between the desired line and a given point. Squared error is important because the deviation could be positive or negative and squaring forces it to be a positive value. Historically, squared values were chosen for two reasons: (1) The result is mathematically easier to analyze, and (2) Outliers or values that are far away from the line have a stronger impact. Today, with fast computers, some systems have the ability to minimize the mean absolute deviation—which uses the absolute value instead of the squared value of the difference. You might use absolute values if you need to decrease the impact of outliers—such as when you believe the outliers are strange or unlikely values. Mathematically, there are some relatively fast methods to find the linear coefficients. Some tools use this direct approach, others use a broader search method. Either way, the tools always report the values of the coefficients.

But, what do the coefficients mean? The best way to understand the value of the linear regression coefficients is to apply a little bit of math. From the two-dimensional case, it is clear that the coefficient on the X-attribute is the slope of the line. This interpretation is true for multiple coefficients as well. Recall that the slope of a line is:

$$dy/dx = \text{change in } y / \text{change in } x$$

Or, in words, the slope coefficient answers the question: If the x-value increases by one unit, what happens to the y-value? Consider a simple sales example where

the income coefficient equals 0.0138. It seems like a small number, but income is evaluated in thousands. So, if the firm can attract customers who make 1000 more than the existing customers, how much will sales increase? The answer is:

$$\text{slope} * 1000 = 0.0138 * 1000 = 13.8$$

That value might not seem like much, but if the average sales value is 69.5, it would represent a 30 percent increase in sales. The point is that you can use the coefficients to predict what will happen if the underlying attribute values change. The coefficients also reveal which attributes are more important than others, so you know where to focus your efforts if you want to change an organization.

Regression coefficients are found by minimizing the sum of squared errors. In matrix terms, the outcome Y variable consists of a matrix of n rows and 1 column. The X observations form a matrix of n rows with k columns of X-attributes. Usually, the first column values are 1 to handle the intercept constant. The errors are written as the actual y value minus the estimated value. The estimated value is XB, where B is the 1 x k matrix of linear coefficients. Then, the goal is to find the B values that minimize:

$$e'e = (y - XB)'(y - XB) = y'y - BX'y + B'X'XB$$

Differentiating the squared error term with respect to the B values to find the minimum point leads to the optimal value for B:

$$B = (X'X)^{-1}X'y$$

Now, as a data analyst you do not need to memorize the formula for finding the coefficients. But, it does make a couple of things clear. First, finding the linear coefficients entails computing the covariance matrix  $X'X$  and then finding its inverse. Several computer algorithms exist to find the inverse efficiently. But, some problems can arise—such as the covariance matrix not having full rank—meaning it is not invertible (similar to the issue of not dividing by zero with constants). This situation most commonly arises when the X-attributes chosen are dependent on each other (linear combinations).

## Data

Regression is a powerful tool, but it is somewhat restrictive in terms of the types of data that can be analyzed. One of the most critical constraints is that the dependent variable must contain continuous numeric data. This data should not be constrained or truncated (e.g., greater than zero, less than 100). Econometricians have devised methods to handle some typical issues with data, but the common data mining or BI tools do not implement most of these changes. High-end (expensive) statistical packages implement these tools, but you probably need an experienced analyst to configure the data and interpret the results.

Data for independent attributes must be numeric, but it can be discrete. Categorical data can be recoded into numbers. Depending on the source of your data, you can use queries (such as a CASE statement) to convert discrete text values to numbers. Or, within a data mining tool, you can create a new calculated variable to handle the conversion. SQL Server uses the same CASE statement in queries and as an expression within a data view.

Figure 7.7 shows the use of the CASE statement in a SQL Server SELECT statement. You simply specify the comparison column (Gender) and then list each of its possible category values in a WHEN clause. The corresponding THEN state-

	CustomerID	Gender	NewGender
<pre> SELECT CustomerID, Gender, CASE Gender   WHEN 'F' THEN 1   WHEN 'M' THEN 2   ELSE 0 END As NewGender FROM Customer </pre>	1	F	1
	2	M	2
	3	M	2
	4	M	2
	5	M	2
	6	M	2
	7	M	2
	8	F	1
	9	M	2

**Figure 7.7**

Using SQL Server CASE to recode data. The CASE statement can be used in a SELECT query to convert text values in the Gender column into numeric values that could be used in a regression model. You need to list each category in a WHEN clause, using the THEN statement to define a unique number to the category.

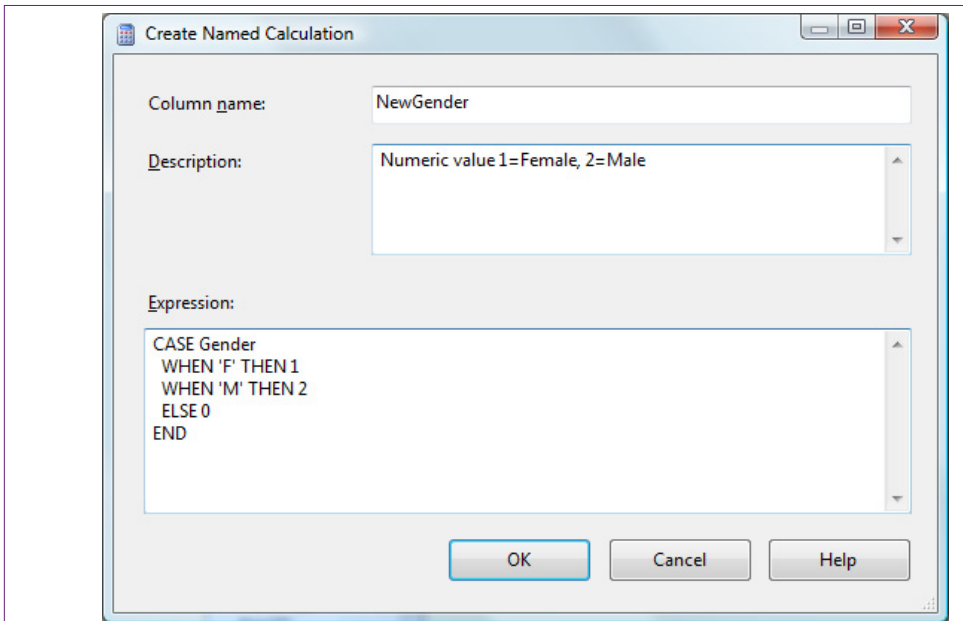
ment converts the text value to the specified number. Note the use of the ELSE statement to handle missing or invalid data. This new column NewGender could then be used in a regression analysis.

Figure 7.8 shows the same CASE statement being used to define a new Named Calculation within a Data Source View in the data mining tool. Notice that the CASE statement is identical to the one used in the SQL query. It is often helpful to test these conversions in a query first so you can run the query and correct any problems that arise as you are developing the formula. The basic steps to create a named calculation in the SQL Server BI tool are:

1. Open the data source view.
2. In the Design screen, right-click the Customer table.
3. Choose New Named Calculation.
4. Enter the values shown and click OK.

After you create the expression in the data mining tool, you can right-click the table and browse it to see the converted values to ensure the expression is working correctly. You should always verify your work as you go. If you make a mistake in the CASE statement, it can be very difficult to find later. Worse, you might never see the error, run the regression analysis and reach the wrong conclusions because the data was bad.

Missing data is usually not allowed in regression analyses. Some tools tolerate columns with missing data, but they generally resolve the problem by discarding the entire observation. Because discarding the row is usually the best option, this approach is acceptable. Hopefully, the tool will warn you about missing data so you can go back and verify that you have included the correct data. In some cases, you might want to discard an attribute if it has too many missing values. If only one attribute is causing most of the problems, you are generally better off without that column—because it will cause the system to discard useful data from



**Figure 7.8**

CASE statement in data mining tool. The CASE statement can be used as an expression to recode text values in the Gender column into numeric values that could be used in a regression model. You need to list each category in a WHEN clause, using the THEN statement to define a unique number to the category.

the other attributes. For example, you might have an Age column that you want to use to predict sales. But, customers are typically reluctant to reveal their age, which leads to have many missing data points. If the Age attribute is included in a regression model, all data associated with that customer will be discarded. The results will be based only on the data from customers who did report their ages—which is probably not a representative group, and could be quite small. When you encounter missing data, you will have to decide whether to discard the rows or the specific column—based on the number or percentage of missing values; and on how important the column might be to your exploration.

## Tools

Many tools exist to perform linear regression. Microsoft Excel even has a built-in regression tool under Data/Data Analysis. (Although, you probably have to install the Analysis Toolpak Add-in to find it.) Regression is also a common feature of most statistical packages. These packages provide detailed control over the analysis, can handle complex data problems, and provide detailed results and evaluation statistics. However, as external packages, you would need to find a way to integrate or convert your data to the statistical software.

Microsoft BI does support linear regression—but it uses a unique computational approach. In particular, it relies on decision-tree methodologies to evaluate each attribute specified in the model. Consequently, although Microsoft BI eventually does produce estimates of the equation, the results might not exactly match those generated by other tools. Additionally, statistical packages provide more detailed



```
SELECT City.CityID, Population2000, Income2004,  
       YEAR(Bicycle.OrderDate) As SaleYear, SUM(SalePrice) As TotalSales  
FROM City  
Inner Join Customer  
       ON City.CityID=Customer.CityID  
INNER JOIN Bicycle  
       ON Bicycle.CustomerID = Customer.CustomerID  
WHERE Population2000 Is Not Null  
       And Income2004 Is Not Null  
GROUP BY City.CityID, Population2000, Income2004, YEAR(Bicycle.OrderDate);
```

### Figure 7.9

SQL query to get sales by city. Note the use of the Bicycle, Customer, and City tables. The YEAR function returns just the year portion of the date. SUM computes the total sales. The query returns the total sales to each city along with the income and population of the city, plus the year of the sale.

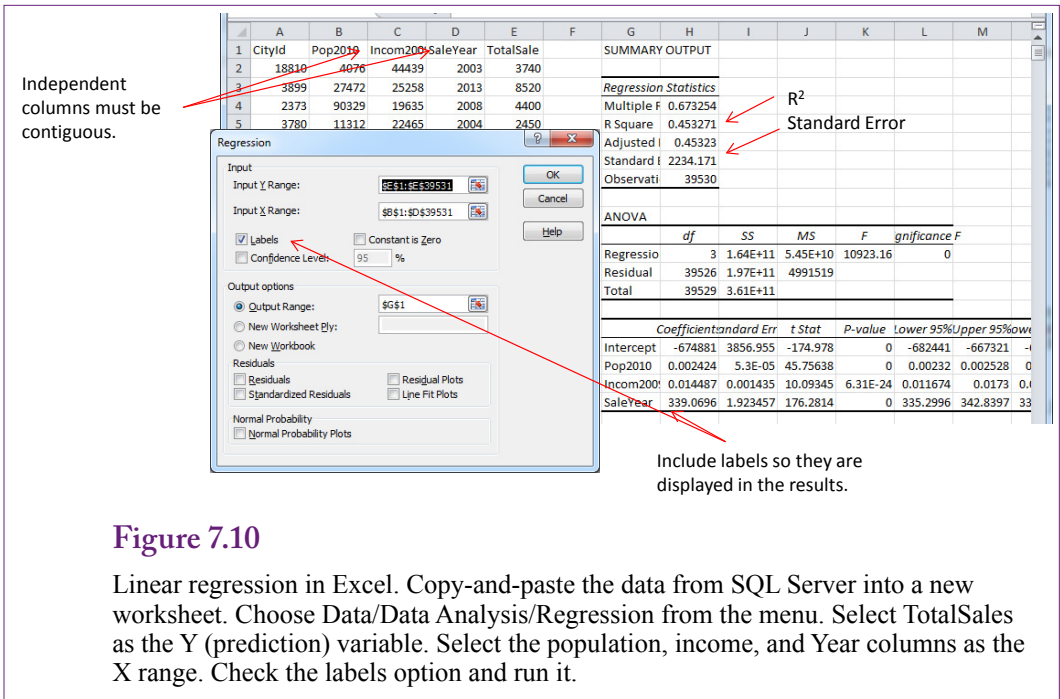
information about the estimates. Still, it is usually more convenient to use the tools provided within the data mining system. To understand some of the differences, results from both SQL Server BI and traditional regression are covered in this section.

#### *Traditional Least-Squares Regression*

Consider the traditional tools first. Organizing the data properly is the first step to using these tools. Typically, data has to be in a flat file of simple columns and rows. Each column represents a single attribute. Each row contains one observation. The Rolling Thunder Bicycle Company presents a useful example. The managers are thinking about expanding marketing in specific cities. Because the company sells high-end bicycles, the managers think that wealthier consumers are probably a better target. But, they also suspect it is important to target larger cities. In browsing the database for the company, you might have noticed the City table—which lists thousands of U.S. cities along with their population (from the 2010 Census) and per-capita income (from 2009 Census data). The Bicycle table makes it easy to find the sales to each customer, and each customer is linked to the City table. It is probably a good idea to include the year of the sale to control for any trends over time. To obtain the data, you create a basic query that retrieves the total sales by city, along with the population and income columns. You can obtain the sale year by using the YEAR function.

Figure 7.9 shows the SQL query. The YEAR function returns just the year portion of the date and SUM computes the total. Notice the GROUP BY clause needs the CityID, plus the population, income, and year attributes. This query produces the total sales by city and year and it drags along the population and income values for each city. Each row represents sales for one city and one year. To run the regression analysis, you need to first run the query and copy or export the data. For small and mid-sized problems, it is relatively easy to copy the data from the query and paste it into an Excel spreadsheet.

The Data/Data Analysis/Regression tool in Excel is a quick way to estimate linear regression coefficients. It does not support missing data and the independent (X) attributes must be in contiguous columns (a



single group with no column gaps). Figure 7.10 shows the basic setup. Simply pick the Y (predictable) column, then the X (independent) columns. Set the Labels checkbox to ensure that the column names are displayed in the output. Choose where you want the results to be displayed and click the OK button to run the regression analysis. The results in this case were displayed on the same worksheet. The coefficient values are displayed in the table. The results include additional useful information and these values are described in the Results section.

### *Microsoft Data Mining Linear Regression*

Instead of the well-known least-squares approach, Microsoft data mining uses a specially-configured version of its decision tree tool to estimate linear coefficients. This approach automatically attempts to break the data into a decision-tree framework as well. However, this section focuses on the linear regression approach; the decision tree issues are covered in a later section. A critical aspect to Microsoft's approach is that the data set must contain a single key column that identifies each row of data. This requirement arises because of Microsoft's approach—it is not a factor in regression analysis. But, you must be extremely careful to set this column correctly. It affects the estimation results.

Setting up the data for linear regression is a little tricky. Most traditional linear regression analyses need data at a single level—in one table or query. Additionally, you might need to compute subtotals before running the regression analysis. Hence, it is usually easiest to create a named query to get exactly the data you need. A **named query** is simply an SQL statement stored within a data source view. It is often used to compute subtotals, concatenated key columns, and select or exclude data to be analyzed. It uses the power of SQL to choose exactly the data needed. To illustrate, consider the bicycle case where the goal is to identify

```

SELECT CAST(dbo.Customer.CityID AS nvarchar) + N'-' + CAST(YEAR(dbo.Bicycle.
OrderDate) AS nvarchar) AS CityYear,      dbo.City.Population2000,
      dbo.City.Income2004, YEAR(dbo.Bicycle.OrderDate) AS SaleYear,
SUM(dbo.Bicycle.SalePrice) AS SaleTotal, dbo.Customer.CityID
FROM  dbo.Bicycle
INNER JOIN  dbo.Customer
      ON dbo.Bicycle.CustomerID = dbo.Customer.CustomerID
INNER JOIN  dbo.City
      ON dbo.Customer.CityID = dbo.City.CityID
WHERE (dbo.City.Population2000 IS NOT NULL) AND (dbo.City.Income2004 IS NOT
NULL)
GROUP BY CAST(dbo.Customer.CityID AS nvarchar) + N'-' + CAST(YEAR(dbo.Bicycle.
OrderDate) AS nvarchar), dbo.City.Population2000,  dbo.City.Income2004,
      YEAR(dbo.Bicycle.OrderDate), dbo.Customer.CityID

```

**Figure 7.11**

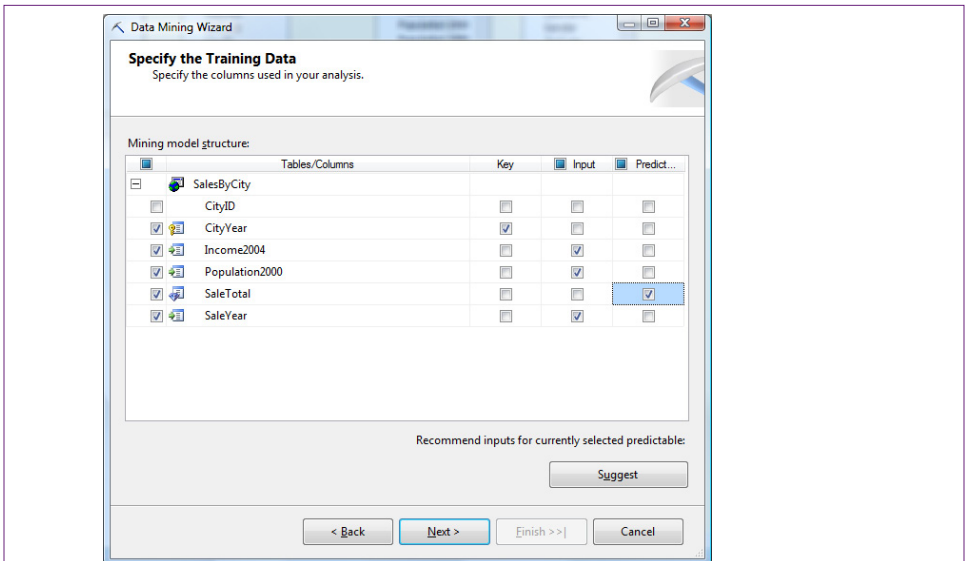
A named query to compute subtotals for sales by city and year. Notice the use of the Cast statement to create a unique key that includes both the CityID and the Year. Also, missing data for income and population are excluded.

attributes that affect sales by city. This approach requires the total sales to each city for each year, plus it needs the population and per capita income. Technically, it would be nice to have population and income for each year; but that data is difficult to obtain.

Named queries are created within a data source view. If you do not have one already, you should create a new data source view that includes at least the Bicycle, Customer, and City tables. Then add a new named query. Figure 7.11 shows the SQL for the named query. Notice the use of the Cast function and concatenation to obtain a key column (CityYear) that uniquely identifies each row. The rest of the query is standard SQL for computing subtotals using the GROUP BY statement. You can test the query as you build it to ensure that it works and retrieves exactly the data needed.

Once the data is available, you can right-click Mining Structures in the Solution Explorer and create a new data mining structure. Choose the linear regression option and pick the data source view that holds the named query you created. Select that query as the Case table. Figure 7.12 shows the selection of the attributes. Ensure that CityYear is selected as the key column that identifies the data. Set SalesTotal as the predictable column. For Income, Population, and SaleYear, check the boxes to make them input attributes. Finish the wizard's steps with the default values and give it a name you will remember later (perhaps Sales by City regression). Notice that 30 percent of the observations will be held out to use for testing the model. If you want to come closer to a traditional regression package, you can set this value to zero so that all rows are used in the estimation.

Once models are defined in Microsoft's analysis project, you have to deploy and process them. Then you can browse the results. Right-click the mining structure in the Solution Explorer and choose the option to Process the model. Follow the steps to Run the process and close any windows it opens. Right-click the mining structure again and choose the option to Browse the results. This simple model should evaluate to a single level. The coefficients and the regression equation are displayed in the bottom-right corner of Visual Studio when you select the node.



**Figure 7.12**

Selecting attributes for regression. The key column must uniquely identify a row. The predictable column is the column you want to predict (SaleTotal). The other columns, population, income, and SaleYear are simple inputs.

## Results

It is relatively easy to set up and run data mining tools. The real key lies in interpreting the results. By understanding the model it becomes possible to analyze problems and predict possible outcomes based on that model. Technically, linear regression measures the correlation between variables, and correlation does not mean causation. That is, without a formal model and explanation of the underlying effects, it is possible that the results you see are simply one-time correlations of events that happened at the same time. Business and economic models are weaker than physical models. The results can reveal how various attributes move together with a dependent variable, but that does not guarantee the correlation will always exist. It does not mean that when the value of an independent variable is changed that the exact effect will always occur.

### *Traditional Regression Results*

Figure 7.13 presents the results from the traditional regression (e.g., Excel). The three most important elements are: (1) The coefficient values, (2) The T-statistics that evaluate each coefficient, and (3) the  $R^2$  value which indicates the accuracy of the overall model. The values shown in this figure came from the Bicycle sales by city and year, but results might vary depending on which data was used.

The coefficient values are important because they are the slope or change in the Y value that will occur for a one-unit change in the input X-value. In the example, sales increase by 339 each year. An increase of 1,000 in Income increases sales by 14.35. You have to pay attention to the units of the underlying data. Yes, 339 appears to be much larger than 0.0145, but Income is generally discussed in thousands. Although a \$1 increase results in less than a penny increase in sales, a \$1,000 increase of per capita income is associated with a \$14.35 increase in sales.

$$\text{Sales} = -674881 + 0.002424 \text{ Population} + 0.014487 \text{ Income} + 339.07 \text{ Year}$$

(-175.0)	(45.76)	(10.09)	(176.28)
----------	---------	---------	----------

$$R^2 = 0.4533$$

**Figure 7.13**

Traditional regression results. Regression provides an estimate of the coefficient values, the T-statistic which is a measure of a coefficient's accuracy, and  $R^2$  which is an overall measure of the accuracy of the model.

Along the same lines, just because a number looks big does not mean it is significantly different from zero. In statistics terms, the coefficient value is a mean. And means are estimations that are subject to error. However, least-squares regression also provides a measure of the standard error of the coefficient. Dividing the coefficient by the standard error gives the T-statistic for the coefficient. This value is used in a simple hypothesis test to see if the coefficient is significantly different from zero. Loosely, if the T-statistic is larger than 2 (in absolute value), then the coefficient is significantly different from zero. Conversely, T-values less than that (e.g., 1.00) indicate that the variation is too high and the coefficient is effectively zero. Hence, the attribute has no important effect on the dependent variable. Technically, T-statistics are evaluated by a T-test which incorporates the degrees of freedom; but with a sufficiently large number of observations, 2 is approximately the value of the critical T-statistic for a test with 5 percent error.

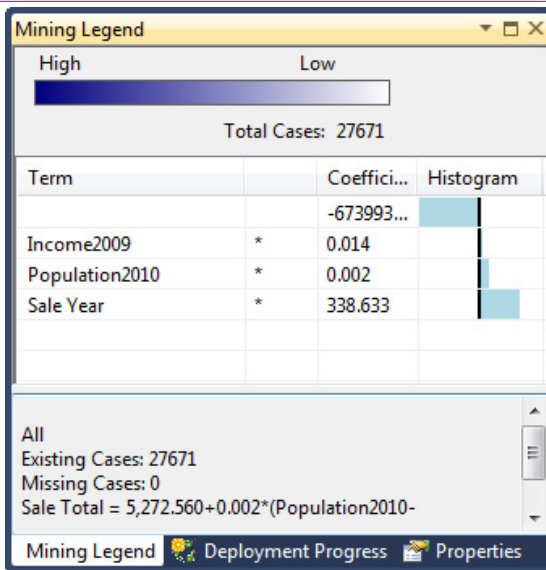
The  $R^2$  value can be interpreted as a percentage of the variation in the Y-values that is explained by the independent X variables. The number ranges from 0 to 1, with 1 representing a complete explanation and a perfectly straight line. Low values indicate that your choice of X variables explains only a small percentage of the variation. How low is low, or how high should the values be? It depends on the problem. In the example, 26 percent seems a little low, but there is a lot of underlying variation in the data. It is unlikely that you will be able to find a better model. Still, you could look to find other attributes that might improve the model. Low  $R^2$  values are indicators that your model might be missing some key attributes.

Taken together, the statistical results of linear regression provide useful information about a process. You can see which attributes have strong (or no) effects on the dependent variable. You have actual measures of the correlation, which you can use to predict future outcomes. And you have a measure of the strength of the model helping you decide if you have examined the proper set of attributes and the linearity.

### *Microsoft BI Results*

Because Microsoft's linear regression tool is based on its decision tree method, it produces somewhat different output than found in least-squares regression. Figure 7.14 shows the results from the linear regression data mining tool. Although not shown, the tool found a single node, which means the coefficients found for the equation should be close to those from traditional regression.

First, notice that the coefficients are presented two ways: In a table and as an equation. The values are the same—except for the intercept term. In the equation, each independent variable is written as a deviation from its mean. The standard intercept value can be computed by plugging in zeros for each of the variables, or it can be read from the top row of the table.



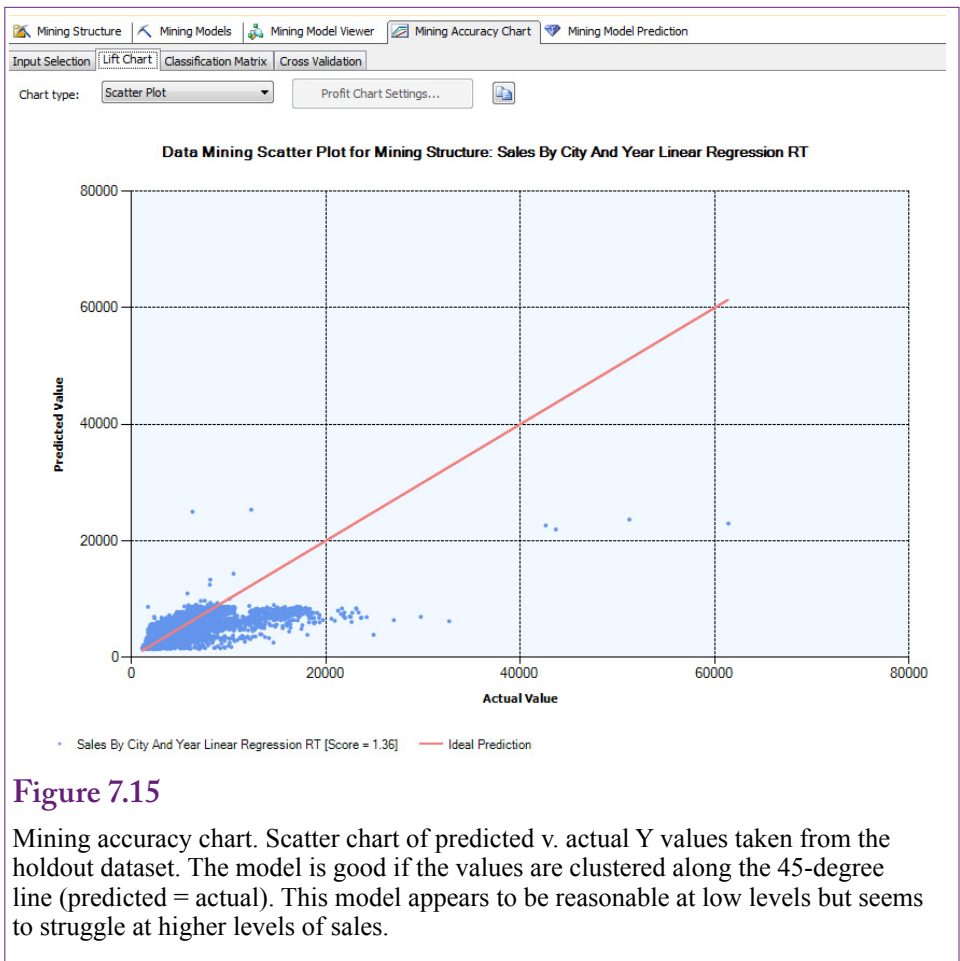
**Figure 7.14**

Microsoft regression results. With a single node, the decision-tree-based approach returns coefficients similar to those of a traditional regression. But, the tool make its own decision about significance of variables and does not display the standard errors or the R<sup>2</sup> value.

Second, observe that the coefficients are close to the values estimated with traditional regression—but not exactly the same; The reason for the difference in the values is because the data mining tool held out some of the data to be used to test the model. The least-squares approach used all of the rows. Because the holdout data was randomly selected, the results will be similar but not exactly the same.

More importantly, the tool does not report standard errors or the R<sup>2</sup>, so it is harder to determine the importance of the variables and the model. Instead, Microsoft generates some accuracy charts that use the holdout data set to evaluate the model. These options appear in most of the data mining tools. Click on the Mining Accuracy Chart tab to begin the process. After verifying the input choices (test cases), you can select the Lift Chart and Cross Validation tabs. Because linear regression uses a continuous variable the Lift Chart tab actually shows a scatter chart comparing predicted to actual values. The values are taken from the holdout dataset and the predicted values are computed from the regression equation coefficients. Figure 7.15 shows the scatter plot for the example. In a good model, the points will cluster around the 45-degree line where predicted equals the actual value. Here, the model appears reasonable at low levels but it seems to be truncated and non-linear so that at the higher levels, the predicted values are consistently low. For values less than 10,000, the results appear to be clustered around the equality line. However, the model does not predict well for values greater than about 15,000. The model appears to truncate its predictions and has trouble forecasting larger numbers. Most likely, the model needs an additional attribute or perhaps the coefficient on population is too low. It is possible that the population effect is nonlinear.





**Figure 7.15**

Mining accuracy chart. Scatter chart of predicted v. actual Y values taken from the holdout dataset. The model is good if the values are clustered along the 45-degree line (predicted = actual). This model appears to be reasonable at low levels but seems to struggle at higher levels of sales.

Microsoft's data mining tool also uses a cross-validation tool to evaluate the overall performance of a model. The tool has more options for a discrete Y-attribute, but it does have a way to evaluate continuous data used in linear regressions. Essentially, cross validation splits the training data set into partitions or folds. Each partition represents one test—the data from all other partitions is used to build the model, which is then applied to the test data and evaluated. The goal is to determine if a model is susceptible to a specific set of data. By building and testing on different pieces of data, models that work on one segment of data but not others will be easy to spot.

In the Mining Accuracy Chart tab, switch to the Cross Validation tab. Set the Fold Count (number of partitions) to 5, set the Max Cases to 0, which uses all of the non-holdout data. The Target Attribute should already be set to the dependent Y-variable (Sale Total).

A big problem that can arise in data mining is that any model estimated is heavily dependent on the specific data chosen. When you have enough observations, you should hold back some of the data to use for testing. Additionally, Microsoft provides a cross validation tool. Cross validation splits the training data into partitions or folds (ignoring holdout testing data). Each partition will represent one test of the model. Iterating through each partition, the system uses the data in the



Fold Count: 5 Max Cases: 0 Get Results

Target Attribute: Sale Total Target State: Target Threshold:

Sales By City And Year Linear Regression RT				
Partition Index	Partition Size	Test	Measure	Value
1	5534	Estimation	Root Mean Square Error	2195.5291
2	5534	Estimation	Root Mean Square Error	2201.3898
3	5535	Estimation	Root Mean Square Error	2217.2016
4	5534	Estimation	Root Mean Square Error	2263.8073
5	5534	Estimation	Root Mean Square Error	2277.0138
			Average	2230.9878
			Standard Deviation	33.2228
1	5534	Estimation	Mean Absolute Error	1430.1982
2	5534	Estimation	Mean Absolute Error	1422.3427
3	5535	Estimation	Mean Absolute Error	1404.7062
4	5534	Estimation	Mean Absolute Error	1451.5387
5	5534	Estimation	Mean Absolute Error	1439.3744
			Average	1429.6311
			Standard Deviation	15.8091
1	5534	Estimation	Log Score	-9.1135
2	5534	Estimation	Log Score	-9.116
3	5535	Estimation	Log Score	-9.123
4	5534	Estimation	Log Score	-9.1441
5	5534	Estimation	Log Score	-9.1502
			Average	-9.1294
			Standard Deviation	0.015

**Figure 7.16**

Cross validation results. Data is split into the number of specified partitions (folds) and the model is estimated and tested in each partition. The error terms should be the same across the partitions.

“other” partitions to estimate the model coefficients. It then computes error statistics using the data in the targeted partition. So, you end up with overlapping tests of the model—one test for each partition. If the error measures are the same in each partition, then the model is considered robust and not susceptible to quirks in the data. If you see major differences in the error measures across partitions, then your model is highly dependent on specific data and you should look for a better model—or at least additional attributes.

Figure 7.16 shows the cross validation results using five partitions. Note that Max Cases specifies the number of observations to use—entering 0 tells the system to use all of the training rows. For linear regression, **root mean square error (RMSE)** is the most common measure. It is computed as the square root of the mean (average) squared error. The formula is almost the same as that used to compute the least squares estimates:

$$RMSE = \sqrt{\frac{1}{n} \sum (Y - \hat{Y})^2}$$

In fact, least-squares regressions return this value as the standard error. Check Figure 7.10 to see the value of 2234 computed by Excel. The average of 2230 computed by cross validation is close to that value. The cross-validation tool also reports the **mean absolute deviation (MAD)** which uses the absolute value to compensate for negative values instead of squaring the errors:

$$MAD = \frac{1}{n} \sum \text{abs}(Y - \hat{Y})$$

The important result in both cases is that the values in the three partitions are almost identical. Technically, you could use the reported standard deviation to conduct simple T-tests if the numbers appear to be different. The conclusion is that the model does not depend on specific groups of data. There is still a chance that the overall data set is uniquely strange, but if it is representative of the population of data, the model should perform consistently in most cases.

Both error measures also convey information about the accuracy of the model. If the measures were zero, the model would have no errors and the data would fit a line perfectly. If you have two or more models using the same data, the model with the lower RMSE is the better model. There is no fixed value of RMSE that is considered good or bad because the number depends on the variation in the Y variable. Technically, the R<sup>2</sup> value can be computed from the RMSE and the sum of squared Y-values ( $1 - \text{sum of squared error} / \text{sum of squared Y}$ ), but if you want R<sup>2</sup> just use least-squares regression to compute it for you.

### Attribute Evaluation

The R<sup>2</sup>, RMSE, and coefficient values are all indicators of the stability of the model. If the R<sup>2</sup> is reasonable and the results show that at least some of the coefficients are significant and have signs that make sense, the coefficients tell you which attributes have important influences on the dependent attribute. The coefficients passing the significance test are the critical ones. These are the items to concentrate on. Pay attention to the signs. A negative sign means increases in the attribute level will decrease the dependent variable. Also, look at the attributes to find ways that the values can be changed.

If an attribute is completely outside of your control, it can be important, but you might not be able to do anything about it. In the example, the sales year is a strong factor, and per capita income of the city is important. Population of the city is also an important factor for increasing sales. At first glance, it might appear that all three variables are outside of management control. The coefficient on the Year attribute simply states that sales for next year depend heavily on what they were in the prior year—for a given city. It is not possible to change the year, but if sales can be increased in the current year, that pattern should carry forward into succeeding years. In other words, increased efforts at selling today will also influence purchases in the future.

City population and per capita income seem even more distant. While it is true that you cannot change either of those values for a given city; you can choose which cities you want to focus on for a marketing campaign. The answer is to focus on larger cities with higher per capita income.

One issue you need to watch when looking at attribute coefficients is the magnitude (size) of the coefficient. A value of 339 on the Year attribute appears much larger than 0.0145 on the Population coefficient. Yet, the size of the coefficient depends on the values used in the attributes. It is often helpful to use a trick from economics and use an elasticity measure instead. **Elasticity** is the percent change in the dependent variable divided by the percent change in the input variable. It indicates how much the dependent variable will change for a one percent increase in the independent variable.

Attribute	Coefficient	Average	Elasticity
Sales		5272.92	
Year	339.07	2004.66	128.908
Population	0.00242	29505	0.014
Income	0.01449	25060	0.069

**Figure 7.17**

Elasticity. Percent change in Y divided by percent change in X is the slope coefficient times the average X value divided by the average Y value. Elasticity is a pure number that does not depend on the units of the attributes, which makes it easier to compare across attributes.

$$\text{Elasticity} = \frac{\text{percent change in } Y}{\text{percent change in } X} = \frac{\% \Delta Y}{\% \Delta X} = \frac{\Delta Y / Y}{\Delta X / X} = \text{slope} \frac{X}{Y}$$

Because the regression coefficient is the slope, elasticity is relatively easy to calculate—particularly at the average point. Simply multiply by the average value of the X and divide by the average of Y. Elasticity is a standalone number and the values can be compared across all of the attributes.

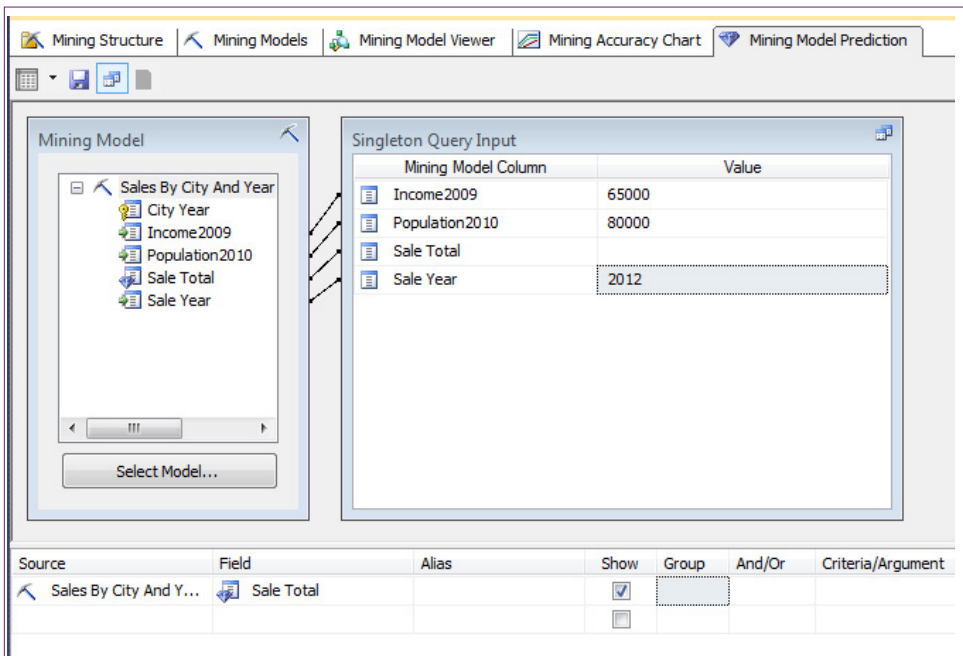
Figure 7.17 shows the computation of elasticity for the sample problem. Looking at the elasticity column, the year is the most important factor—although years can only increase in one-unit amounts, so a percentage increase is hard to understand. The elasticities for population and income are more interesting. Both values are low, but sales are about five times more sensitive to increases in income than to increases in population. So, any marketing campaign should focus on wealthier cities. Note that the effects are additive, so by increasing marketing to cities that are one percent over average in income and in population, sales should increase as a percentage by the sum of the two numbers.

## Prediction

Regression is a key tool for predicting outcomes. The estimated linear relationship makes it relatively easy to predict values for the dependent variable for any combination of input attributes. Simply multiply the attribute values by the respective coefficients and add up everything. Make sure to include the constant intercept value. The result is an estimate of the resulting Y value. It is also possible to compute the variance of any predicted value.

$$s_y^2 = [\hat{x}(X'X)^{-1}\hat{x}]s_e^2$$

The value is written using standard matrix notation for regression. The X matrix consists of rows of data for the independent attributes, with a first column set to 1 for each row to handle the intercept. The smaller consists of the single row of data values to be forecast. And is the square of the standard error of the regression. The resulting variance enables you to compute a confidence interval for the estimated value. A confidence interval is useful because it provides a range of potential values instead of a single point. Values outside the interval are highly unlikely



**Figure 7.18**

Prediction with singleton query. Choose the mining model and click the Singleton query icon. Enter the x-attribute values you want to predict. Choose the mining model (Sales By City) in the Source row and the Y-attribute (Sale Total) as the field to be predicted. Switch to result view to see the predicted total (8446.5).

to occur. For example, a predicted value might be 6924, and a 95% confidence interval might be (6886, 6962). The confidence interval provides a great deal of useful information, because it identifies the most likely range for the prediction. On the other hand, if the confidence interval were (6545, 7304) because of a large standard error, the forecast essentially would be meaningless. In the example, because of the relatively high standard error, the actual confidence interval is (6848, 7000), which is a relatively wide interval, making it difficult to have much confidence in the forecasts.

From the formula, it is clear that the interval depends on the standard error of the regression. If the overall regression has a large error, any individual prediction will have high errors and a wide range. However, the formula reveals that is relatively difficult to estimate the variance of the predicted value. Fortunately, most statistical regression tools provide an option to compute and display both the forecast value and its variance, and often can display the confidence interval as well.

Microsoft BI has the ability to compute predictions—based on a set of X values. Data attributes can be entered into a sample table to be run as a group, or you can switch to a singleton query (toolbar icon) and enter just one row of data directly. Figure 7.18 shows the basic prediction pane for a singleton query. Enter the values of the X-attributes, then pick the mining model (Sales By City) in the Source column. Select the Y-attribute to be predicted (Sale Total). Switch to the Results view to see the computation value (8446 in this case). Notice that the tool

does not compute the standard deviation or confidence interval for the value. With some effort, it might be possible to compute the standard error using the formula and MDX commands. But, ultimately, if you want the statistical error results, it is better to use a complete statistical package.

## Logistic Regression

**How can attributes for discrete Y-values be estimated?** Linear regression is a powerful tool and has many extensions. For instance, it is relatively easy to handle polynomials simply by computing squared, cubic, or higher powers of the X attributes. It is also easy to use discrete X attributes by simply numbering the alternatives. But, linear regression always requires a continuous dependent (predictable) Y variable. Otherwise the results will be biased and predicted values will be difficult to compute correctly. The **logistic regression** method was created to handle discrete Y data. Technically, the binomial logit handles Y values of zero or one, while multinomial logit handles Y data with multiple discrete values. Binary values are common in problems where you are interested in whether some event happens or if a person or object falls into a specific class or not. For instance, if a person defaulted on a loan, or purchased any items, or paid with cash. Multinomial situations arise when an event has several discrete outcomes, such as five choices on a survey, payment methods (such as cash, credit, debit, or check), type of bicycle, or a specified category range (such as one-time, casual, common, or frequent customer).

### Goals

The problem is solved by estimating a function that determines the probability that the specified event happens. This probability is based on the independent X attributes. For instance, a marketer wants to know the probability that a person with a certain income, gender, and age makes a purchase. The marketer has observations on many people of various genders, ages, and income groups; as well as the outcome of whether a purchase was made. Early approaches tried to estimate a simple linear regression:

$$P = b_0 + b_1X_1 + b_2X_2 + b_3X_3$$

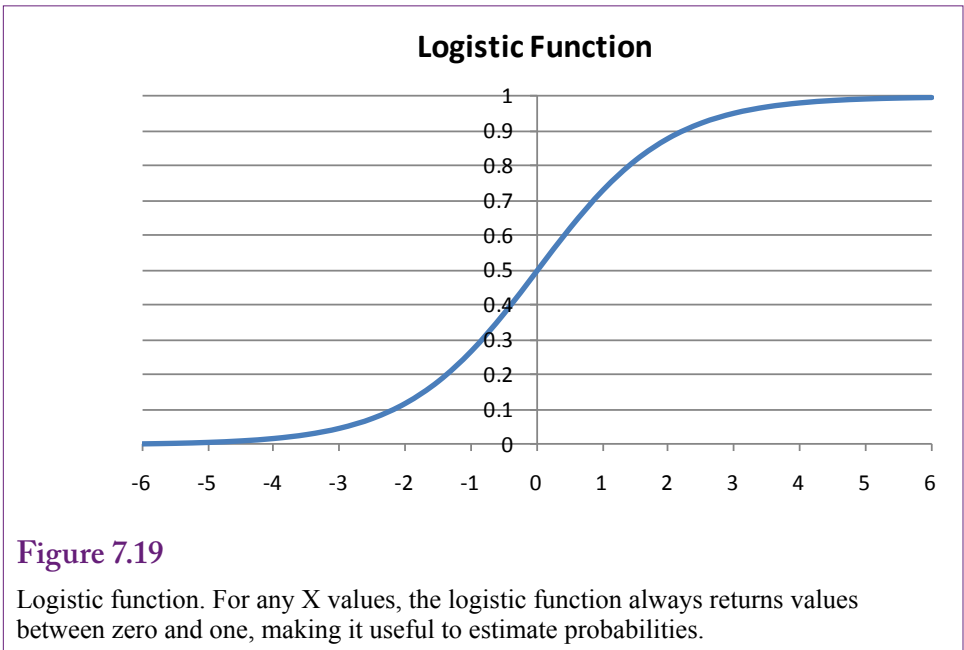
However, probability values must lie between zero and one and the linear regression forecast could generate almost any value for P. The solution is to transform the data to ensure that it stays between zero and one—regardless of the b and X values. Two common functions are the logistic and cumulative normal distribution. The logistic function is used most often because it is mathematically much simpler:

$$P = \frac{1}{1 + e^{-x}}$$

Figure 7.19 shows the logistic function. No matter what value of Z is chosen, the dependent value always lies between zero and one. The function is monotonic (increasing values of X always lead to increasing values of Y), so it does not distort the input values. In terms of estimating probabilities as a function of input attributes (X), the Z variable is written as a linear combination of the attributes:

$$Z = b_0 + b_1X_1 + b_2X_2 + b_3X_3$$

Intriligator [1978] shows another way to approach the problem that results in the same logistic solution. Begin by looking at the odds ratio or the odds in favor of an individual event, but use the natural logarithm:



$$\log\left(\frac{P}{1-P}\right) = b_0 + b_1X_1 + b_2X_2 \dots = Z$$

Taking the antilog (exp) of both sides and solving for P leads to the logistic function. Not that some algebra is involved, but the approach yields two useful points for understanding logistic regression. First, in the binary (0/1) case, it is possible to estimate the b coefficients by using the log-odds ratio as the dependent variable in a simple linear regression. Second and more importantly, in a traditional logit regression, the values computed from the sum of the linear terms results in the log of the odds ratio in favor of decision occurring. That is, once the b coefficients have been computed, it is easy to determine the odds (e.g., 5-to-1) that a customer with a given set of attributes will make a purchase. However, Microsoft BI does not use traditional logit regression to find the coefficients, so this relationship does not work when using that tool.

When the dependent variable consists of multiple choices, the mathematics and interpretation is similar. For example, managers of Rolling Thunder Bicycles might want to determine which attributes affect the purchase of each model type. Essentially, the traditional problem is written as a collection of binary dependent variables:  $y_1$  is set to 1 if a Race bike is purchased,  $y_2$  is set to 1 if a Road bike is purchased, and so on. However, this approach leads to estimation of separate equations for each value of the choice variable. In the bicycle example, the results will consist of separate sets of b coefficients for each model of bicycle. Judge [1985] shows how to compute the probability of each outcome. Each outcome j generates a set of  $B_j$  coefficients. An individual i has a set of  $X_i$  attribute values. To keep the equations easier to read, X and B are matrices, but you can think of the product as just the sum of the multiplications. Then the probabilities of a specific individual selecting outcome j are given by:

$$P_{i1} = \frac{1}{1 + \sum_2^J e^{X_i B_j}} \quad P_{ij} = \frac{e^{X_i B_j}}{1 + \sum_2^J e^{X_i B_j}}$$

The computation for the first probability is slightly different because the choices are normalized by using differences and assuming that  $B_1$  is zero. There is no value in memorizing these results, but they make it easier to understand the purpose of the regression. Essentially, the estimation of the B coefficients leads directly to the estimate of the probabilities of selecting each of the outcome choices. Typically, you can simply let the regression routine compute the probabilities for you in a prediction.

The key goal is that logistic regression estimates the B coefficients which indicate the importance of each attribute and the equations provide the ability to predict the probability that various choices will be made—based on different values of the X attributes.

## Data

Data for logistic regression is similar to that for linear regression—with the difference in the dependent predictable variable. The outcome to be predicted should be a discrete or categorical variable. It can be binary or it can contain multiple choices. In some cases, you can use a continuous dependent variable—by converting it into discrete groups or categories. BI tools, including Microsoft, provide options to discretize the data. However, in most situations, the data should already be in logical categories.

With traditional tools, all of the data will have to be numbers. If necessary, you can recode or use the SQL CASE command to assign numeric values to categories. As an example, consider the Rolling Thunder Bicycle Company with a goal of identifying attributes that affect the sale of the various model types. Model Type will be the dependent variable; independent attributes can consist of Gender, and SaleYear, and city characteristics of population and income. The data will have to be combined into simple columns and rows for export to an external tool. A basic query can be used to combine the data from the three tables: Bicycle, Customer, and City. As shown in Figure 7.20, the query can also compute SaleYear from the OrderDate and assign numbers to the ModelType and Gender variables.

Data for Microsoft logistic regression is simpler because it handles categorical data automatically so recoding is not needed. Ultimately, the tool is easiest to use if all of the data is collected into a simple named query that holds just the columns and rows needed for the analysis. The sample problem uses the Bicycle, Customer, and City tables. If you do not already have one, you need to create a data source view and add the three tables. Then create a named query, call it Bike-CityDetail, and use SQL to join the three tables and retrieve at least SerialNumber, Population2010, Income2009, Gender, and ModelType. To check for changes over time, you should also compute a SaleYear column as Year(OrderDate).

Why choose these particular columns? The ModelType will be used as the outcome or predictable column. The others are the only available attributes that might influence the decision of choosing a model type. Additional attributes might be considered, such as SalePrice, EmployeeID, and StoreID. Conceivably, some employees or retail stores might push certain models harder than other ones. Price might be an issue if there are large differences in prices across models. The goal is to select attributes that might have an effect on the outcome; and that might be



```

SELECT Bicycle.SerialNumber,    City.Population2010/1000 As Population,
      City.Income2009/1000 As Income,
      YEAR(Bicycle.OrderDate) AS SaleYear,
      CASE
        WHEN ModelType='Race' THEN 1
        WHEN ModelType='Road' THEN 2
        WHEN ModelType='Track' THEN 3
        WHEN ModelType='Tour' THEN 4
        WHEN ModelType='Mountain' THEN 5
        WHEN ModelType='Mountain full' THEN 6
        ELSE 0
      END AS NModelType,
      CASE
        WHEN Gender='F' THEN 1
        WHEN Gender='M' THEN 2
        ELSE 0
      END AS NGender
FROM   Bicycle
INNER JOIN Customer
      ON Bicycle.CustomerID = Customer.CustomerID
INNER JOIN City
      ON Customer.CityID = City.CityID

```

**Figure 7.20**

Converting categorical data to discrete numbers. The SQL CASE statement makes it easy to assign numbers to categorical data so that the data can be exported and used by external tools.

controllable in some fashion or at least helpful in explaining the decisions of consumers. But, for a first try, it is helpful to keep the list short so the results do not become overwhelming.

## Tools

High-end (expensive) statistical packages perform multinomial logit regression without too much effort. Some open-source econometrics packages, such as gretl, support logit (and probit) analyses. In any case, you would need to run the data query to generate numerical attributes, export the data to a format that can be read by the package, then configure and run the analysis within that system. Remember that the systems will return separate estimates of the B values for each outcome variable. These equations can be used to estimate the probability of a particular choice occurring.

Microsoft logistic regression, and often other BI tools, uses a different method to handle logistic regression. It turns out that many neural network systems use a logistic function to transform data for the same reason—to limit output values between zero and one. Consequently, the systems use decision trees and simplified neural networks to search for the best solution. As shown in the next section, the results are useful but often more fragmented. Consequently, the probability interpretations are altered.

Once the data view and named query have been set up, it is straightforward to configure Microsoft's logistic regression. Start a new mining structure and choose

Model	Constant	Pop	Income	Year	Gender
Race	-245.6**	0.00027**	0.0058	0.124**	-0.199**
Road	-225.8**	0.00013	-0.0062	0.114**	-0.203**
Track	1967.4**	0.00007	-0.0015	-0.986**	-0.273
Tour	-111.1**	0.00009	-0.0099*	0.056**	-0.251**
Mountain	5.20	0.00010	-0.0055	-0.001	-0.174**
Mountain full	-212.8	0.00014	-0.0072	0.108**	-0.205**

Cases correctly predicted: 30.3%

### Figure 7.21

Logistic results. The asterisks mean that the coefficient is significantly different from zero at the 5 percent level. All but one are actually significant at a 1 percent error level.

the logistic regression tool. Select the appropriate data source view and pick the named query as the Case table. Choose the attribute columns to match the problem: SerialNumber is the key column because the data is arranged by bicycle. ModelType is the Predictable column. For now, stick with a short list of Input (X attribute) columns: Gender, SaleYear, Income2009, and Population2010). Finish the wizard with the default values and give the model a name that you will recognize later.

## Results

Results for logistic regression are different for traditional tools versus specific data mining systems, particularly Microsoft's. The difference is not necessarily bad, but some interpretations are harder to obtain. Both approaches do a good job of evaluating the contribution of the independent X attributes. The traditional model is easier to convert to probabilities, while the BI tools are more focused on comparing the effects of the attributes.

### *Traditional Logistic Regression*

Figure 7.21 shows a set of results for the basic logit regression. Observe that one key to understanding the results is to compare the coefficients across the different choices. To keep the coefficients reasonable, the population and income values were converted to thousands. The resulting coefficients are still small. First note that population does not appear to influence the choice of model type, except for Race bikes. Second, income appears to affect only the Tour models. The lack of significance in the population variable seems unusual but it means that customers from almost any size city might purchase any model type. The one significant income effect is negative indicating a slight tendency for customers from lower-income cities to purchase the tour model. The tour bike makes some sense because it tends to be cheaper than race bikes.

In the example, a critical difference in the years is the negative sign on the Track bikes. If you look at the data, it will show that Track bikes sales were discontinued after 1995. The Gender term is instructive. Remember that it is coded as 0 for unknown/missing, 1 for female, and 2 for male. The relatively higher

Hybrid	NULL	9
Hybrid	F	396
Hybrid	M	601
Mountain	NULL	346
Mountain	F	2791
Mountain	M	4381
Mountain full	NULL	965
Mountain full	F	4554
Mountain full	M	7556
Race	NULL	877
Race	F	3743
Race	M	6483

Road	NULL	756
Road	F	3303
Road	M	5660
Tour	NULL	180
Tour	F	848
Tour	M	1370
Track	NULL	3
Track	F	28
Track	M	42

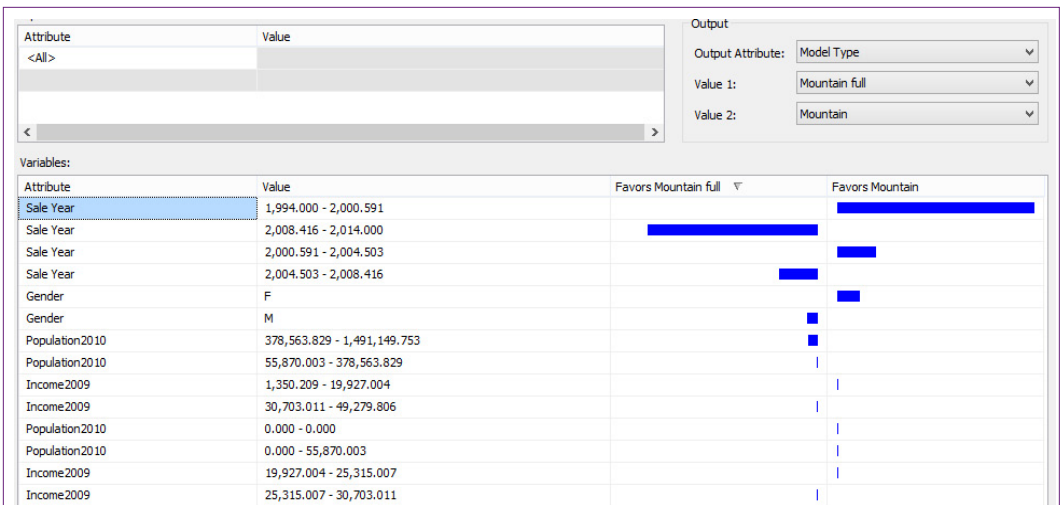
**Figure 7.22**

Bikes sold by ModelType and Gender. Notice that men buy more of every type. The null values coded as zero might cause problems. Rerunning the logistic regression without customers with missing gender removes the statistical significance from the Gender attribute for all except the full suspension mountain bikes.

coefficients on mountain bikes make sense, but the lower values on the track and tour bikes seem strange. Including the null values might cause some problems. However, it needs further investigation.

A quick way to investigate the situation is to run a GROUP BY query or go back and browse the data cube to examine total number of bikes sold by ModelType and Gender. Figure 7.22 shows the result. Notice that men buy more of all model types. However, the statistical results control for the other variables (population, year, and income), and indicate that after compensating for those effects, women buy more than men. Still, the result appears unusual—perhaps it is a quirk in the data. Another potential problem is the missing values for gender. Some customers used only a first initial and did not indicate a gender. Although those numbers seem low in most categories, it might affect the results. The logistic regression can be re-run after filtering out the customers with those missing values. The results are not shown here, because the coefficients are similar to those in the original regression. However, most of the Gender coefficients are no longer significantly different from zero. Gender remains negative and significant only in the case of full suspension mountain bikes. Intuitively, that is an unexpected result, but it might make sense—it could indicate that women want mountain bikes that are more comfortable and easier to ride on harsh terrain.

For this book, the conclusions regarding bicycle sales are unimportant. The key is to understand how to examine the initial results, look for patterns, and begin interpreting the results. When you see unusual or interesting items, use the tools to drill down and examine the underlying data. Be cautious regarding missing values.



**Figure 7.23**

Microsoft logistic regression results. The results are shown as comparisons between any two items that you select. This comparison between mountain and mountain full suspension bikes shows the year break when full suspension was introduced. It also shows small effects for income, population and gender.

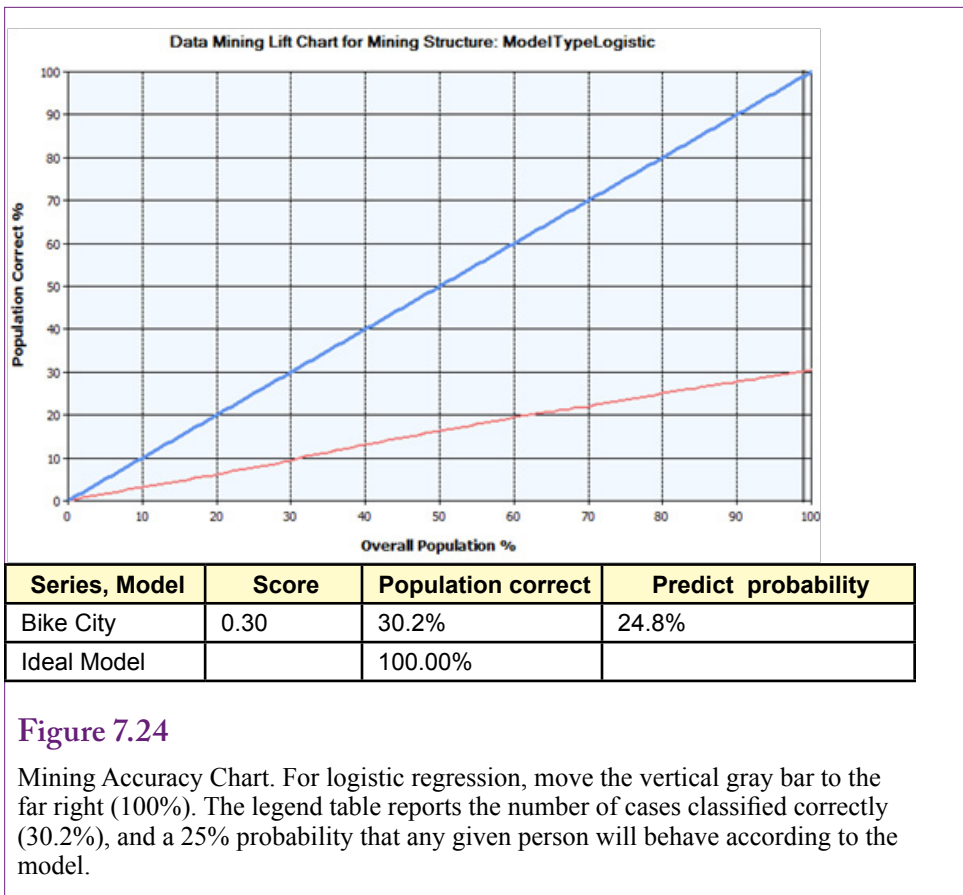
### *Microsoft Logistic Regression*

Microsoft's logistic regression is different from the traditional approach, so the results and interpretation are quite different. The key difference is that the logistic regression uses a decision tree approach; which automatically incorporates interaction effects. With this approach, the system examines the independent X attributes for breakpoint values that will alter the decision path. Yet, like the traditional logistic regression solution, the results depend on the individual model choices.

The results are easiest to understand with an example. However, as you will see, the results are difficult to present in print. Microsoft's presentation results are designed for interactive use. As shown in Figure 7.23, you select any two items from the dependent attribute (ModelType). The side-by-side comparison investigates the question of why consumers choose one model over the other. In this example, the importance of the Year shows that when full suspension bikes were introduced, consumers almost immediately switched to them. However, the gender effect does appear; along with tiny effects for city income and population (larger, wealthier cities lean towards full suspension). However, these city effects are small so they might arise simply because of the large number of sales of full suspension bikes (skew effect).

Note that Microsoft's approach automatically handles the missing gender values as a separate case. Rolling the mouse over one of the bars in the chart causes the system to report the underlying data for that item. In the case Gender for females (F), it reports 36 percent favor full suspension compared to 15.4 percent for mountain bikes. The percentage represents the probability of this range (female) for the specified outcome (full suspension or mountain).

The system also reports the lift for each value. In this situation, Microsoft defines **lift** as the impact of using this particular variable for predicting the speci-

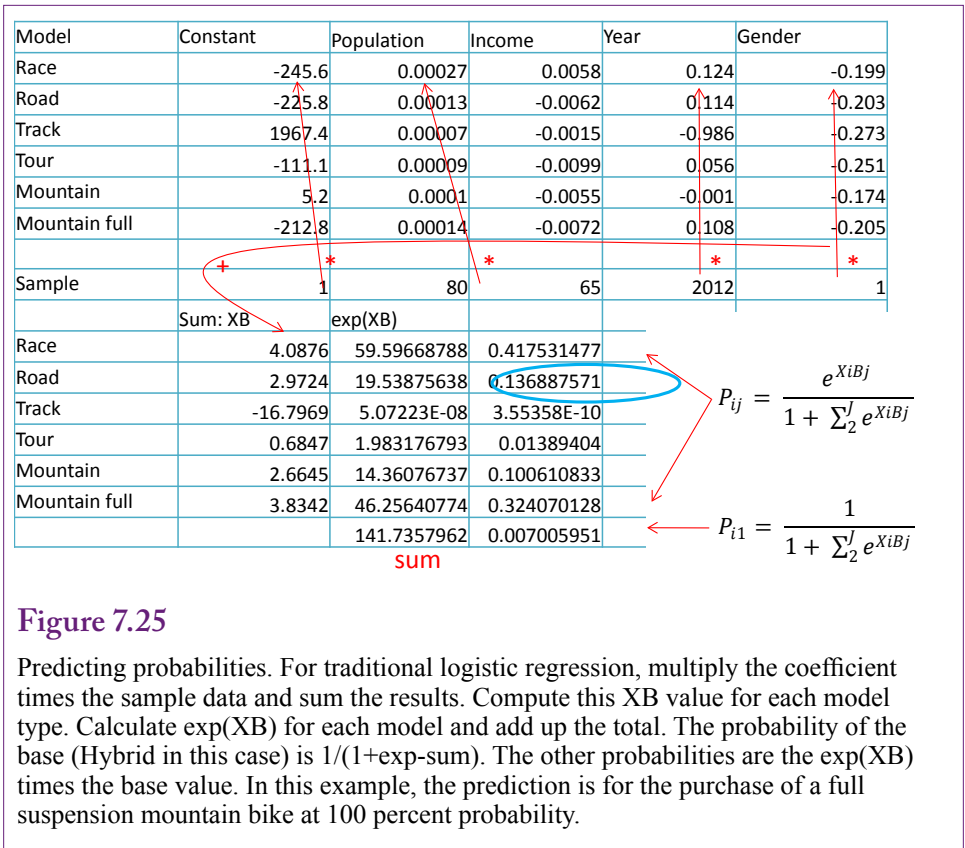


**Figure 7.24**

Mining Accuracy Chart. For logistic regression, move the vertical gray bar to the far right (100%). The legend table reports the number of cases classified correctly (30.2%), and a 25% probability that any given person will behave according to the model.

fied outcome. For females, the lift is computed to be 1.1 for full suspension and 1.0 for mountain bikes. These are relatively low values. Compare them to the lift for a sale year from 2008-2014, which yields 1.24 for full suspension and 0.55 for mountain bikes. In terms of developing the model and evaluating the attributes, it is much more important to include sale year than to include gender in the model. The size of the bars in the chart are based on a score computation that measures the effectiveness of the model using normalized data. Essentially, the data attributes are standardized by computing a Z score  $(X - \text{mean}) / \text{standard deviation}$ . The computation relies on probabilities for discrete values, but the effect is the same: it makes it possible to compare the coefficients across various attributes that have different measures. Recall the problem from the traditional regression—with income in the tens of thousands, years in thousands, and gender measured by ones—the coefficients are not directly comparable. Normalizing the data makes the coefficients comparable, and Microsoft generates a score value to show the relative strength of each attribute.

As shown in Figure 7.24, the results also include a Mining Accuracy Chart. Because the dependent variable is discrete, the chart and its interpretation are quite different from the chart created for simple linear regression. First, move the gray vertical bar to the far right so that the legend in the bottom right corner displays the statistics for the entire input population. This table is useful for comparing different models—although only the one logistic model is being used at the mo-



**Figure 7.25**

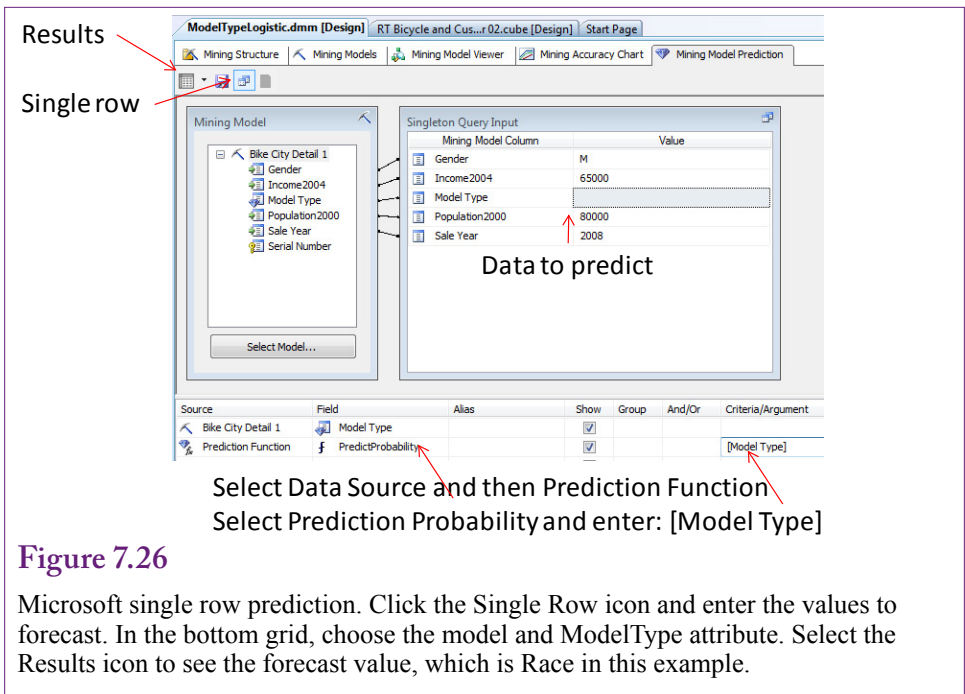
Predicting probabilities. For traditional logistic regression, multiply the coefficient times the sample data and sum the results. Compute this XB value for each model type. Calculate exp(XB) for each model and add up the total. The probability of the base (Hybrid in this case) is 1/(1+exp-sum). The other probabilities are the exp(XB) times the base value. In this example, the prediction is for the purchase of a full suspension mountain bike at 100 percent probability.

ment. Notice that the percentage of cases classified correctly is 30.1 percent which is close to the 30.2 percent value from the traditional logistic tool. The Predict probability value of 24 percent represents the probability that an actual person selected from the group would respond as predicted by the model. Both percentages are lower than you might prefer, so it might be worthwhile to search for a better model. Still, the model has some value and it has provided some insight into the importance of the various attributes.

### Attribute Evaluation

The examples make it clear that logistic regression does a good job of identifying the strength of the X attributes. The coefficients on the traditional logistic equation identify the strength of each attribute. However, they do not provide measures of slope or elasticity because of the nonlinear form of the model. More importantly, the logistic regression approach shows the breakdown of how the individual attributes will affect each of the outcome values, making it easy to compare and identify how input attributes can have different effects on each outcome.

Microsoft's logistic regression tools go even further in evaluating attributes for each outcome. The Neural Network Model Viewer is designed to identify attributes and rank their influence on side-by-side comparisons of outcomes.



**Figure 7.26**

Microsoft single row prediction. Click the Single Row icon and enter the values to forecast. In the bottom grid, choose the model and ModelType attribute. Select the Results icon to see the forecast value, which is Race in this example.

## Prediction

The traditional logistic regression can be used to estimate the probability that a person with a specified set of independent attributes will make a certain decision. In fact, the regression coefficients can be used to compute the probability of each of the outcomes occurring. Figure 7.25 shows the manual way to compute the probabilities based on the probability equations. For each model type, multiply the model's regression coefficients by the sample data and sum the results for that model to get  $XB$  (the sum of the  $x$ -value times the matching  $b$  coefficient). Compute  $\exp(XB)$  for each model type and add the values. The probability of the base event (Hybrid) is  $1/(1+\text{sum-}\exp)$ . The probabilities of the other events are found by multiplying the respective  $\exp(XB)$  times the base probability. In this example with relatively high income and high population, the model predicts a 43-percent chance of purchasing a road bike with a 31-percent chance of a full suspension mountain bike.

Microsoft BI has an automated option to predict the outcome based on values of the input attributes. If you want to examine many data points, you can place them in a new table and have the system predict all of the rows in one pass. For a handful of values, it is easier to use the single-row option and compute them interactively one at a time. Figure 7.26 shows the basic process. Once you pick the Singleton Query option, you simply plug in the values of the  $X$ -attributes in the table. The second step is a little trickier. Select the data source in the bottom grid (Model Type Logistic), and set the predicted field Model Type as the Field. To see the probability value, you need to select Prediction Function on the second row. Enter [Model Type] including the brackets in the Criteria column. The results button computes and displays the most likely outcome (highest probability) for the case. In the example, the result is similar to the traditional logistic regression, which had Mountain full as a secondary choice with a slightly lower probability.



Predicted	MTN full (actual)	Track (actual)	Tour (actual)	Mountain (actual)	Hybrid (actual)	Race (actual)	Road (actual)
MTN full	3326	0	483	1530	138	2696	2460
Track	0	0	0	0	0	0	0
Tour	0	0	0	0	0	0	0
Mountain	444	21	186	671	158	567	426
Hybrid	0	0	0	0	0	0	0
Race	111	1	14	83	11	78	57
Road	0	0	0	0	0	0	0

**Figure 7.27**

Mining Accuracy Classification Matrix. Values on the diagonal were the number of cases predicted correctly. The other values are the number of items incorrect for each prediction. For example, the model was wrong 444 times while predicting mountain bike purchases that were actually full suspension purchases.

Microsoft also provides a method to examine the potential accuracy of the model. As you plug in various values for the X-attributes, you will begin to see a pattern—the model almost always predicts Mountain full or mountain bikes. At that point, you should become a little suspicious of the model and investigate its accuracy.

The Microsoft Accuracy Chart has an option to display the **Classification Matrix**, which uses the holdout data to show exactly where the model predicts correctly and incorrectly. The results for the ModelType are shown in Figure 7.27. When building a model to predict outcomes, it is always useful to know where the model is likely to be right and what happens when it is wrong. The numbers on the diagonal show the number of items that were predicted correctly. For example, 3326 times the model correctly predicted the purchase of full suspension mountain bikes, but 1530 times it predicted a full suspension purchase that was actually a mountain bike purchase. Check the rows for Tour and Road bikes and you will see the real weakness of the model—it predicted zero purchases of those model types. It appears that the model needs another factor to identify when people will purchase tour and road bikes instead of mountain bikes.

## Naïve Bayes

**How do you begin an analysis when you know little about the data?** The **naïve Bayes** approach is a good tool for examining data when you have little background about the relationships. The method begins by assuming most of the data are unrelated (hence the name “naïve”), and it has proven to be relatively robust in finding initial relationships. It is not the most accurate or most powerful tool—it can miss details and it requires relatively broad categories of data. But, it is easy to implement, fast to run, and the results are relatively easy to understand. One important catch is that the Bayes method requires that the X-attributes and the Y-variable be discrete. Microsoft BI can convert continuous data to discrete groups, but the Y-attribute works best if it is legitimately a discrete set of values. Otherwise the results are highly dependent on how the discretization is handled.

Gender/Purchase	Yes	No	Total
Female	89	65	154
Male	172	139	311
total	261	204	465

**Figure 7.28**

Simple joint distribution. Observations on the number of people entering a store and whether or not a purchase was made. Goal is to find probability a purchase will be made if a new person enters the store.

In its simplest form, Bayes' Theorem is relatively easy to understand and use. Yet, Bayes' Theorem leads to an entirely different way to understand probability—and it provides a useful approach to machine learning. The theorem is a simple mathematical relationship of conditional probabilities. A **conditional probability** defines the probability of an event (B) occurring, given that another event (A) has already happened. It is written  $P(B | A)$  and pronounced “the probability of B given A.”

## Goals

The primary goal of the naïve Bayes data mining tool is to identify which attributes have a strong effect on the outcome variable. The approach uses an advanced version of Bayes' Theorem, where the system begins with a relatively neutral probability and uses data observations to estimate an updated posterior probability distribution function.

### *Simple Version*

Before trying to understand that last sentence, it is best to begin with a simple description of Bayes' Theorem using a single binary attribute. Figure 7.28 shows a **contingency table** or collection data on a group of customers based on two attributes. The fictional customers were observed to be either male or female and the outcome consists of whether or not the person purchased an item during that visit. The outcome variable (purchase) is binary and the x-attribute (gender) is also binary. The table represents a complete tabulation of the joint distribution because it covers all  $2 \times 2$  ( $= 4$ ) possible cases. The table also shows the margin totals because they are important to Bayes' Theorem. Each cell represents the combination of both attributes. For example, 89 people were observed to be Female **and** make a purchase. In terms of frequency, the  $P(\text{Gender}=\text{female and Purchase}=\text{yes}) = 89/465$ . To simplify notation, this probability will be written as  $P(F \cap Y)$ .

The margin totals have their own interpretation—they are used to compute the conditional probabilities. To understand the term, answer the question: If you observe a representative female shopper, what is the probability that person makes a purchase? In notation the probability is written:  $P(Y | F)$ . The answer is readily available from the table. Once you are given the gender, simply cover up or ignore the row of Male data. It is clear that 89/154 of these women made a purchase, which is the conditional probability. Similarly, it is easy to compute the  $P(M | N)$ —the probability the shopper was a man given that no purchase was made. Look at the information provided (No purchase) and cover up the rest, so the answer is 139/204.

The conditional probabilities work in any direction—the mathematics does not know which piece of information you are given first. So if  $P(Y | F)$  is  $89/154$ , what is the probability that it was a female shopper if you are told that a purchase was made  $P(F | Y)$ ? Because the table shows all of the possible combinations, again you look only at the data you are given (Purchase=yes) and find the answer directly as  $89/261$ . The genius of Bayes was to realize that sometimes data is received in sequence and you do not have all of the parts—it is still possible to compute the conditional probability. First, the simple statement of Bayes' Theorem:

$$P(Y | F) = \frac{P(F | Y)P(Y)}{P(F)}$$

You want to predict the probability that female shopper will make a purchase, but all you have from past observations is that if someone did make a purchase, you know the number of times (frequency) that the shopper was female which is  $P(F|Y)$ . You also know the conditional probability  $P(F | N)$ —the probability the customer was female given no purchase was made. Additionally you know the probability that any shopper will make a purchase. From these values you can compute  $P(F)$  and  $P(Y | F)$ . Note that the denominator is always the sum of all possible numerators in the equation. In this case:

$$P(F) = P(F|Y)P(Y) + P(F|N)P(N)$$

The formula is easy to apply from the data in the table:

$$P(Y|F) = \frac{\left(\frac{89}{261}\right)\left(\frac{261}{465}\right)}{\left(\frac{89}{261}\right)\left(\frac{261}{465}\right) + \left(\frac{65}{204}\right)\left(\frac{204}{465}\right)} = \frac{\left(\frac{89}{465}\right)}{\left(\frac{89}{465}\right)\left(\frac{65}{465}\right)} = \frac{89}{154}$$

It is clear that this approach gives the same answer as that found directly from the table. Bayes' Theorem is relatively easy to prove using the definition of conditional probability:

$$P(A|B) = P(A \cap B) / P(B)$$

Simply write the corresponding definition of  $P(B|A)$  and combine the two equations.

### *Bayes' Theorem for Updating Probabilities*

The bigger question is why does Bayes' Theorem matter? In the sample problem, it was easier to read the answer directly from the table instead of trying to remember the formula and plug in the values to compute the conditional probability. The answer to this question explains the value of the Bayesian data mining method. The answer consists of two factors: (1) It is rarely possible to obtain all of the data needed to generate the complete tabulation, and (2) Bayes' theorem has a critical interpretation for applying new information and learning.

First consider the issues of size. The sample problem had the binary outcome and a single binary attribute, leading to a  $2 \times 2$  table of values. What happens when another binary attribute is added? Essentially, the table would have to expand into a third dimension, multiplying the number of cells by 2 again, for a total of  $2^3 = 8$  cells. In general, with  $n$  binary attributes, the complete joint distribution would contain  $2^{n+1}$  cells. With even 9 binary attributes, the problem contains 1024 cells. If the attributes are not binary and contain additional attributes, the number of

cells can be substantially larger. For example, if each attribute has 4 options, the number of cells in a joint probability distribution with 9 attributes is  $(2^2)^{10}$ , which is slightly over one million cells. You should begin to see why dimension reduction becomes important.

The number of cells is just the beginning. Estimation of the distribution requires observations within each of those cells, and a sufficient number of observations to hold down the bias. Even the small binary problem with a tiny 10 observations per cell would require at least 10,000 evenly distributed data points. A more realistic problem quickly jumps to 10-100 million. Because these levels of data are rare, Bayes' Theorem becomes more important and the data mining tools need special assumptions to compute probabilities.

The real importance of Bayes' Theorem is found by slightly restating it and focusing on the interpretation of the elements:

$$P(Y|F) = P(Y) \frac{P(F|Y)}{P(F)}$$

In this version,  $P(Y)$  is the **prior distribution** (or *a priori* in Latin).  $P(Y|F)$  is the **posterior distribution** (*a posteriori*). The prior distribution is simply the initial guess for the solution—it is typically assumed to be a neutral distribution. It could be a uniform distribution or perhaps an early measure of data. The known conditional distribution  $P(F|Y)$  is the likelihood measure obtained from data observations. It is normalized by the  $P(F)$  value which is also observed in the data. Consequently, Bayes' Theorem provides a way to take a prior belief about the probability and modify it with observed information to obtain a new, better belief about the probability. In Bayesian terms, probability is subjective and decision makers use new information to revise their subjective probabilities using this formula. As documented by Zellner (1971), this approach can be used to define statistical theory. However, it also illustrates the foundations of data mining and machine learning: Begin with a general estimate of probabilities, use observation on conditional data to compute a refined conditional probability that can be used for prediction.

### Estimation

The naïve Bayes approach is a relatively simple method to examine the effect of discrete X-attributes on a discrete predictable variable. Its primary objective is to estimate the probability of each Y-value outcome given each of the independent X-attribute values.

Figure 7.29 illustrates the goal with the results of a Bayesian model. Each value of the Gender attribute is matched to each value of the ModelType outcome attribute. Each number is an estimated probability that a person with the specific attribute will purchase the listed model type. For instance, there is a 61.5 percent probability a man will purchase a Track bicycle. The process and the results chart become more complex when the problem has multiple attributes—particularly when the attributes have several values. Also, note that the results are rough, so remember that the goal is to provide an initial idea of potential attributes.

The probabilities in Figure 7.29 are the conditional probabilities from Bayes' Theorem (e.g.,  $P(\text{Gender}=\text{Male} \mid \text{ModelType}=\text{Track})$ ). Keep in mind that these probabilities are only a small portion of the overall results. The problem ultimately has more attributes (such as city population and income). Adding more attributes

Model/Gender	Male	Female	Missing
All	0.584	0.352	0.064
Track	0.615	0.346	0.038
Race	0.591	0.331	0.078
Tour	0.570	0.360	0.069
Road	0.579	0.348	0.073
MTN full	0.577	0.355	0.068
Hybrid	0.611	0.387	0.002
Mountain	0.591	0.371	0.038

**Figure 7.29**

Bayes objective. Compute the probability each attribute value (Gender) leads to a specific outcome (Model Type). In this partial table, 58.4 percent of the customers are Male. There is a 61.5 percent chance a man will buy a Track bike.

dramatically increases the size and complexity of the problem. One of the first steps to reduce the problem is to assume that the various x-attributes are independent. For instance, gender might affect the model choice and income might affect the model choice, but gender is not related to income and there is no interaction effect on the choice of model. This simplifying assumption leads to the “naïve” name. But, it means that the joint probabilities can be estimated with simple multiplications. From probability theory, if two events A and B are independent:

$$P(A \cap B) = P(A)P(B).$$

Consequently, Bayes Theorem for multiple X-attributes can be written as

$$P(Y|X_1 \dots X_n) = \frac{P(Y) \prod_i^n p(X_i | Y)}{P(X_1 \dots X_n)}$$

Much like the values in the simple tabular example, the conditional probabilities in the numerator can be estimated from frequency counts within each cell. However, it is difficult to obtain enough data to fully fit even the naïve independence model. Cells with few observations can bias the results—particularly with a large number of cells. Consequently, most tools estimate a probability density function instead. Assume that each probability arises from some underlying, relatively neutral distribution. Use the mean and variance to estimate the parameters of the distribution. Covariances are not needed because of the independence assumption. Often the data is smoothed to reduce problems with limited data. Different tools use different smoothing and estimation techniques, so final results can vary by tool.

Despite the naïve independence assumption, Bayesian classification has been shown to perform reasonably well in many cases. The actual probability numbers can be off, but the classification or relative importance of the x-attributes is generally good—particularly with datasets that contain noise or complex relationships. In the end, the naïve Bayes tools produce a set of conditional probability estimates shown in Figure 7.29—for each attribute.

```
SELECT dbo.Bicycle.ModelType, dbo.Bicycle.SalePrice,
       dbo.Bicycle.SerialNumber, dbo.Bicycle.FrameSize,
       YEAR(dbo.Bicycle.OrderDate) AS SaleYear,
       dbo.Bicycle.LetterStyleID, dbo.Bicycle.StoreID,
       dbo.Bicycle.EmployeeID, dbo.City.Population2000,
       dbo.City.Income2004, dbo.Customer.Gender
FROM   dbo.Bicycle
INNER JOIN dbo.Customer
        ON dbo.Bicycle.CustomerID = dbo.Customer.CustomerID
INNER JOIN dbo.City
        ON dbo.Customer.CityID = dbo.City.CityID
```

### Figure 7.30

Data query. Combine Bicycle, Customer, and City tables. Compute SaleYear, and retrieve at least ModelType, Gender, Population, and Income attributes. SerialNumber is needed as the key column.

## Data

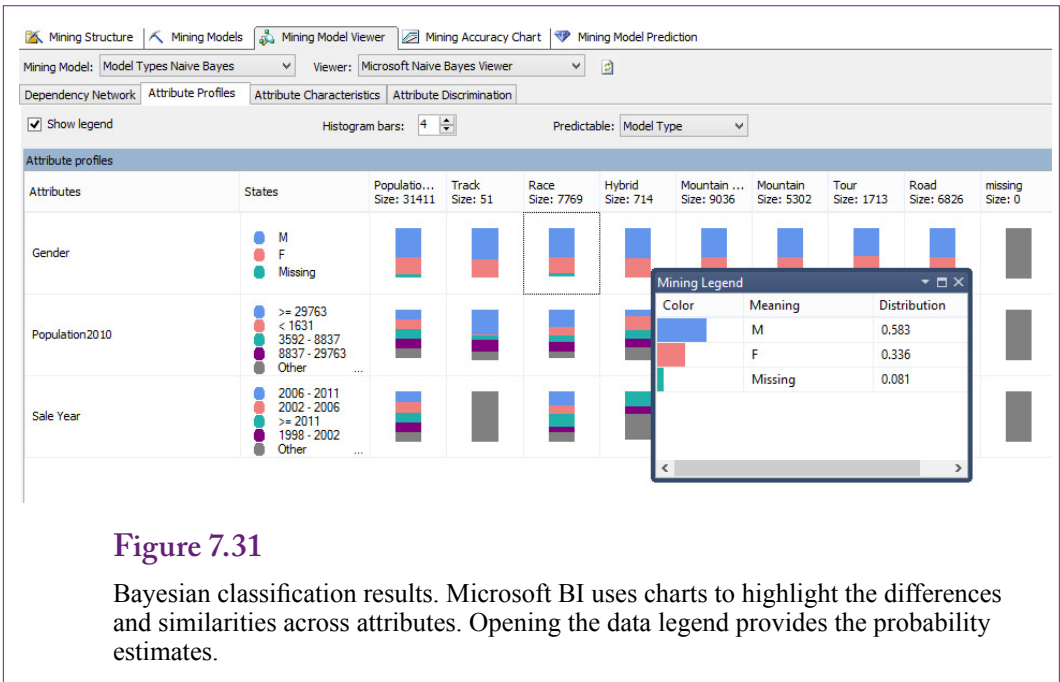
Data for the naïve Bayes classification must be in discrete categories. In particular, the outcome values should be a discrete list. It is possible to use tools to create categories for continuous data for X-attributes. It does not make much sense to use a continuous variable as the predictable (Y) attribute—because the results would be highly dependent on the categories. It could be possible to use a continuous Y-attribute if a clustering algorithm was able to find strong groupings.

To illustrate the technique, assume Rolling Thunder Bicycle Company managers want to determine which types of people buy each model type. The company does not have much personal information about customers—it knows their gender, the year, and the model type purchased. All of these are discrete values. With some research through the Census Bureau, the company can find the average income of people living in the same cities as each customer, along with the size of the city. These last two attributes are continuous, but Microsoft BI can automatically define discrete groupings of these values. If predefined categories exist (perhaps for wealthy, typical, and low-income), the categories could be defined manually. If necessary, create a new data source view that contains the Bicycle, Customer, and City tables. Create a named query that links the tables and selects the desired columns. As shown in Figure 7.30, it should include a SaleYear column as Year(OrderDate).

## Tools

For tiny problems, you might be able to use subtotals from a cube browser to compute all of the joint probabilities needed to fill a small table. From there, it would be possible to compute the Bayesian conditional probabilities directly. However, specialized naïve Bayes tools are almost always used to estimate the probabilities—both because they are efficient at handling large amounts of data and because they automatically solve complications such as smoothing.

Microsoft's naïve Bayes model is relatively easy to create and run. Add a new Mining Structure and choose Microsoft Naïve Bayes. Pick the appropriate data source view and select the named query you created as the Case table. Choose the ModelType as the predictable column, verify that SerialNumber is set as the key



column, and add the Gender, Income2009, Population2010, and SaleYear columns as input attributes. You will be given the option to set the data type for each column. Income and Population should be Discretized because they are continuous variables. SaleYear should be a Discrete variable. For some problems, year can become an issue—if a company has been collecting data for dozens of years, it will probably be necessary to group them into categories to reduce the complexity of the problem. Once the model is built, right-click its name in the Solution Explorer and choose the Option to Process and then Run the model.

## Results

Figure 7.31 shows the results of browsing the Attribute Profiles from the Bayesian classification. It would be difficult to show all of the probability numbers for every attribute value on one page. Consequently, Microsoft BI uses small charts to show the probabilities. The charts make it easier to look for similarities and differences across attributes. You can open the data legend for a specific attribute to see the probability details. Glancing at the charts, it appears that Gender and Income might have strong effects on the choice of models—the probabilities are high and they appear to vary across model types. Population might be less important, but it appears to be different for Track and Race bikes.

## Attribute Evaluation

The Bayesian tool provides more detailed evaluations of attributes within the results under the Attribute Characteristics tab. In Figure 7.32, notice the importance of the Gender attribute values, followed by the effect of city size (population). In particular, it appears that people from larger cities have a higher probability of purchasing the full suspension mountain bike. Use the drop-down list to select different model types (outcome attributes). Also, rolling the mouse cursor over one of the bars pops up the actual probability value.



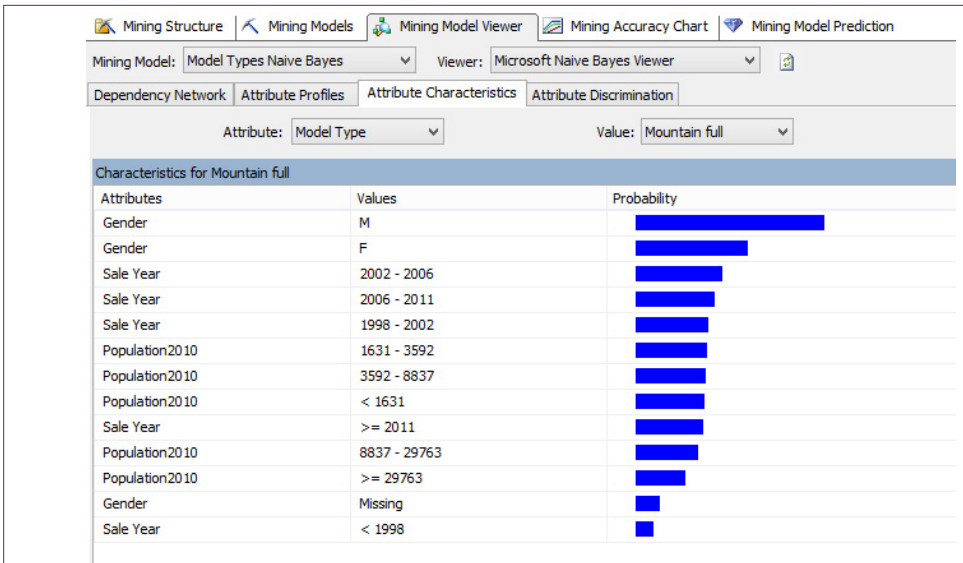
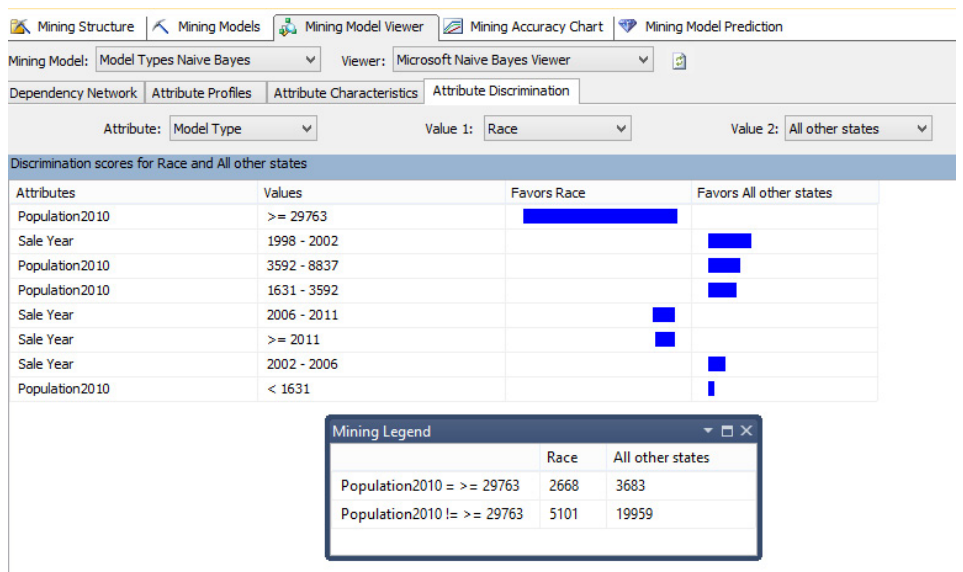


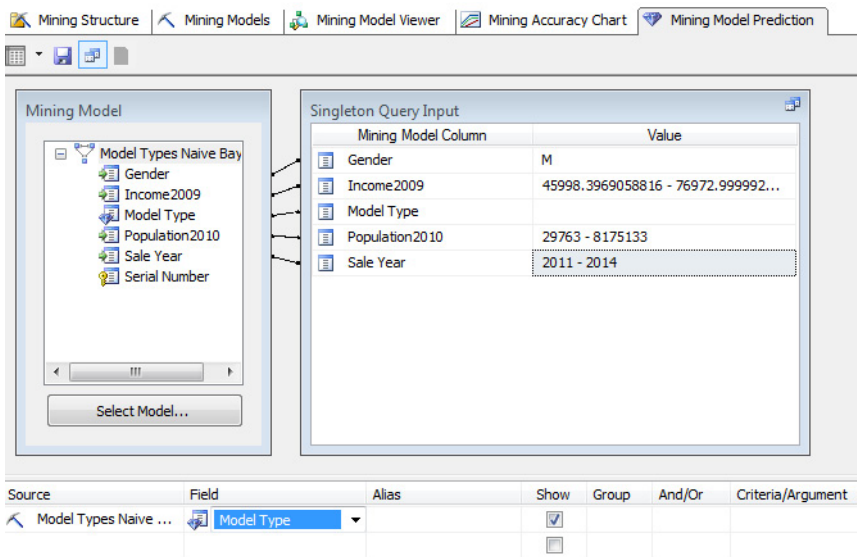
Figure 7.32

Attribute evaluation. Details for each model type value provide more information about the conditional probabilities. The attributes at the top of the list have a stronger effect on the outcome.

Figure 7.33

Attribute discrimination. The ability of attributes to discriminate among the outcomes is important in analyzing the data. This chart enables you to focus on one value (e.g., Race) and compare it to all other values or to a second specific value.





**Figure 7.34**

Microsoft single row prediction. Click the Single Row icon and enter the values to forecast. In the bottom grid, choose the model and ModelType attribute. Select the Results icon to see the forecast value, which is Race in this example.

Microsoft BI provides another chart to help evaluate attributes. Click the Attribute Discrimination tab to compare each model type to other model types—both as a group or individually. Figure 7.33 shows a discrimination chart for the Race model type versus all others. The drop-down boxes can be used to compare to another specific model, or to change the main value. Attribute discrimination is important when analyzing the data. It represents the ability of the model to differentiate among the selections. For example, an analysis of attribute importance might reveal that gender and population are important for every model type. An evaluation of discrimination ability might find that sale year or a specific population group is critical to explaining which group buys mountain bikes instead of road bikes. If a model has low discrimination ability, the analysis will lack detail. It might be able to predict increases or decreases in total sales, but it will not have the ability to specify exactly which model types will be sold. In this example, large cities are clearly key to predicting sales of race bikes. Examine the other model types and see if large cities play the same role. You will find that it is strongest for the race bikes, which is a useful piece of information when managers begin devising a marketing strategy.

## Prediction

The naïve Bayes model can be used for prediction. Microsoft BI provides the same prediction tool it uses for other models. You can create a separate table of X-attribute values that you want to predict, or you can enter data for a single set of values and interactively get a prediction. However, you should keep in mind that the Bayesian probability estimates are typically relatively rough. The category

Predicted	MTN full (actual)	Track (actual)	Tour (actual)	Mountain (actual)	Hybrid (actual)	Race (actual)	Road (actual)
MTN full	2959	0	386	1384	102	1667	1982
Track	0	0	0	0	0	0	0
Tour	0	0	0	0	0	0	0
Mountain	151	5	145	438	118	237	297
Hybrid	0	0	0	0	0	0	0
Race	837	16	155	394	62	1430	614
Road	0	0	0	0	0	0	0

**Figure 7.35**

Mining Accuracy Classification Matrix. Values on the diagonal were the number of cases predicted correctly. The other values are the number of items incorrect for each prediction. For example, the model was wrong 233 times while predicting mountain bike purchases that were actually full suspension purchases.

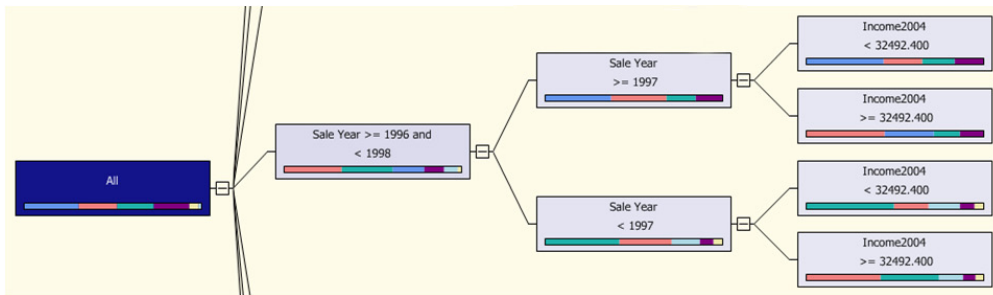
classifications tend to be good, but the actual probability numbers can be wild—particularly when the independence assumption is violated. See Rish (2001) for a study of the performance of the naïve Bayes classifier.

Figure 7.34 shows the process for generating a simple forecast. Note that because all of the attributes are discretized, you must choose from among the values; you cannot enter random numbers. Keep this constraint in mind if you choose to build a data table with values to be predicted. The predicted outcome of these values is the Race model type. Recall that large-city population is a strong discriminator for this model, so virtually any input values using the large-city choice are going to result in the Race outcome. Yet, the data show that the company sells road and mountain bikes to large cities as well, so the forecast is a little biased.

As shown in Figure 7.35, the classification matrix is a useful way to examine the model forecasts. The row represents the forecast item and the column is the actual number. For example, the Bayesian model correctly predicted the sale of 2959 full suspension mountain bikes, but 386 times when it predicted mountain full, the sale was actually tour bikes and 1384 times it was wrong when it predicted full suspension sales that were actually purchases of hard tail mountain bikes. More importantly, notice that the model predicts zero sales of Track, Tour, Hybrid, and Road bikes. The first three might be valid because those models were sometimes discontinued; however, zero sales for road bikes is a big problem. Clearly, the model lacks the ability to differentiate road bikes from other model types—particularly full-suspension mountain bikes.

## Decision Trees

**Is there a way to organize the attributes to see how they explain the decision?** A **decision tree** is a tree or graph that models how attribute values influence a categorical outcome variable. Each node classifies a break point in an attribute that has a different effect on the outcome variable. Trees begin with a single starting node and follow a path to new nodes. Splits are made based on the values of an attribute. If the attribute values have significantly different effects on the outcome variable, a separate node and path are created. Decision trees



**Figure 7.36**

Partial decision tree. The nodes represent split points where an attribute has a significantly different effect on the outcome variable (Model Type). Following a path from the top (left) node represents a distinct classification of the impact of data on choosing a model type.

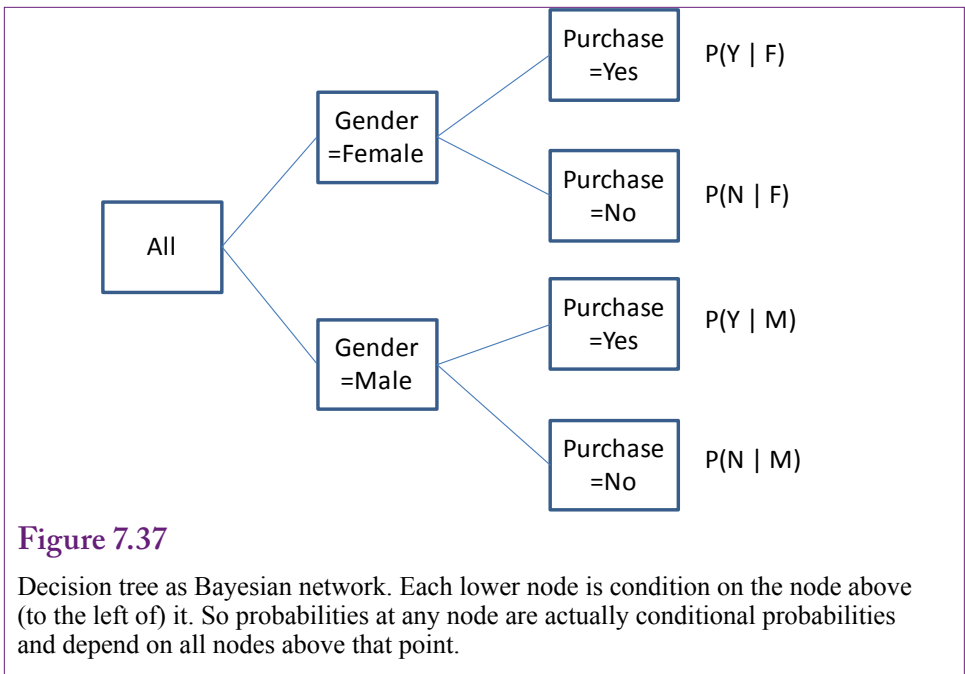
can be descriptive, but in data mining, the ultimate goal is to discover a tree that best predicts the outcome variable. Each node split leads to a different outcome.

Figure 7.36 shows a portion of a decision tree. Following a path from the starting (left) node yields a classifier for how the attribute values affect the model type outcome variable. For example, follow the top-most path in the figure to see that (1) SaleYear between before 1996, (2) SaleYear before 1995, and (3) Gender not missing (null) leads to a specific purchase decision. The combination of model types unique to that path is shown by the small, color-coded bar chart on the node. Based on results presented in a later section, that particular path has a 29 percent probability of purchasing a race bike (the largest, orange bar). Simplifying that particular path: Before 1995 (that is, 1994) customers with an identified gender (M/F) 29 percent chance of purchasing a race bike; followed by a 26 percent chance of purchasing a mountain bike. Because the value of SaleYear is set to a single value, this particular node is not very useful for predicting what will happen next year. However, other nodes cover different years, and if managers have additional or subjective information about those years, the conclusion can still be useful. Decision trees are probably the most popular data mining tool because the results are easy to understand and the model is relatively easy to create. Microsoft BI selects decision trees as the default method and many of the other routines use at least portions of the method.

## Goals

The decision tree model is one of the important attribute classifier tools available for data mining. It requires minimal supervision by the analyst—simply choose the Y-attribute outcome and select a handful of X-attributes that might influence the outcome variable. The outcome variable should hold categorical values. The input attributes can be discrete or continuous. The primary goal of the method is to determine subgroups of the attributes that have different effects on the outcome variable. The model created can be used to predict the outcome. Simply use the classifications to identify the most appropriate group based on the input attributes and the probability of the outcome is provided.

Decision trees are built iteratively. From any node, the system searches for correlations between input attributes and the output variable. When an important dif-



ference is identified, the node is split into new paths. The number of paths emanating from a node depends on how the data affects the output variable. Various algorithms have been defined to create trees with differences largely in how correlation is measured and how the gain from splitting a node is defined.

A decision tree is sometimes considered to be a Bayesian network, because the lower nodes on the tree are conditional on the prior nodes. Figure 7.37 illustrates the basic principle. Any probability at a node depends on the nodes that came before it. The assumption of independence of attributes essentially means that the nodes do not cross, leaving a relatively simple tree.

Microsoft exploits these relationships and the Microsoft decision tree algorithm uses a Bayesian approach by default. Microsoft BI offers two variants of the Bayesian approach—it can use a diffuse prior which assumes no bias and all options are equally likely, or the parameters can follow a Dirichlet distribution. Microsoft explains the Dirichlet approach and other statistical details of the algorithm in a white paper (Heckerman, Geiger, and Chickering 1995).

Decision trees are typically built up one node at a time. At each point, the system uses the underlying data to determine if splitting a node beyond that point will improve the network's ability to explain the results. The data consists primarily of the counts of the observations within each category, which is applied to estimating the underlying probability density functions. In a Bayesian context, the search systems typically use a **maximum likelihood estimator (MLE)** to see if a new variation will improve the likelihood measure. The MLE measure is typically computed as the log of the probability function.

A slightly different approach to choosing whether to split a tree is to use **Shannon's entropy** or information measure. Shannon was a scientist who developed several important theories regarding the transmission of data. These theories have been useful in evaluating the information content of data, and provide a method to

x	p(x)	-p log(p)	p2(x)	-p log(p)
1	0.25	0.150515	0.7	0.108431
2	0.25	0.150515	0.05	0.065051
3	0.25	0.150515	0.2	0.139794
4	0.25	0.150515	0.05	0.065051
		0.60206		0.378328

**Figure 7.38**

Example of Shannon entropy. The first distribution is diffuse, evenly distributed data. The entropy is high with little information. The distribution is boring. The second example is more interesting where some values have higher probabilities. The entropy is lower, the information is higher.

compare models. The basic information definition is straightforward. Given a set of events (x) defined by their probabilities (p), the information content is:

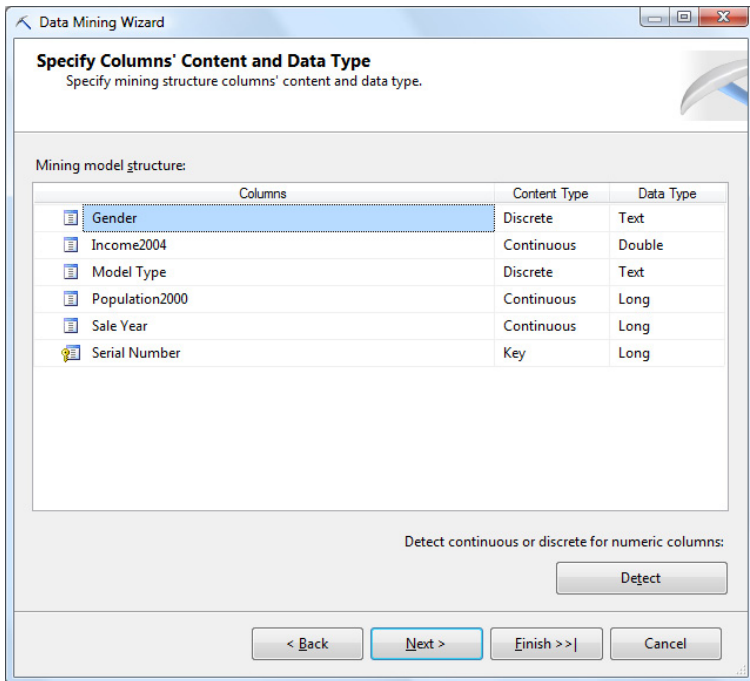
$$H(X) = -\sum_i p_i \log(p_i)$$

Many decision tree systems use this information measure to decide how to split nodes. Microsoft BI offers it as an option. To understand the measure, Figure 7.38 shows two versions of a distribution. The first is a rectangular diffuse distribution where each point has the same probability. Plotted, it is a flat line. Its entropy is relatively high and there is little information in the data. The second distribution is more interesting because some values have substantially higher probabilities. The entropy measure is much lower so the information content is higher. The measure is easy to compute and the change in its value from one model to the next provides a good measure of the information gained.

## Data

Traditionally, the data for a decision tree consists of discrete values; however, continuous values can be used for the X-attributes. It is possible to use a continuous variable for the predictable variable, but then the model becomes a regression model tree. This is the method Microsoft uses to solve the basic linear regression problem. So, this section considers the dependent variable to hold categorical data. With Microsoft BI, attributes with too many discrete values are often automatically compressed to fewer categories using feature selection. A large number of attributes with multiple values can present problems in terms of the size of the estimation and in terms of over fitting the model. Microsoft BI automatically checks the data and reduces the number of dimensions if needed.

To compare this approach with the others, you can use the same data source. The objective is to build a decision tree model that can predict the sale of the various model types. The basic attributes to be tested consist of Gender, SaleYear, and income and population taken from the City table. If the data source does not yet exist, create a new one and add the Bicycle, City, and Customer tables. Create a new named query that joins those three tables and uses SQL to select the main attributes: SerialNumber, ModelType, Gender, Population2010, and Income2009. Create SaleYear as a new column using the expression Year(OrderDate).



**Figure 7.39**

Microsoft decision tree setup. SaleYear is seen as continuous. Leaving it that way enables the tool to find ranges of values instead of focusing on single years.

## Tools

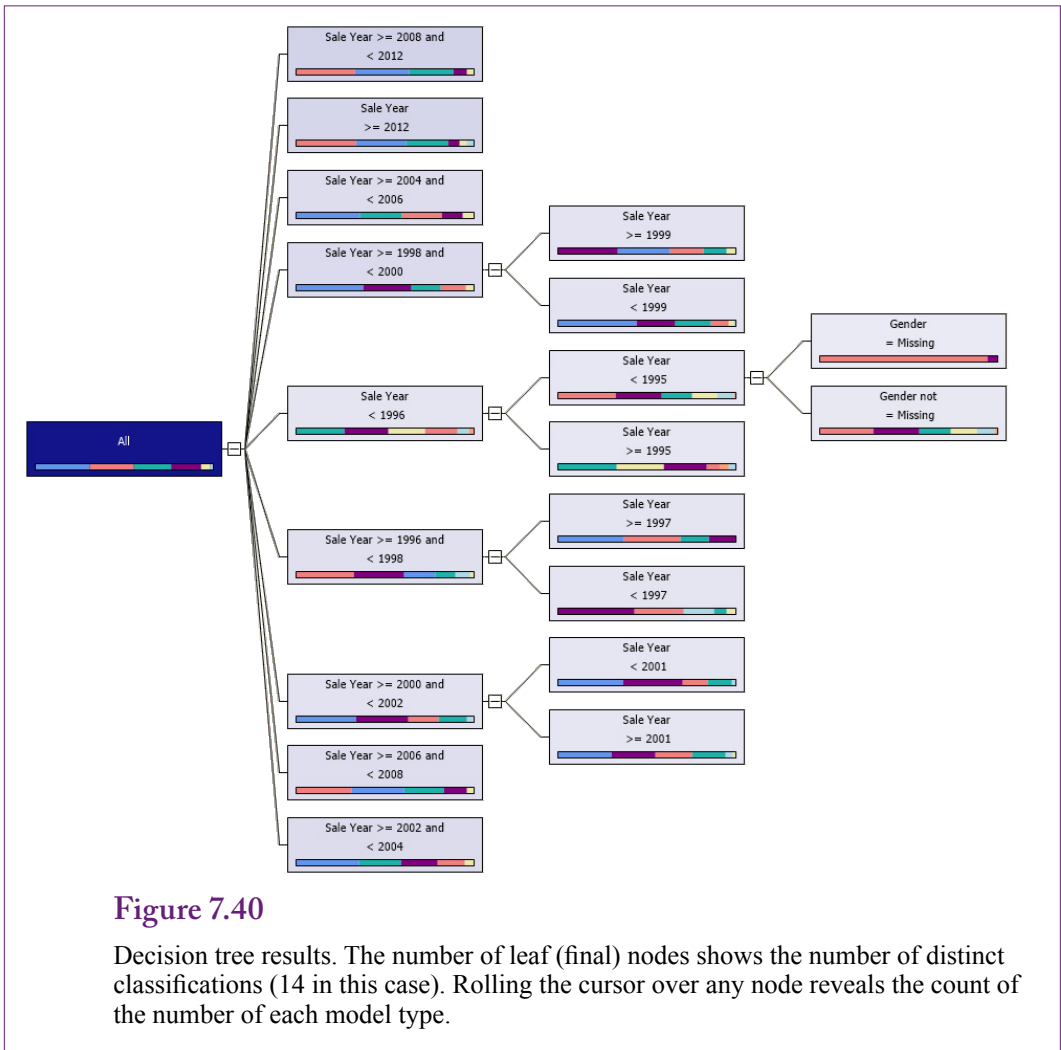
The Microsoft decision tree tool is relatively easy to configure. Once the data has been established, add a new Mining Structure and choose the decision tree tool. Select the data source view that contains the named query holding the data. Pick that query as the Case table. Choose the columns and assign them to the appropriate categories. SerialNumber is the unique key column because the analysis is focused at the bicycle level. ModelType is the predictable variable. The other columns are simply input attributes. The wizard shows you the data types. Notice in Figure 7.39 that SaleYear is evaluated as continuous data, even though it holds integer values. With a smaller number of years, it might be useful to leave this attribute as a discrete type. However, to reduce the number of dimensions, it is more flexible to treat it as a continuous variable. This approach will also enable the tool to identify important ranges of years instead of examining one year at a time.

When the model has been defined, it is straightforward to select the model, and Process it to deploy it to the analysis server and Run it to obtain the base results. Once the model is processed, it can be Browsed to see the results.

## Results

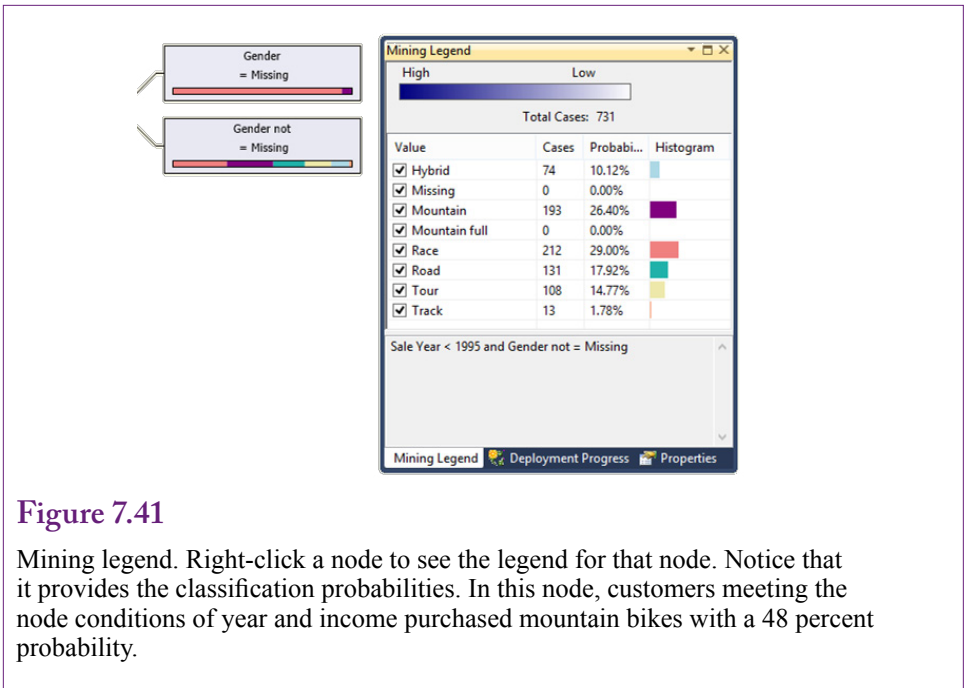
Figure 7.40 shows the results presented as the entire tree. One problem with decision trees using many attributes is that the trees can become large and complex. They are difficult to view on paper. The Tree Viewer contains options not shown in the figure to zoom in, or collapse the tree to a limited number of levels. The





bar chart on each node indicates the distribution of model types within that node. Rolling the mouse cursor over a node will show the actual count by model type for that node in a popup box. If you checked the box to Enable Drill-Through as one of the final steps in the wizard, you can right-click any node and choose the Drill Through option to see a table of the underlying data that matches the conditions of the chosen node.

To get more details about any node, right-click a node and choose the option to Show Legend. As shown in Figure 7.41, the legend provides the number of cases in the node for each model type and the estimate of the percentage of items that would choose each model type. In this example, the customers were likely to purchase a race with a 29 percent probability. Examining the legends for various end nodes should help you see the conditions that lead to each of the outcomes. If managers have additional subjective information about the nodes, such as marketing campaigns or weather, it can help explain how customers behave.



**Figure 7.41**

Mining legend. Right-click a node to see the legend for that node. Notice that it provides the classification probabilities. In this node, customers meeting the node conditions of year and income purchased mountain bikes with a 48 percent probability.

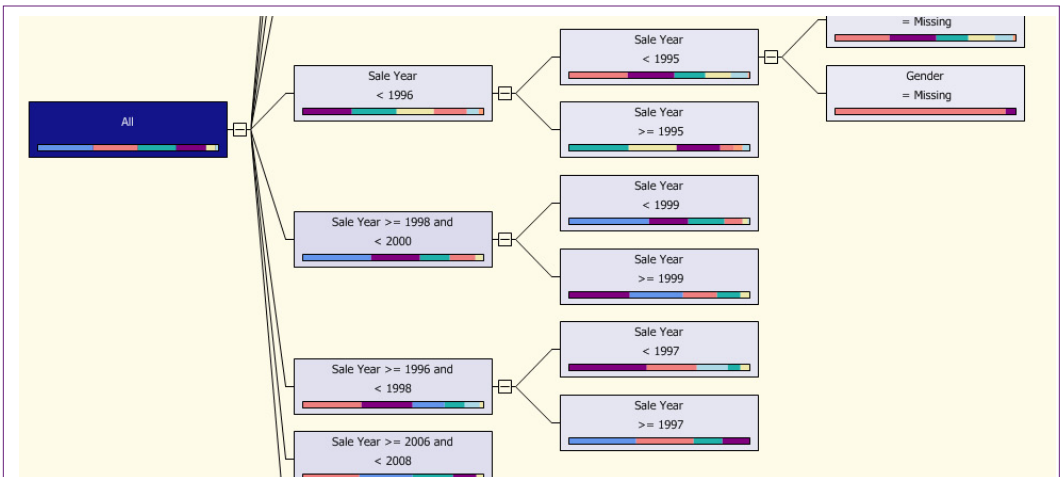
## Attribute Evaluation

The main purpose of the decision tree is to find classifications that describe how combinations of the attributes affect the outcome variable. The decision tree itself provides a visual way to see patterns and how the attributes combine to affect the outcome. When examining the tree, you should look for similarities and differences in the node bar charts. It would be nice if the viewer made it easy to sort and reorganize the tree to group similar items together. As it is, you get to use those skills honed as a child searching two similar comics to find the seven differences between images.

You might want to focus on one specific model type at a time. For instance, find the nodes that predict a relatively high number of full suspension mountain bikes (medium-blue bar), then look for similarities in the attributes. Most of the differences appear to be driven by the year. Starting from the left-most node, the first division appears to be sales between 2004 and 2006. Full suspension bikes appear to be in the lead for those years. After 2006, the sales are split and depend mostly on year but somewhat on gender. In the later years, race bikes appear to have become more popular. When looking through the decision tree, remember that each node is dependent on its parent nodes—the conditions apply in a chain.

## Prediction

In many ways, decision trees are automatically built to enable prediction. The easy way is to start at the top (left) of the tree and apply the if-then classification conditions to the desired data. Follow the path that matches the data to an end node. Check the probabilities within that node to get the most-likely outcome as well as the probabilities of the other outcomes. Figure 7.42 shows how to apply the sample data to obtain a prediction. The first level is set by the sale year. The applicable node (Sale Year  $\geq$  2006) has only one split into two children based on



**Figure 7.42**

Prediction by following the tree nodes. X-attributes are year=1997, income=60,000, population=80,000, and gender=male. The path is short and determined only by year. The full suspension model at 36 percent is the most likely outcome.

income. The node (Income  $\geq 32,492$ ) matches the sample data and it is the end node. Notice that the gender and population attributes do not affect the classification or prediction. Examining the legend details for the ending node reveals that Race is the leading outcome with a 39 percent probability, followed by mountain full at 27 percent, and road at 19 percent probability. This manual path through the tree is easy to use and easy to explain to others. It is available with all tools that generate decision trees.

Microsoft BI also provides a prediction method where data rows to be predicted can be entered into a special table. The method predicts all rows in one batch and is useful when several outcomes need to be predicted and compared. The table results can be analyzed separately or exported to other tools. Figure 7.43 shows the basic setup for entering the X-attribute values interactively. Switching to the Result View reveals the prediction of the Mountain full model type. However, it does not provide the probability details or information about other model types. Consequently, for decision trees, this tool is more useful for batches of data. For single rows, it is usually better to simply trace through the tree.

Microsoft BI provides several common tools to evaluate the accuracy of the model including the Lift Chart which compares the predicted values to a perfect model. It can also be used to compare variations of models to each other. As shown in Figure 7.44, the Classification Matrix is a tool that compares individual forecasts to the actual results. Simply glancing at the matrix reveals that in at least one area it does a better job than some of the other tools. It predicts some sales for Road bikes, while some of the other tools always predict zero.

Decision trees often have a problem with over fitting. The models tend to adapt too much to the sample data used. If the data are somewhat limited or do not accurately match the population, the over fitting can cause serious prediction errors. Some tools use techniques to reduce the problem. For example, the tools can automatically divide the data into smaller groups and fit decision trees to each group.

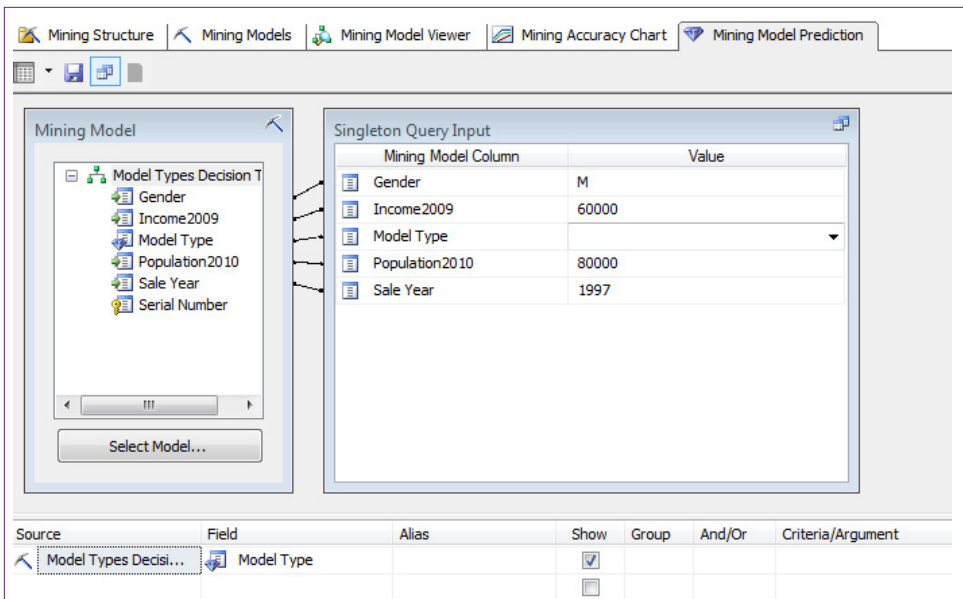


Figure 7.43

Prediction of decision tree. Be sure to set the model in the drop-down list at the bottom and choose the Model Type outcome variable. Switch to Result View to see that the system predicts the most likely outcome as Mountain full.

Figure 7.44

Mining Accuracy Classification Matrix. Values on the diagonal were the number of cases predicted correctly. The other values are the number of items incorrect for each prediction. For example, the model was wrong 339 times while predicting race bike purchases that were actually full suspension purchases.

Predicted	MTN full (actual)	Track (actual)	Tour (actual)	Mountain (actual)	Hybrid (actual)	Race (actual)	Road (actual)
MTN full	<b>1901</b>	0	212	1029	53	1074	1147
Track	0	<b>0</b>	0	0	0	0	0
Tour	0	0	<b>0</b>	0	0	0	0
Mountain	239	0	76	<b>530</b>	110	374	166
Hybrid	0	0	0	0	<b>0</b>	0	0
Race	1773	2	339	577	135	<b>1849</b>	1494
Road	0	19	93	83	25	37	<b>124</b>

Error	Pass	Fail	Log Like.	Lift	RMSE
1	2092	4189	-1.4654	0.1198	0.6676
2	2097	4185	-1.4662	0.1189	0.6677
3	2097	4186	-1.4664	0.1190	0.6674
4	2095	4187	-1.4658	0.1202	0.6674
5	2092	4191	-1.4693	0.1167	0.6676
<b>Avg</b>	2094.6	4187.6	-1.4666	0.1189	0.6675

**Figure 7.45**

Cross validation summary. Five different error measures were computed on five folds of data. The Pass/Fail numbers represent the number of cases predicted correctly and incorrectly (out of about 6282). Because the errors are relatively consistent, the model is reasonably robust and not over fit.

The final tree is an average of results from each group. If the amount of data is limited, some systems use **bootstrapping** to generate additional data points that match the characteristics of the sample with some degree of randomness. The Microsoft decision tree algorithm does not support bootstrapping.

Use the Cross Validation tool to check for over fitting. The tool divides the data into sets (typically choose 5 or 10 folds). It then fits a decision tree to all of the data less one of the sets. It repeats this process until each set has been held out. The cross validation tool computes error measures for each application. If the error measures vary radically across the sets, then the model is not very robust and is susceptible to over fitting.

Figure 7.45 summarizes the results of the cross validation tool applied to 5 folds or partitions of data. The pass/fail measures are counts of the number of predictions correct or incorrect. Each fold has about 6282 observations. The log likelihood value is derived from the probability density function. Likelihood is the probability of each probability arising and is computed as the product of all of the probabilities. Lift is the gain in predictive value of the model compared to random selection. Microsoft's technical notes state that it is measured from the log of the actual probability. RMSE for a categorical variable is slightly different from that used for continuous data. It is the square root of the mean of the squares of the complement of the probability scores (logs). The details of the definitions are not critical. The key is that each of the measures should be consistent across the various folds. If the results revealed significant differences, it would indicate that the model fitting is highly dependent on specific data points and would be less useful in predicting values on any other set of data. In this case, all of the measures are highly consistent, so over fitting does not appear to be a problem.

## Neural Network

**How can the modeling process be automated even more and handle nonlinear relationships?** At several points in the discussion and development of computers, people have asked questions comparing computers to humans. Computers are amazingly faster and more precise than humans at computations and retrieving specific items from memory. Yet, people are incredibly fast at tasks involving pattern recognition and retrieving associated items from

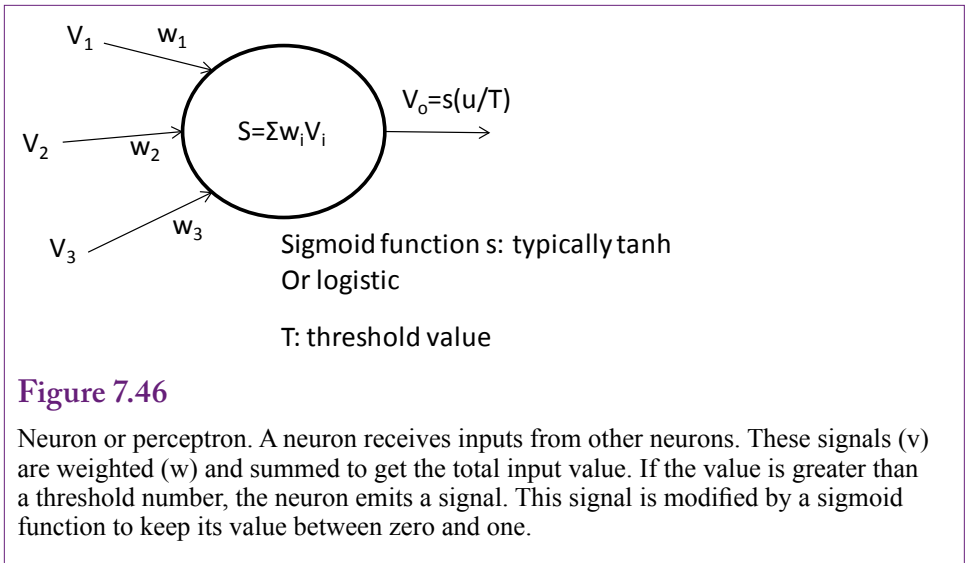
memory. These discussions led to the question of how people think, and what it would take to make computers think the same way. After a few wrong turns and many years, one of the most important answers to these questions led to the development of neural networks. Check out Rumelhart and McClelland (1986) for details on the early development concepts.

The name **neural network** comes from the description of the way the human brain is built: as a large interconnection of neurons. Think of a neuron as a single cell that has inputs and an output that emits an electrical signal. When total input values reach some level, the neuron “fires” and emits its own output signal. The network consists of connections among many of the cells. The network holds patterns and images. Fortunately, you do not need to become a specialist on human brains to use the tools. In fact, the tools probably depart significantly from human anatomy, because the tools have been adapted to computer processing and to solve computational problems.

From a data mining perspective, the best way to think about neural networks is in terms of the output or goals. Essentially, a neural network defines a relationship between the input X-attributes and the output Y-attribute. Conceptually, this relationship is similar to a regression relationship where the system minimizes errors. However, neural networks have the ability to define highly non-linear relationships. The non-linearity makes it possible to estimate complex interactions among the attributes, but it also makes it difficult to understand and analyze these relationships. In many ways, the result can become a black box, where the system might produce accurate predictions, but the workings of the relationships are essentially hidden. This tradeoff becomes an important decision when you use the tool. Are you willing to give up the ability to understand the relationships to gain an improvement in prediction? If you cannot completely understand the relationships, will the predictions be valid as input data changes? The answers depend on the details of the problem being studied. You can certainly run a neural network on almost any problem, and sometimes the predictive ability is high and the model is easy to understand. But, as models become more complex, you should always take a step back and ask yourself if the model is going to work for the specific situation you are studying.

## Goals

The ultimate goal is to define a relationship between the input attributes and the output variable. The primary purpose is to create a model with high predictive accuracy. The technique for achieving this goal appears a little unusual when you first encounter it, and you need to understand some of the terminology to work with the results. As shown in Figure 7.46, neurons or perceptrons are the foundation of the method. Think of a neuron as a smart light bulb. It receives input energy ( $v$ ) from other cells. Each input is weighted by some value ( $w$ ) and the result is totaled. If the input energy exceeds a certain threshold ( $T$ ), this neuron lights up and emits a signal. Technically, the signal emitted is modified by a sigmoid (s-shaped) function to ensure that the values range from zero to one. Microsoft BI uses the most common sigmoid functions: hyperbolic tangent ( $\tanh$ ) for the hidden layer, and a logistic function for the output layer. The immediate goal of this tool is to find the values of the weights ( $w$ ) that provide the best fitting model.



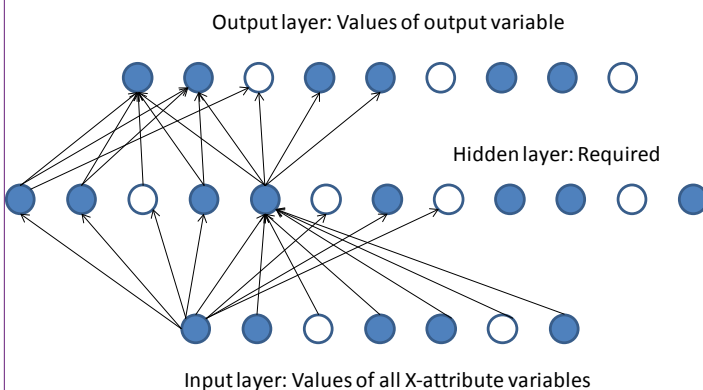
**Figure 7.46**

Neuron or perceptron. A neuron receives inputs from other neurons. These signals ( $v$ ) are weighted ( $w$ ) and summed to get the total input value. If the value is greater than a threshold number, the neuron emits a signal. This signal is modified by a sigmoid function to keep its value between zero and one.

Neurons exist in groups because the patterns are stored within the network. A key feature of neural networks is that they need three layers of neurons: Input, Output, and Hidden. As indicated in Figure 7.47, the neurons in the hidden layer are connected to every neuron in the input layer and every neuron in the output layer. In a data mining application, each input neuron represents one of the  $X$ -attributes. If the attribute is categorical, separate input neurons are created for each categorical value. The input values are normalized to keep them within reasonable ranges. For example, continuous values are converted to a version of a  $Z$ -score ( $[\text{value} - \text{mean}] / \text{standard deviation}$ ). An output neuron is created for each

**Figure 7.47**

Neural network layers. The input layer consists of values from the  $X$ -attributes. The output layer contains the outcomes to be predicted. The hidden layer is critical and provides the main flexibility in the relationships. Only a few connections are shown here. Typically, every hidden neuron is connected to every input and every output neuron.





possible outcome of the predictable variable. If there are too many outcome variables, the system typically splits into a second network. With Microsoft BI, 500 is the definition of “too many.” In the Rolling Thunder Bicycles example, each output neuron represents one model type. The hidden layer is a buffer that is crucial to provide the flexibility to model complex problems.

The neural network method uses the sample data to train the network—essentially finding the weights and threshold values that lead to the best predictions. Along the way, the system has to determine the number of neurons in the hidden layer. A larger number improves the predictive ability but increases the complexity of the model. If too many neurons are added, the model becomes over fitted and the model will work only with the sample data. Microsoft BI provides you some control over the number of neurons in the hidden layer through parameters. The `HIDDEN_NODE_RATIO` parameter defaults to 4.0 and provides an initial estimate of the number of neurons by multiplying by the square root of the number of input cells times the number of output cells.

## Data

Neural networks can analyze almost any type of data. The method is general and has been used successfully for many complex problems, including predicting continuous and categorical variables. The technique is also commonly used for pattern-matching problems, such as text and speech recognition. The most important step is to identify the outcome variable. If the dependent (predictable) variable holds continuous data, Microsoft BI converts it to discrete bins.

The Rolling Thunder Bicycle company case is easy to configure as a neural network problem. Again, the goal is to predict selection of model type based on the limited customer attributes available. The easiest approach is to create a data source view that contains the three main tables: Bicycle, Customer, and City. Then build a named query that includes at least the `SerialNumber`, `ModelType`, `Gender`, `SaleYear` computed from `OrderDate`, `Income`, and `Population` columns. This query is the same as the one used in the other sections of this chapter.

## Tools

Configuring Microsoft BI to estimate the neural network is straightforward. Add a new Mining Structure and choose the Neural Network method. Pick the data source view that contains the data. Set the named query as the main case table. For choosing columns, be sure that `SerialNumber` is set as the key because the data is organized by bicycle. Set `ModelType` as the predictable column, then select `Gender`, `Income2004`, `Population2000`, and `SaleYear` as input columns. Technically, it is possible to select a column as both predictable and input, but choosing that approach makes the results difficult to understand and use for prediction.

## Results

Because of their underlying nature, results from a neural network can be difficult to comprehend. What does a weight on a hidden neuron really mean? Predicting outcomes is usually reasonable, and Microsoft BI provides a useful tool for predicting outcomes based on various input data. It is more difficult to understand relationships and the impact of input variables on the outcomes. Some tools convert the internal weights into a nonlinear equation from the input attributes to the outcome variable. Although the equations are often complex, they can aid in understanding the relationships. Microsoft BI does not attempt to provide equations.

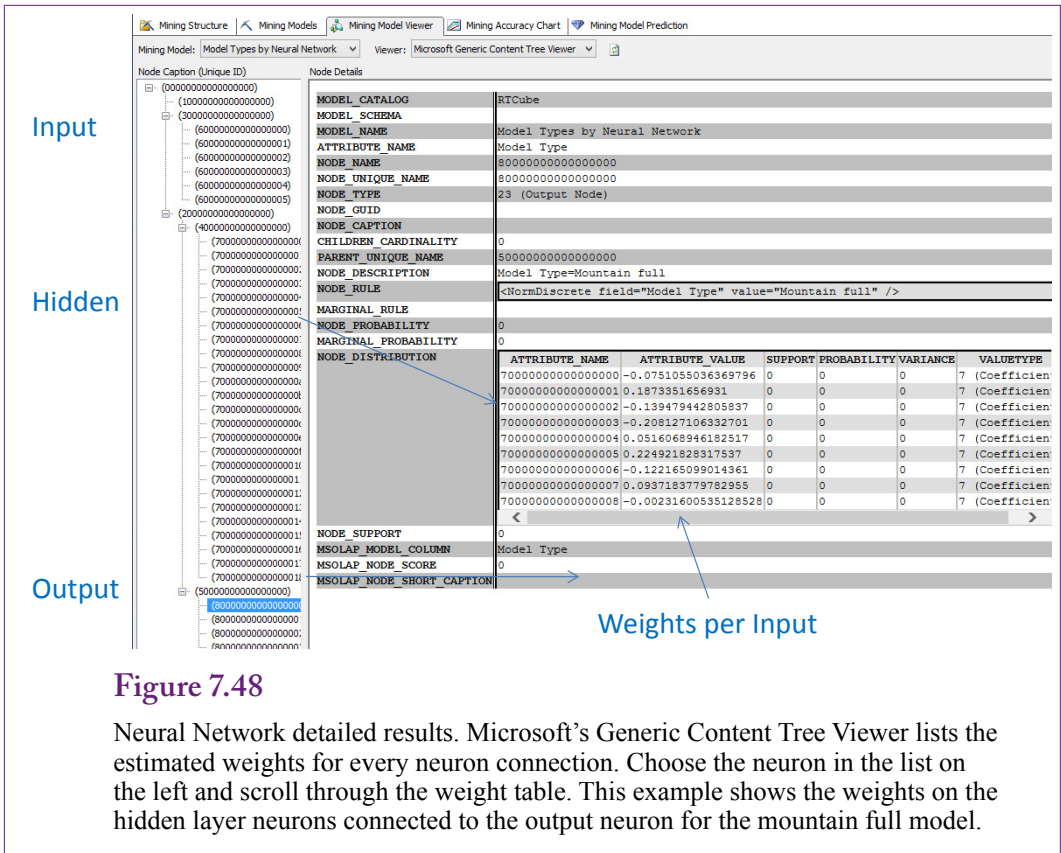


Figure 7.48

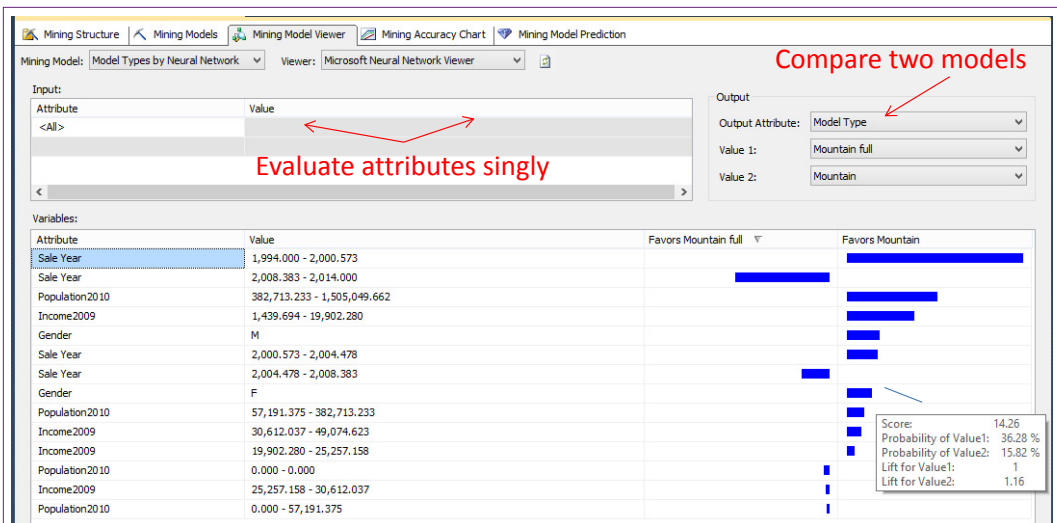
Neural Network detailed results. Microsoft's Generic Content Tree Viewer lists the estimated weights for every neuron connection. Choose the neuron in the list on the left and scroll through the weight table. This example shows the weights on the hidden layer neurons connected to the output neuron for the mountain full model.

Instead it provides the values of the weights estimated for each neuron. However, Microsoft does provide a tool to compare outcomes side-by-side.

Figure 7.48 shows some of the detail weights for the output neuron for the mountain full model type. This view is available in the Mining Model Viewer by selecting the Generic Content Tree Viewer and expanding the nodes in the tree list. Remember that these weights come from the 19 neurons in the hidden layer. By themselves, the numbers are difficult to understand. By hand, it would be possible, but difficult, to track the input weights through all hidden-layer neurons to the weights on the outcome variable. Microsoft BI provides a second viewer to help explore the importance of the various attributes.

## Attribute Evaluation

Microsoft BI provides the Neural Network Viewer as the primary tool to explore the strength of various input attributes. The most important attribute values are listed at the top, so exploring the tables for various combinations can provide insight into the strength of each attribute. You can select any pairs of outcomes. The combination shown in Figure 7.49 should be useful to managers wishing to decide if customers are going to switch more to full suspension bikes instead of hard tail mountain bikes. Clearly, year is important—because of when full suspension bikes were introduced. An income effect shows that customers from cities with lower per capita incomes favor hard-tail bikes—which makes sense because suspension bikes are more expensive. A small population effect indicates that people from



**Figure 7.49**

Neural Network attribute exploration. This viewer shows the side-by-side comparison of two output values to help judge the impact of various input attributes. The analyst can choose any two attributes for comparison. Rolling over a bar in the chart provides the details in the popup box. It is also possible to specify a single attribute filter and see the impact of the remaining attributes.

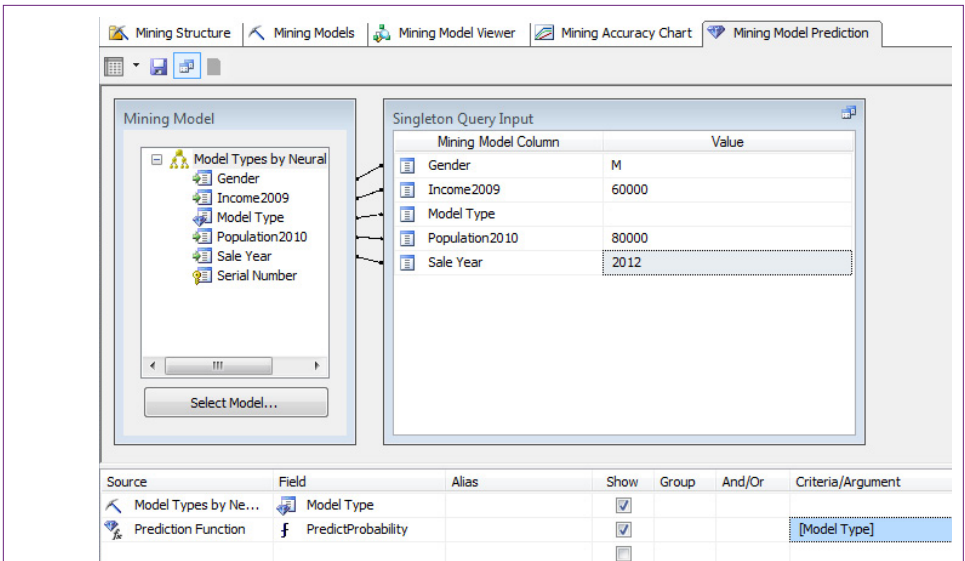
larger cities also favor hard tails. To examine the situation of higher-income cities in more detail, change the <All> attribute in the top-left table to Income2009 and select the highest income category in the Value column. You will see that although other attributes influence the strength of the relationship, every case favors simple mountain bikes instead of full suspension, except for years 2008 – 2014.

## Prediction

Most tools provide a method to predict the outcome value for a given set of input attributes. The process basically feeds the attributes to the network input neurons and reads the probabilities from the output neurons. It is possible to perform the calculations manually using DMX queries to retrieve the estimated coefficients. However, in most cases, it is easier to use the Microsoft Prediction tool.

Figure 7.50 shows the basic process for the Microsoft prediction tool. For predicting many different input combinations, it is easier to create a special table and enter the input values into that table. Then run the prediction to return the results to that table. SQL and other exploratory tools can be used to examine the results. However, you can right-click and use the singleton query to enter one set of attributes at a time. Switching to the Results view reveals that these attributes predict the customer will purchase a race bike. The Prediction Function (PredictProbability) will supply the probability (35.9%).

The Mining Accuracy Classification Matrix provides more details on the accuracy of the predictions. Figure 7.51 shows the results for the Rolling Thunder Bicycles model type predictions. The biggest issue is the prediction of zero sales of Road bikes. The predictions for Mountain full, Mountain, and Race are not great either, but at least the highest number of cases for each type is correct. Because



**Figure 7.50**

Prediction with neural network. Choose the singleton query to enter X-attribute data interactively. Switching to the results view reveals the answer of Race with a probability of 32.8 percent.

**Figure 7.51**

Mining Accuracy Classification Matrix. Values on the diagonal were the number of cases predicted correctly. The other values are the number of items incorrect for each prediction. For example, the model was wrong when predicting full suspension sales (row) of 1337 that were actually simple mountain bike purchases (column).

Predicted	MTN full (actual)	Track (actual)	Tour (actual)	Mountain (actual)	Hybrid (actual)	Race (actual)	Road (actual)
MTN full	3015	0	470	1337	123	2523	2237
Track	0	0	0	0	0	0	0
Tour	0	0	0	0	0	0	0
Mountain	554	23	206	708	149	570	525
Hybrid	0	0	0	0	0	0	0
Race	189	1	43	189	12	295	201
Road	0	0	0	1	0	1	1

Predicted		MTN full (actual)	Track (actual)	Tour (actual)	Mountain (actual)	Hybrid (actual)	Race (actual)	Road (actual)
<b>Full</b>	Logistic	<b>3326</b>	0	483	1530	138	2696	2460
	Bayes	<b>2959</b>	0	386	1384	102	1667	1982
	Tree	<b>1901</b>	0	212	1029	53	1074	1147
	Neural	<b>3015</b>	0	470	1337	123	2523	2237
<b>Track</b>	Logistic	0	<b>0</b>	0	0	0	0	0
	Bayes	0	<b>0</b>	0	0	0	0	0
	Tree	0	<b>0</b>	0	0	0	0	0
	Neural	0	<b>0</b>	0	0	0	0	0
<b>Tour</b>	Logistic	0	0	<b>0</b>	0	0	0	0
	Bayes	0	0	<b>0</b>	0	0	0	0
	Tree	0	0	<b>0</b>	0	0	0	0
	Neural	0	0	<b>0</b>	0	0	0	0
<b>MTN</b>	Logistic	444	21	186	<b>671</b>	158	567	426
	Bayes	233	6	144	<b>438</b>	118	237	297
	Tree	239	0	76	<b>530</b>	110	374	166
	Neural	554	23	206	<b>708</b>	149	570	525
<b>Hybrid</b>	Logistic	0	0	0	0	<b>0</b>	0	0
	Bayes	0	0	0	0	<b>0</b>	0	0
	Tree	0	0	0	0	<b>0</b>	0	0
	Neural	0	0	0	0	<b>0</b>	0	0
<b>Race</b>	Logistic	111	1	14	83	11	<b>78</b>	57
	Bayes	837	16	155	394	62	<b>1430</b>	614
	Tree	1773	2	339	577	135	<b>1849</b>	1494
	Neural	189	1	43	189	12	<b>295</b>	201
<b>Road</b>	Logistic	0	0	0	0	0	0	<b>0</b>
	Bayes	0	0	0	0	0	0	<b>0</b>
	Tree	0	19	93	83	25	37	<b>124</b>
	Neural	0	0	0	1	1	1	<b>1</b>

**Figure 7.52**

Comparison of accuracy by method. Notice that only the Decision Tree method predicts any sales of Road bikes, and the Ree correctly predicts more Race bike sales than the other methods.

of the flexibility of the neural network, it is unlikely that any other method will do substantially better. In other words, the model is likely to be improved only by collecting data on additional attributes.

As always, it is a good idea to run cross validation on the model to ensure that the model is not over fit and too dependent on specific observations. The results are not shown here, but the error rates are strongly consistent across the partitions, so over fitting is unlikely to be a problem.

## Model Comparisons

**How do the model results compare to each other?** The ultimate question is whether one modeling approach is better than the others at identifying attributes and making predictions. In general, there is no single answer. However, if the outcome variable holds continuous data, linear regression has several nice properties; it is easy to use; it is easy to forecast; and the results are easy to explain.

Method	# Correct	Percent	RMSE
Logistic	1915.6	30.5	0.647
Bayes	2175.2	34.6	0.657
Tree	2094.6	33.4	0.668
Neural	1928.8	30.7	0.622

**Figure 7.53**

Comparison of model predictions. Notice that the Naïve Bayes approach does a better job overall than the other methods on this data.

## Prediction

It is easiest to compare models in terms of predictions. Several measures including RMSE and lift are used to evaluate models. It is important that the same items are being measured in each model. Be even more cautious if different software tools are used to compute the measures because they might be defined and computed differently in each tool. Microsoft's prediction matrix is another useful tool when comparing forecasts. Figure 7.52 summarizes the data for the Model Type forecasts created with the four methods for discrete data covered in this chapter. Notice the similarities of the forecasts, except that the decision tree is the only one to predict any sales of Road bikes, and the Decision Tree approach correctly predicts Race bikes much better than the other methods.

Figure 7.53 summarizes the models by examining the number of correct predictions. These predictions were generated from the holdout data sets in the Microsoft BI tool. Notice that the Naïve Bayes and Decision Tree approaches have clearly predicted model types better than the other methods. This result might not hold for other data sets. Notice that overall, none of the models does a great job of predicting the choice of model type. But, the data is somewhat limited. More and better input attributes would be useful—but they can be difficult to obtain.

Another way to examine the predictions is to look at the forecast of each model type for each method. Managers of Rolling Thunder Bicycles probably need to

**Figure 7.54**

Comparison of model predictions by Model Type. Notice the prediction problems with Race and Road types and the over prediction of full suspension sales.

	Logistic	Bayes	Decision Tree	Neural Net	Actual Sales
MTN full	79.0	63.0	40.2	72.6	28.41
Track	0.0	0.0	0.0	0.0	0.0
Tour	0.0	0.0	0.0	0.0	4.62
Mountain	18.4	10.9	11.1	20.5	7.32
Hybrid	0.0	0.0	0.0	0.0	4.58
Race	2.6	26.1	45.8	6.9	31.02
Road	0.0	0.0	2.8	0.0	24.05

<b>Logistic</b>	Sale Year	Gender	Population	Income
<b>Bayes</b>	Gender	Sale Year	Population	
<b>Decision Tree</b>	Sale Year	Gender		
<b>Neural Net</b>	Sale Year	Population	Income	Gender

**Figure 7.55**

Summary of attributes by method. The most important attribute is listed first. This list was derived by examining the comparison between Mountain full and Mountain model types.

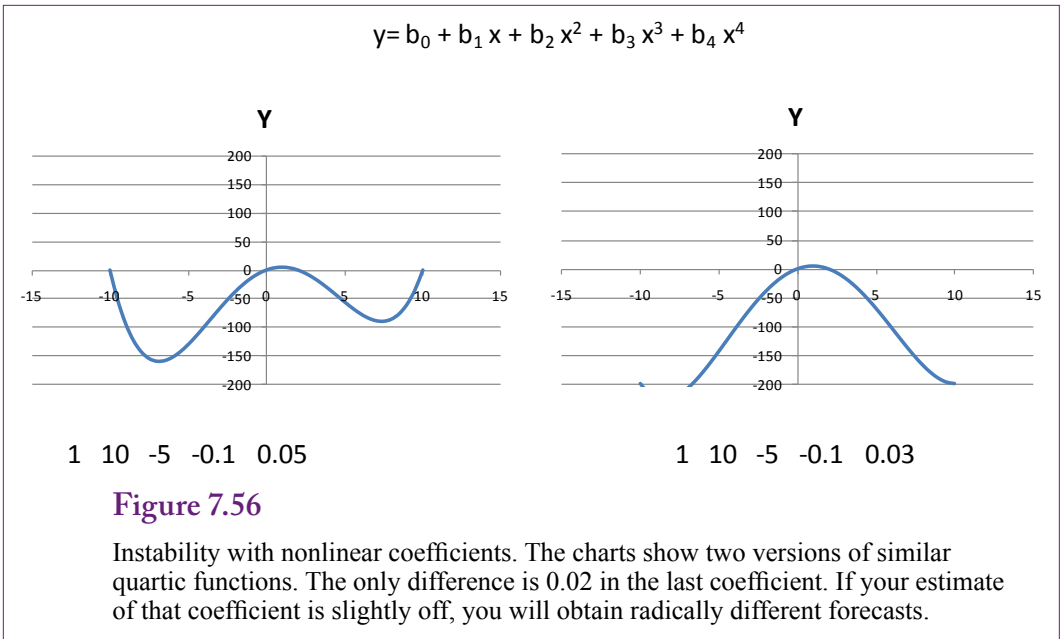
know which models will be popular next year. It is one of the reasons for undertaking this analysis. Knowing which models will sell the most will make it easier to order the correct number of components at the start of the year. To obtain the best forecast, you should first estimate all of the background variables (income, population, gender, and sale year); then enter those values into the prediction equations for each model. However, for an initial rough pass, Figure 7.54 shows the average values taken from the classification matrix data which uses all years. The actual sales by model type were computed using a simple SQL query for 2012. The purpose of this table is to look at the prediction with less detail. If managers do not care about income, population, gender, and year; are some of the models more accurate than others? The answer is challenging. All of them are over predicting the sales of Mountain full bikes. Logistic and Neural network both over predict the sales of Mountain bikes at the expense of Race bikes. All of the models are missing the sales of Road bikes. The Decision Tree predicts the most number of race bikes, but basically over-predicts those as well.

The basic problem with all of the predictions is that the underlying model needs more attributes and better data. The methods are doing as well as possible, but ultimately more data needs to be collected for additional attributes to improve the model. Alternatively, it is possible that the underlying consumer decision of which model to purchase has a large random element. Some decisions and events simply cannot be predicted with much accuracy. In this particular case, more data on individual customers would be nice but it is not available. However, there are other ways to get a better prediction model. Looking at the average sales for 2012 seems to provide better estimates than anything else, which implies that a better model could be built by examining sales over time. These techniques are covered in the chapter on time series. If you assume that sales for next year will be roughly based on sales for the current year, the forecasts should improve dramatically. But the point of this chapter was to examine how specific attributes affect the choice of model types, not to provide the most accurate forecast possible.

### Attribute Evaluation

With Microsoft BI, each of the estimation tools has a slightly different method of presenting the effect of the X-attributes. These differences provide different perspectives on the data and examining all of them should help the analyst see a bigger picture of the data. However, the differences make it more difficult to compare the methods. If all of the methods returned numerical coefficients on the attributes—similar to traditional logistic or linear regression, a formal comparison of the attributes is easy.





**Figure 7.56**

Instability with nonlinear coefficients. The charts show two versions of similar quartic functions. The only difference is 0.02 in the last coefficient. If your estimate of that coefficient is slightly off, you will obtain radically different forecasts.

Figure 7.55 summarizes the attribute evaluations of the four main discrete-attribute methods examined in this chapter. The most important attribute is listed first. For some methods, specific values of the attributes are more important than others—such as years after 1997 for Mountain full. The list was generated by finding the most important attributes when comparing Mountain full to the Mountain model type. In most of the methods, these rankings are relatively consistent across model types, with some variation in the lower level spots. However, Naïve Bayes is an exception. In the general model, Gender takes the top spot for influencing the choice of model type. This result is important to know because Naïve Bayes was better than the other methods for predicting sales of Race bicycles.

### Nonlinear Complications

Neural networks are powerful tools that can estimate complex relationships. However, nonlinearity can cause problems. To begin, nonlinear relationships are difficult to understand and hard to explain to others. A basic exponential or log function might be close enough to linear to work, but complex interactions among attributes quickly become meaningless.

More importantly, higher-order nonlinear equations can be unstable in the sense that large changes in the outcome variable arise from tiny changes in the coefficient estimates. Figure 7.56 illustrates the problem with a basic fourth-degree polynomial function. The two equations are different by only 0.02 in the last coefficient. Even tiny errors in estimating the function can lead to drastic changes in interpretation and prediction. And this chart uses only a single variable. The problems magnify when the attributes intermix through a nonlinear equation.

Forecasting is the other classic problem with nonlinear functions. Many start-up firms exhibit exponential growth for the first few quarters or even a couple of years. A nonlinear function does a good job of predicting the historical growth. But, exponential growth patterns are extraordinarily hard to maintain. At some point, the organization simply cannot maintain exponential change and it reverts

to a linear growth (or crashes completely). Consequently, even if a nonlinear function does a great job of predicting past growth, it is often safer to fit a conservative linear function for prediction. Ultimately, the regression, naïve Bayes, or decision tree forecast can be more accurate than a nonlinear forecast generated from a neural network. This statement does not mean you should never use a neural network—but you should be cautious when using the model for predicting beyond the immediate future.

## Summary

---

It is important and helpful to identify which attributes influence a predictable outcome variable. Knowing which attributes and which values influence others is a critical step in building models and predicting what will happen in the future. If the relationships are strong enough, you might be able to recognize causality. If the overall model is good enough (has a sufficiently small error), you will be able to predict what will happen in new situations. As long as you can measure the X-attributes, you can plug them into the estimated model and forecast a value or find the probabilities of any outcome occurring. For example, if you develop a model that relates personal attributes such as job, income, and education to loan payments; you can use the model to predict whether the next person applying for a loan will make payments on time. Almost any business model raises similar questions. The key is to identify the outcome variable and then find data on various attributes that might affect that variable. Once enough data is collected, the data mining tools will find and measure correlations and provide prediction estimates.

Linear regression is a powerful tool used in science to test hypotheses and evaluate claims. It is heavily used in economics and forecasting. In the context of data mining, it provides good measures of attribute coefficients, but the outcome variable must measure continuous data. Simple regression does not work if the dependent variable contains discrete categorical values. Logistic regression is one of the first tools developed to handle categorical dependent variables. The traditional method estimates linear equations for the effect on each outcome of the dependent variable. The Naïve Bayes tool uses Bayes' Theorem to use data observations to update a prior distribution and create a posterior distribution that more accurately describes the data. The probability distribution can identify the importance of each input attribute and can be used to predict the probability of any given outcome. Decision Trees are created to define a path through the attribute values. Each path results in a different outcome and highlights the values of the X-attribute data nodes that have a significantly different impact on the outcome. Decision tree results are good for understanding and explaining relationships.

Neural networks are probably the strangest evaluation method of the group. Loosely based on human brains, a neural network consists of three layers of neurons: Input, Hidden, and Output. Training the network consists of applying the sample data to estimate the weights of the connections between layers and to define the threshold firing value for a given neuron. These estimated weights define a nonlinear network that relates the input values to the outcomes of the predictable variable. The weights make it difficult to understand the contribution of each attribute, but the network can be used to predict the likely outcome of any new data.

All of the tools in this chapter are commonly used in data mining. Often, an analyst will apply all of the tools to the same data set—because each tool provides a slightly different perspective on the relationships. Comparing the results both in terms of prediction and the importance of each X-attribute provides insight and knowledge into the entire process.

## Key Words

---

bootstrapping	linear regression
causality	logistic regression
classification	maximum likelihood estimator (MLE)
classification matrix	mean absolute deviation (MAD)
conditional probability	model
contingency table	naïve Bayes
correlation	named query
decision tree	neural network
decision tree	posterior distribution
dependent	prediction
discretized data	prior distribution
elasticity	root mean square error (RMSE)
independent	Shannon's entropy
lift	

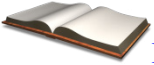
## Review Questions

---

1. Why is it useful to evaluate the effect of attributes on outcome variables?
2. How is missing data handled by each of the tools in this chapter: regression, logistic regression, naïve Bayes, decision trees, and neural networks?
3. How is data organized for all of the tools in this chapter?
4. What is the role of the key attribute in data sets for Microsoft BI tools?
5. What do the coefficients mean in linear regression results?
6. How is logistic regression different from linear regression?
7. What is the fundamental assumption of the naïve Bayes method?
8. What is the primary strength of the decision tree method?
9. How is the decision tree method different from the naïve Bayes approach?
10. How is a neural network result different from the other tools?
11. What is the problem of over fitting and how is it tested?
12. Should one of the methods in this chapter be preferred over the others?

## Exercises

---



### Book

1. Set up and run the linear regression example from the chapter, using both a standalone program and the data mining tool. Summarize the results and interpretation.
2. Set up and run the logistic regression example from the chapter. Summarize the results and interpretation.
3. Set up and run the naïve Bayes example from the chapter. Summarize the results and interpretation.
4. Set up and run the decision tree example from the chapter. Summarize the results and interpretation.
5. Set up and run the neural network example from the chapter. Summarize the results and interpretation.
6. For the main problem in the book with ModelType as the predictable variable, identify other attributes that would be good candidates for analysis. For items not in the original database, how would you obtain values?



### Rolling Thunder Database

7. Expand the analysis of total sales by city and see if you can identify attributes that reliably predict sales. What additional data might you want to collect?
8. Looking at the customers who purchased more than one bicycle from the Rolling Thunder Bicycles. What attributes do they possess—particularly which ones could be used to target a marketing campaign at other customers?
9. Are there attributes that affect whether customers buy bicycles through retail stores or directly from the company? Look at StoreID values of 1 and 2 versus the rest.
10. Examine the patterns of sales for carbon fiber versus aluminum and steel bikes. The bike can be classified by the type of material used in the down tube.
11. Managers want to increase prices and profit margins on Campagnolo (Campy) equipped bicycles. Who will be affected by this change? Or, what attributes lead people to purchase Campy-equipped bikes over Shimano?



## Diner

12. The managers want to know how to increase the sales of desserts (without changing the price). Who buys desserts now? Who does not?
13. What factors affect the total amount of a bill?



## Corner Med

14. What factors affect the total amount of revenue per visit? Hint: Consider at least the number of visits, procedures per visit, patient demographics, and the insurance company.
15. Are there factors that affect which physician treats a patient? For example, is it affected by diagnosis code or procedure?



## Basketball

Note: Every team is listed twice in the Games table. Load and create the view: TeamGameTotals which links to BaseTeam to eliminate the duplicates. Also, pick one season to answer each question.

16. What attributes affect whether a team wins a game?
17. What attributes affect the number of points scored in a game? In particular, how many points is the home court worth?
18. Are some divisions better or worse than others in terms of winning? What about in terms of total points scored?
19. Which players and player characteristics were key to wins by the LA Lakers?
20. Did importance factors change for the Lakers in the playoffs? For example, were some players more (or less) important in the playoffs than in the regular season?



### Bakery

21. Determine how the month and day-of-week (DOW) impact the sales of products by category.



### Cars

22. Do any of the attributes affect the price of the vehicles?
23. Which attributes affect the acceleration (SecTo60)?



### Teamwork

24. Using the basketball database, each person in the group should choose one team and determine which player statistics affect that team's ability to win.
25. Using the Rolling Thunder Bicycles database, assign one technique from this chapter to each team member and find the best model for predicting the selection of model type.

## Additional Reading

---

Heckerman, David, Geiger, Dan, and David M. Chickering, 1995, *Learning Bayesian Networks: The Combination of Knowledge and Statistical Data*, Microsoft Technical Report MSR-TR-94-09, Microsoft: Redmond. <http://research.microsoft.com/apps/pubs/default.aspx?id=65088>. [The technical description of Microsoft's Bayesian and Decision Tree methods.]

Intriligator, Michael D. 1978, *Econometric Models, Techniques, and Applications*, Prentice-Hall, Englewood Cliffs. [Basics of econometrics with good coverage of logistic and probit methodologies.]

Judge, George G., W.E. Griffiths, R. Carter Hill, Helmut Lütkepohl, and Tsoung-Chao Lee, 1985, *The Theory and Practice of Econometrics, second edition*, Wiley: New York. [A classic complete work on econometric theory for those who want to know how to handle problems that arise with regressions. Graduate level.]

Rish, Irina, 2001, "An Empirical Study of the Naïve Bayes Classifier," IBM Research Report RC 22230 (W011-014). <http://www.research.ibm.com/people/r/rish/papers/RC22230.pdf>. [An analysis of factors that affect the performance of the Bayes classifier.]

Rumelhart, David E. and James L. McClelland, 1986, *Parallel Distributed Processing, Vol. 1 and 2*, MIT Press: Cambridge, Massachusetts. [A classic collection of work on the foundations and start of neural networks. Includes descriptive and technical content.]

Trevor Hastie, Robert Tibshirani, and Jerome Friedman, 2009, *The Elements of Statistical Learning/2e*, Springer: New York. [An outstanding book on data mining, with an emphasis on theory. A graduated-level book that requires a strong mathematics background.]

Zellner, Arnold, 1971, *An Introduction to Bayesian Inference in Econometrics*, Wiley: New York. [A classic book on Bayesian theory, particular focus on subjective probabilities and how they can define traditional analyses. Graduate level with mathematics.]



---

# Time Series Analysis

## Chapter Outline

Introduction, 377	Microsoft Time Series Estimation, 406
Business Situation, 378	Goals, 406
Model, 379	Data, 408
Time Series Components, 379	Tools, 409
Auto Regression, 381	Results, 412
Moving Average, 384	ARTxp Model, 413
Trends, 387	Forecasts, 415
ARIMA, 389	Seasonality Evaluation, 417
Cross Correlations, 392	Cross Correlation and Linear Regression, 417
Evaluating Models, 395	Goals, 418
Data, 396	Data, 418
Attributes and Observations, 396	Tools, 419
Missing Data, 397	Comparison, 425
Traditional ARIMA Estimation, 397	Summary, 426
Goals, 398	Key Words, 427
Tools, 398	Review Questions, 428
Results, 400	Exercises, 428
Forecasts, 404	Additional Reading, 431
Seasonality Evaluation, 405	

## What You Will Learn in This Chapter

- How do you forecast what might happen tomorrow or in the next few months?
- Who needs to forecast data over time?
- What are the components of a time series and how can they be measured?
- What types of data can be used in time series analysis?
- What results does traditional ARIMA provide?
- How does the Microsoft Time Series estimation work?
- How can the effects of other series be incorporated into the prediction?

### Aunt Bessie's

Aunt Bessie's is a British food manufacturing company part of the William Jackson Food Group. It began in 1974 making millions of frozen Yorkshire puddings for Butlin's Holiday Camps [<http://www.auntbessies.co.uk/about>]. By 2011, the company produced 20 million Yorkshire puddings per week at one plant in Hull. In 2011, the company installed a new QD ERP tool tightly integrated with a Preactor APS (advanced planning and scheduling) system to help the company not just monitor production but also optimize schedules and handle capacity planning. Bhris Buckle, supply planning manager, notes that "the increased visibility from Preactor has also helped us to respond quicker, especially when we have a problem on a line. Beforehand, it could take a day for us even to notice, and then additional time to work out how best to react. Now we can see much more quickly when a problem is occurring and investigate different scenarios for dealing with it." [Tinham 2011] The forecasting capabilities helped reduce the amount of inventory cluttering up manufacturing facilities. They are also useful for integrating maintenance with operations. Maintenance can be scheduled for lower production points leading to fewer issues with disrupting critical production runs. The Preactor system models the production process and matches the desired production runs with the quantity of input materials needed. The planning system can search the schedules and plant capacity valuations to find optimal schedules [Preactor Web site]. The historical data feeds the long-run forecast and planning system to identify needed stock levels and resource scheduling.

Time series statistics are used to identify patterns and make forecasts. They can identify and handle random errors, seasonality, and cyclical changes, but might not catch major structural shifts.

Brian Tinham, "Sense and Sensitivity, *Works Management*, September 14, 2011. <http://www.worksmanagement.co.uk/information-technology/features/sense-and-sensitivity/36778/>

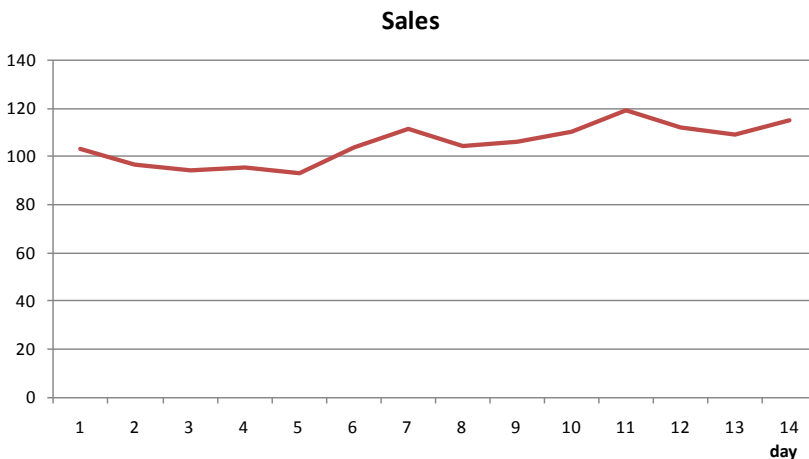
## Introduction

**How do you forecast what might happen tomorrow or in the next few months?** It is not really possible to see into the future. However, in many cases, it is relatively easy to predict that if the same thing happened for the last three days, it will probably happen again tomorrow. This concept is called **auto regression**. To forecast the future, you can look back at the past few time periods and use those as a starting point. The information from the past will carry into the near future. Of course, as you try to predict farther into the future, an increasingly random number of events can arise that will make your forecast wrong. A **time series** is a set of outcome data that depends only on a time variable. As shown in Figure 8.1, a time series is relatively easy to understand. A single time series consist of one variable of interest (Sales) and shows how it varies over time. Time can be measured in almost any interval: hours, days, months, quarters, years, and so on. Months are a useful measure in business, but sometimes only quarterly data is available. Data might change due to external and random events, but at least to some extent, it follows some pattern based on recent values. If you can determine the relationship between the data and time, it is relatively easy to make forecasts.

Of course, a time series can be more complex. In many business settings, sales will often have a **seasonal effect**—where sales will increase or decrease in certain months of the year. For instance, as an extreme case, typical toy stores in the U.S. experience about 70 percent of their sales in the last quarter of the year—as holiday gift purchases. The essence of time series forecasting is to find these patterns to the data and use them to predict what will happen at the same point in the pattern in the future.

**Figure 8.1**

Simple time series. The simplest time series data consists of a single attribute plotted over a time interval. Here, total Sales are shown for each day. In many cases, a simple forecast can be made based on the information from the most recent time periods.



Almost every time series will have a random component that cannot be predicted. If the randomness is large, the prediction will almost always be inaccurate. Hence, every analysis should examine the degree of reliability based on the size of the random component compared to the predictable patterns. Some problems are relatively stable with small random elements. Others have high inherent random components which make it difficult to predict with any certainty. It is important to measure the level of randomness to provide a measure of variability—so a range forecast can be made instead of just a single point estimate.

Several tools can be used to analyze and forecast time series data. Simple tools include exponential smoothing—which averages out the random element; trend analysis—similar to regression which looks only at the basic direction over time; and auto regression—which measures the impact of prior period on trends. More complex tools combine these techniques. The most common method was developed by Box-Jenkins (see Box and Jenkins 1970) and is often called **ARIMA** analysis, which is an acronym for auto-regressive, integrated moving average. Because forecasting is so important to so many areas, several other techniques have been invented to analyze patterns over time. Some, such as ARIMA are relatively easy to automate and use for data mining. Others can be complex, such as spectral analysis, and can require considerable knowledge and supervision by the analyst. This chapter focuses on the easier methods that require less supervision.

## Business Situation

---

**Who needs to forecast data over time?** Finally, a question that is easy to answer: everyone makes forecasts. Some disciplines and some people are better than others, but some processes are more stable with fewer random elements. Predicting the weather is hard, particularly with microclimates, but mostly because of enormous random effects. Predicting economic trends is relatively straightforward—up to a couple of months. But economic activity is also subject to catastrophic (sudden) changes.

Why do businesses need to make forecasts? Consider the situation of a retail store. How long does it take to get products from the manufacturer onto the shelves? A few days, weeks, months? Although American production and distribution systems have become more efficient, months is usually the correct answer. So, a store has to order products several weeks or months before they are sold. That means the managers need a good estimate of what the sales level will be several months out. Even if you are not running a retail store, the same problem applies. Manufacturers need to know how many items to produce, what raw materials to order, how many employees to hire, and what size plant will be needed. All of these decisions are based on forecasts of sales.

Sales forecasts are critical, but the sales number is not the only item that needs to be predicted. Businesses also want to know what will happen to interest rates, foreign exchange rates, stock prices, and other financial variables. Investments and borrowing play key roles in managing any firm. Small changes in rates can have huge impacts on profits. Some financial variables can be forecast, others are difficult, but everyone tries to predict what will happen. Similar issues arise in terms of costs, such as supplies of raw materials, wage rates and worker availability, rents, maintenance expenses, advertising costs, and so on.

Time series analysis is also useful for separating out the basic elements present over time: seasonality, trend, and random effects. Many times decision makers want to observe larger trends without being distracted by seasonal changes. The

classic example is unemployment data. Because so many students graduate (at all levels) in the spring of each year, raw unemployment rates tend to be high. Yet, when looking at overall trends, these numbers cannot be compared directly with those for fall or winter. Consequently, time series tools are used to estimate and remove the seasonal effects to provide a more accurate estimate of the trend. The same situations can exist in production, sales, inventory levels, birth rates, and physical factors such as temperature.

Many business and manufacturing problems involve time series analysis. Almost any measurement can be treated as a time series, such as sales value, quantity of items sold, employee illness, accident rates, and quality measures. Weight measures taken every week are a classic time series. To minimize depression, you probably do not want to analyze your own weight; but growth rates and variations are critical in agriculture.

## Model

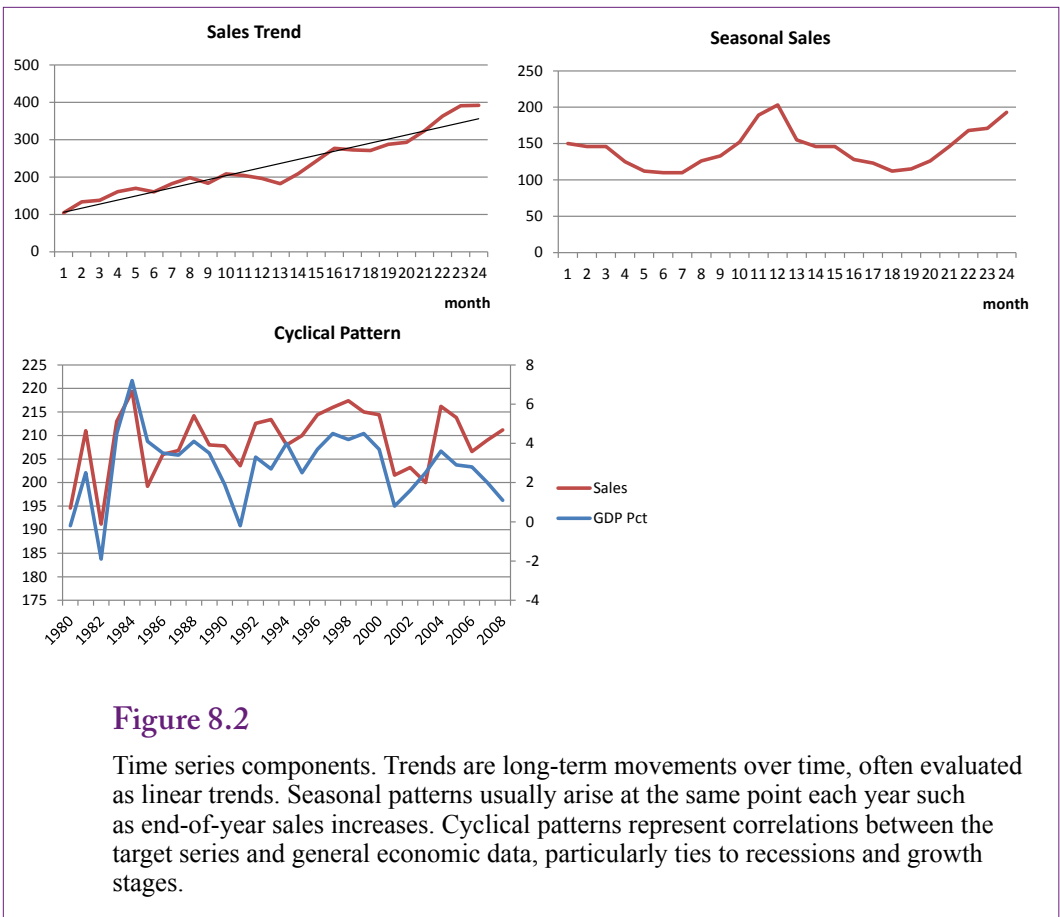
---

**What are the components of a time series and how can they be measured?** A single time series consists of one variable to be predicted, with observations collected over time. The time interval has to be fixed (such as days, months, or quarters). For the most part, it cannot contain missing observations. Data of this form typically exhibits some internal pattern—sales can increase over time, some months of the year might consistently be better or worse than others. If these patterns can be identified and measured, they can be used to predict future values of the series. The challenge lies in identifying and measuring the various effects. The focus of this section is to describe the common components of time series and define some of the mathematical background to show how they can be estimated. Evaluating individual components is useful not only for prediction, but also because it provides information about the underlying process. For example, managers can form better plans if they have good measures of seasonal effects.

### Time Series Components

A time series, particularly a business or economic series, is often defined in terms of four parts: (1) Trend, (2) Cycle, (3) Seasonal variation, and (4) random or irregular fluctuations. Some writers skip the cyclical components or lump them with trend changes. **Trend** represents an underlying pattern over time that is usually somewhat constant or at least independent of other time effects such as seasonality. For example, sales might increase at some base growth rate each quarter. **Seasonality** represents known changes that arise at about the same time every year. Typically they are defined in terms of years, but a process might have quarterly or monthly “seasons.” However, measuring seasonal patterns requires detailed data. For example, quarterly or monthly data is needed to see annual seasonal changes. If the season is defined as a shorter time period, it could only be measured with even finer data; and it can be difficult to obtain data at those levels.

**Cyclical** fluctuations are changes that are influenced by other economic data. Data for a specific business process could be influenced by broader economic measures. For instance, when the economy does well, personal income increases so customers might purchase more items from a specific company. When the overall economy dips into a recession, consumers buy less from this company.



**Figure 8.2**

Time series components. Trends are long-term movements over time, often evaluated as linear trends. Seasonal patterns usually arise at the same point each year such as end-of-year sales increases. Cyclical patterns represent correlations between the target series and general economic data, particularly ties to recessions and growth stages.

Because the government and other organizations track business cycles, it is possible to identify and measure the timing of these effects. In many ways, cyclical effects are really cross-correlations across time series: The overall economic GDP is a time series and its values are correlated with sales of specific businesses. Cross-correlations are more difficult to estimate than the other time series effects.

Random or irregular changes are movements that have no discernible pattern. Essentially, it consists of fluctuations that exist after the other effects have been removed. In the end, any time series will contain some random effects. Almost no process is completely predictable. The point is to examine the remaining effects and see if they truly are random and to see if they are small relative to the other effects.

Figure 8.2 illustrates the three main components. Trends are long-term movements, usually representing intrinsic growth rates. Most trends are estimated to be linear because long-term nonlinear trends can be risky to forecast. Sometimes a system exhibits a true nonlinear growth (or collapse) rate, and can be estimated with a small-order polynomial or log function. Seasonal data is tied to a regular distance in the calendar. The most common interval is an annual pattern (12 months or 4 quarters). It is recognizable by external knowledge (e.g., sales always increase at the end of the year), and by the fixed number of time periods between peaks and troughs. Cyclical data is the most difficult to identify and measure.

Many books do not count it as a major component because of these difficulties. Technically, it is represented by the correlation between the time series of interest (Sales) and an economic time series that measures the business cycle (GDP or GDP percent change). Notice how the two series move together in the figure. These correlations involving multiple time series are a relatively complex problem and many data mining tools do not handle them. If cyclical measures are critical to a problem, you should seek out sophisticated tools; and probably an expert analyst. Simple versions can be handled with the Microsoft BI tool so they will be considered briefly in this chapter.

The random fluctuations can be seen in the charts as small deviations from the presented patterns. In the examples presented, the random element was deliberately kept small to highlight the trend, seasonal, and cyclical components. Now, imagine the charts with a high random element. At some point, the random element would overwhelm the underlying pattern and it would be difficult to determine if a pattern existed at all or if the series was just random noise.

It is common to write an additive component model of a time series—where the components add up to the total value. At any time period  $t$ , the observed value  $y_t$  is

$$y_t = \text{Trend}_t + \text{Cyclical}_t + \text{Seasonal}_t + \text{Random}_t$$

Sometimes the model is written as multiplicative, where the plus signs are replaced by multiplication. But, this case can be converted easily to the linear model by taking the logs of the data.

Some methods of estimating time series data attempt to estimate the components directly. If you have measures for cyclical and seasonal data, linear regression is easy to use to estimate the main components. One trick is to simply use time  $t$  as a variable, and create dummy variables for seasons with values of one or zero, so you might have a linear form:

$$y = b_0 + b_1t + b_2\text{Summer} + b_3\text{Autumn} + b_4\text{Winter}$$

Other tricks include estimating seasonal values by computing averages for each month (all values for January, then February, and so on). However, a couple of models now dominate most discussions of time series analysis, and they are easier to automate. To understand the models, you need to understand two fundamental patterns in time series: auto regression and moving averages.

## Auto Regression

Auto regression is based on the concept that the next value in a time series is likely to be correlated with the current value. Think of it as momentum—most data make relatively smooth changes from one period to the next. Even if something radical changes, it takes a few periods for everything to be impacted. Yes, some systems are more **chaotic** and it is possible that small changes in underlying factors will lead to radical changes in the output variable. However, auto regression is a useful feature to estimate, and many systems will exhibit some of its features.

In mathematical terms, auto regression can be written so that the value at time  $t$  can be influenced by any of several prior points in time, from 1 to  $p$  periods back:

$$Y_t = a_0 + a_1Y_{t-1} + a_2Y_{t-2} + \dots + a_pY_{t-p} + \varepsilon_t$$

A separate coefficient ( $a$ ) is measured for each lag. The error term ( $\varepsilon_t$ ) represents a simple random error with zero mean and constant variance. The model is often abbreviated to the form: AR( $p$ ), such as AR(1) for a single-period lag, or



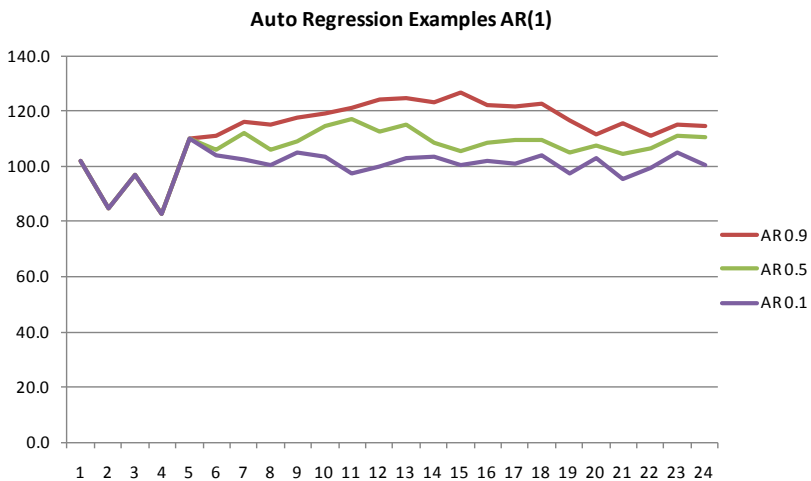
Time	AR 0.9	AR 0.5	AR 0.1
1	102.0	102.0	102.0
2	85.0	85.0	85.0
3	97.0	97.0	97.0
4	83.0	83.0	83.0
5	110.0	110.0	110.0
6	111.0	106.0	104.0
7	115.9	112.0	102.4
8	115.3	106.0	100.2
9	117.8	109.0	105.0
10	119.0	114.5	103.5
11	121.1	117.3	97.4
12	124.0	112.6	99.7

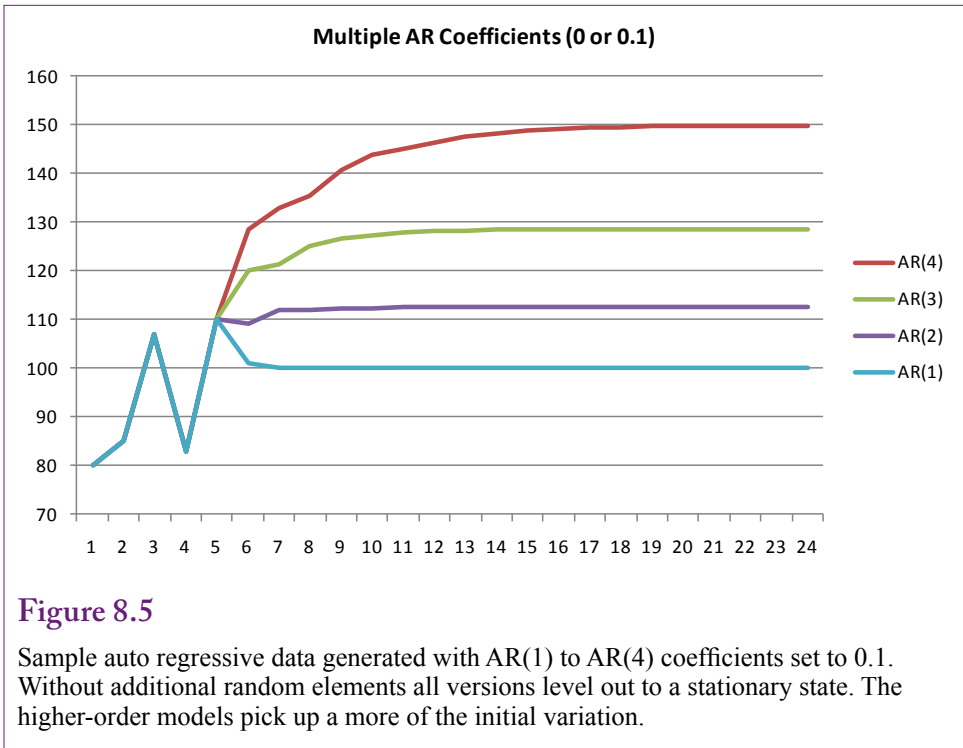
**Figure 8.3**

Sample auto regressive data generated with AR(1) set at three values (0.9, 0.5, 0.1). The mean was adjusted for each case to keep values within similar ranges.

**Figure 8.4**

Sample auto regressive data generated with AR(1) set at three values (0.9, 0.5, 0.1). Notice that the series with AR(1)=0.9 is smoother than the other two with smaller AR(1) coefficients. The high-proportion of value carried over from the prior time leads to slower changes in the data.





AR(3) to represent lag effects for the first three periods. Mathematically, the AR equation is a difference equation and some coefficient values will make it unstable. For example, the coefficient in an AR(1) model must be between -1 and 1, otherwise the model will predict an “explosion” where each subsequent value will increase rapidly, either positive or negative.

To understand the concept of auto regression, it helps to examine a few artificial examples. Figure 8.3 presents sample time series that were generated using three different values for AR(1). The mean coefficient was adjusted each time to hold the results to similar ranges. Random elements were added to each point. Note that the first five terms are the same in each series—to provide the same foundation for each series. Data were generated for 24 periods, but only the first few are shown in the table.

The effect of the parameters is difficult to see in the raw data. Figure 8.4 shows the charts of the three series. The most important result is that when  $AR(1)=0.9$ , which is a high value, the chart is smoother. The data shows less variation than in the other cases. The reason illustrates the role of the AR coefficient. A high value means that a large portion of any forecast consists of the prior value. Carrying such a high percentage of value each time means that small changes are smoothed out. Instead, the curve adopts longer, slower changes.

Figure 8.5 takes a slightly different approach. In this example, all of the lag coefficients have the same value or they are zero. The difference is the number of lags included in the model. AR(1) has one non-zero lag (0.1), and AR(4) has the first four lag coefficients set to 0.1. That is, AR(4) picks up effects from the four prior time periods. Notice that all four models eventually level off to a stationary state—because no additional random data was added and the model is stable.

Time	Sales	MA 3-c	MA 5-c
1	75		
2	89	85.3	
3	92	90.3	89.6
4	90	94.7	94.2
5	102	96.7	97.6
6	98	102.0	97.8
7	106	99.0	97.2
8	93	95.3	96
9	87	92.0	96.6

**Figure 8.6**

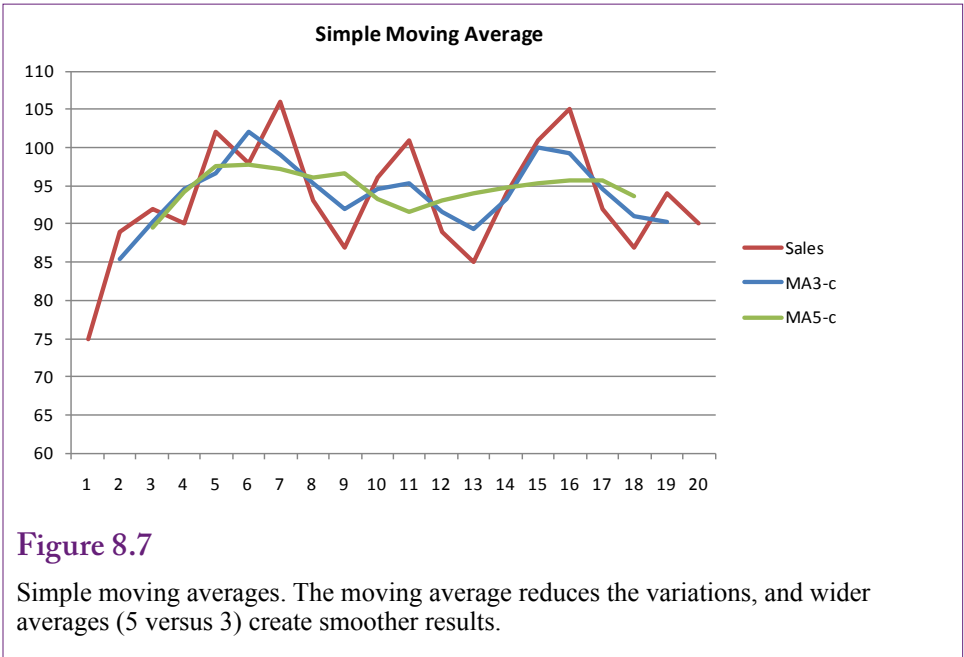
Simple moving averages. A three- and five-period average centered to keep the series aligned.

Also, the same mean was used in all four models, so the models with more lags will have higher average values because of the added terms. This decision was made to keep the lines separate so the patterns would be more visible.

In terms of differences, all series have time points 1 – 5 in common. Time 6 is the first predicted value. The AR(1) model drops simply because the mean stays the same and the overall values are lower. There is no variation in the line due to earlier lags. AR(2) picks up the drop at t-2 (time 4) and it picks up the gain at t-1 then levels off. AR(3) picks up those two effects plus the peak at lag t-3. AR(4) also picks up the peak, but it is mitigated by the additional drops at lag t-4. The point of the charts is that if a time series has a seasonal effect at some periodicity that lag value needs to be included in the estimation. For instance, if the data is collected at quarterly intervals and the model estimates only an AR(1) lag, then any annual changes present at AR(4) will not show up in the prediction. So a key aspect of using auto regression is identifying the specific lags that should be included in the model. It might be tempting to include all possible lags up to a year, but it takes enough data observations to estimate all of the values. Plus, many times the in-between lags are not interesting, so models are often estimated with the first two or three lags, plus a seasonal lag; skipping the ones in the middle.

### Moving Average

In some ways a **moving average** is relatively easy to understand, in others, it is complex. By its basic definition, a moving average is used to smooth data. Figure 8.6 shows the calculations for a three-period and five-period moving average. Both computations are centered to keep the series aligned; otherwise the moving average series will be offset and difficult to compare to the original. As you can tell from the name, a three-period moving average computes the average of three consecutive values. Shifting up a time period, the system computes the average of the next three items shown in each box. A five-period average uses five consecutive entries. The width of the average (3 or 5) is somewhat arbitrary. If you stretch the interval to cover an entire year; such as 4 points for quarterly data or 12 for monthly data; the average will eliminate all seasonal variations.



The results are easiest to see in a chart. Figure 8.7 shows the original data and the two moving averages. Clearly the two averages reduce the variability in the overall series. The five-period average is also smoother than the three-period values. The technique can be used to reduce noise due to random events. It can also be used to average out seasonal effects to highlight underlying trends or other patterns.

The term moving average derives from this common method of calculating averages to reduce or eliminate seasonal effects. However, you will rarely actually perform this computation. Instead, the underlying mathematical model is more important. The moving average process defines the outcome variable ( $Y$ ) at a point in time as a mean plus a weighted average of the differences from that mean in prior periods:

$$Y_t = \mu + \psi_0 \varepsilon_t + \psi_1 \varepsilon_{t-1} + \psi_2 \varepsilon_{t-2} + \psi_3 \varepsilon_{t-3} \dots$$

### Figure 8.8

Simple moving averages. Consider a simple moving average model that has four independent seasonal effects (quarters). Computing the four-period moving average yields the mean plus an average of the seasonal effects. If the seasonal effects net to zero, the moving average yields just the overall average.

$$\begin{aligned} Y_1 &= \mu + S_1 \\ Y_2 &= \mu + S_2 \\ Y_3 &= \mu + S_3 \\ Y_4 &= \mu + S_4 \end{aligned}$$

Compute the average:  $(Y_1 + Y_2 + Y_3 + Y_4)/4 = \mu + \text{Avg}(S_i)$

		S1	S2	S3	S4	S1	S2	S3
Yt	$\mu$	$+\psi_0\epsilon_t$	$+\psi_1\epsilon_{t-1}$	$+\psi_2\epsilon_{t-2}$	$+\psi_3\epsilon_{t-3}$			
Yt-1	$\mu$		$+\psi_0\epsilon_{t-1}$	$+\psi_1\epsilon_{t-2}$	$+\psi_2\epsilon_{t-3}$	$+\alpha_3\epsilon_{t-4}$		
Yt-2	$\mu$			$+\psi_0\epsilon_{t-2}$	$+\psi_1\epsilon_{t-3}$	$+\psi_2\epsilon_{t-4}$	$+\psi_3\epsilon_{t-5}$	
Yt-3	$\mu$				$+\psi_0\epsilon_{t-3}$	$+\psi_1\epsilon_{t-4}$	$+\psi_2\epsilon_{t-5}$	$+\psi_3\epsilon_{t-6}$
Avg	$\mu + \text{Wtd. Avg (S1)} + \text{Wtd. Avg(S2)} + \text{Wtd. Avg(S3)} + \text{Wtd. Avg(S4)}$							

**Figure 8.9**

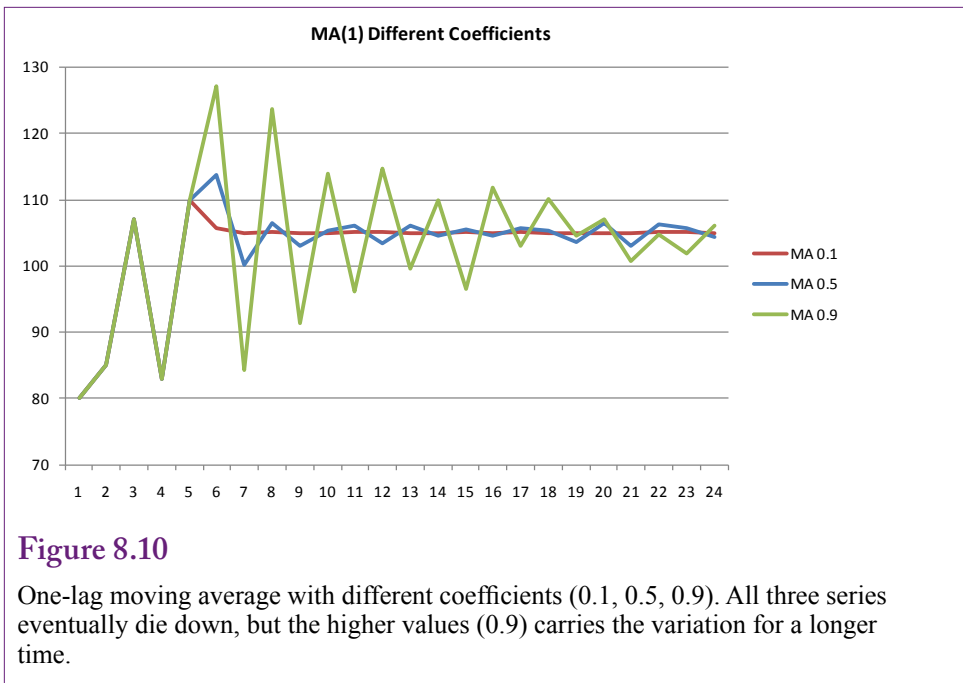
Quarterly moving average. In the more general model, the moving average over all four seasons results in the overall mean plus weighted averages of each of the four seasonal effects. If the seasonal effects are neutral, and the weights meet some identity conditions, the weighted sums will effectively average out to zero.

Conceptually, the errors or deviations ( $\epsilon$ ) could be considered as seasonal effects, and pure random error. Figure 8.8 shows a simplified model that contains independent seasonal effects for four seasons or quarters. Computing the four-period average yields the underlying mean plus an average of the seasonal effects. If the seasonal effects are neutral in total, their average will be zero and the moving average yields just the overall mean. That is, a moving average that covers the entire season should eliminate the seasonal effects.

The general model for the moving average leads to more complex results, but the concepts are similar. Figure 8.9 shows a quarterly model with a four-period moving average. If the seasonal effects are neutral—gains in one season are offset by declines in other seasons—and the weights meet some identity constraints that are beyond this book, then the weighted sums effectively average out to zero when the moving average spans the entire year. So, again, an average across the entire year yields an estimate of the overall mean. The situation is more interesting and more valuable when the moving average spans less than a full year. For example, a problem might compute a three-month average. In this case, the moving average model shows that the predicted value will consist of the mean plus weighted errors (e.g., seasonal effects) from prior months. That is, the moving average model transfers information forward from the errors or seasonal effects. Remember this interpretation of the model. It will be needed in the ARIMA section that combines the models for auto regression and moving averages.

Charts can help show the effect of the moving average coefficients and the different lengths of the average. Figure 8.10 shows three series created with the general moving average model using only a one-period lag on the error term. The coefficient weight of the lag was tested with three different values (0.1, 0.5, and 0.9). The series were started with relatively high variation in the first five periods. Notice that all three series eventually die down to the mean with just random errors. Also, notice that the larger coefficients (particularly 0.9) carry the initial variation for a longer time. This result is clear from the equation because the next-period value is computed by adding the multiple of the coefficient and the prior error.

Figure 8.11 examines moving averages with more lags—from one to four time periods. All of the coefficients were set to the same 0.5 value. All series had the same starting values with relatively high variation. No random effects were add-



**Figure 8.10**

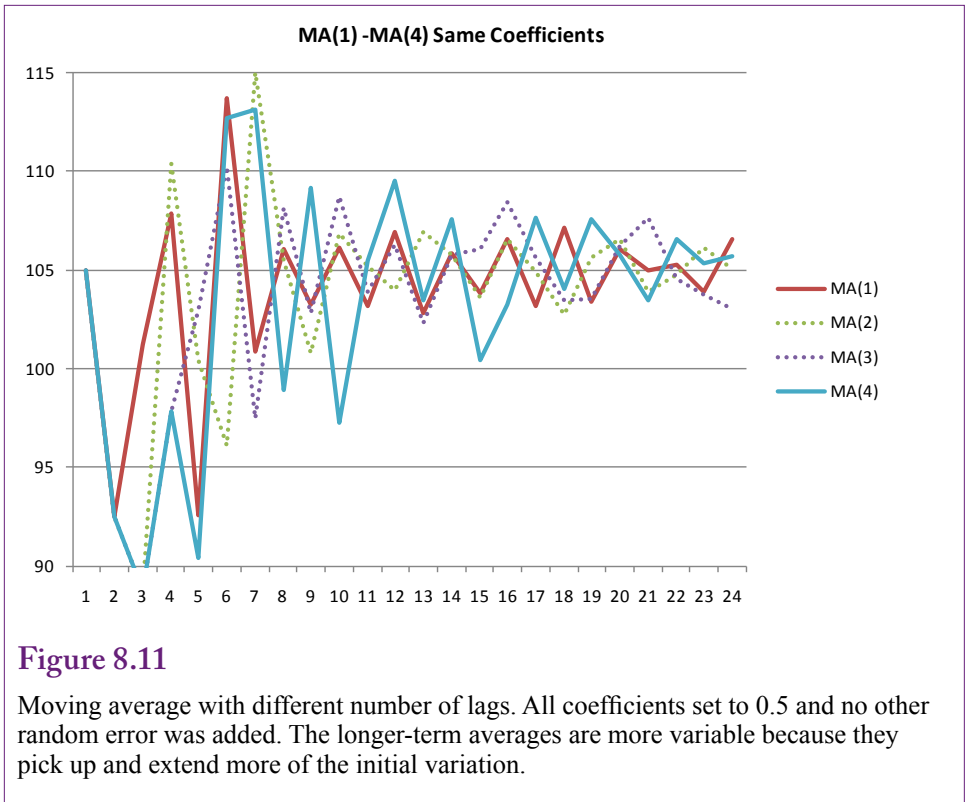
One-lag moving average with different coefficients (0.1, 0.5, 0.9). All three series eventually die down, but the higher values (0.9) carries the variation for a longer time.

ed beyond the starting values. The series that includes four lag effects is clearly more variable for a longer period of time than the single lag. The point is that using more lags in the model can capture seasonal effects for more time periods. These effects are carried into future forecast values. Eventually, you will face the question of how many lag periods should be included in a real-world model. The answer is often difficult. Using more lags can lead to a more realistic model—if the seasonal effects truly cover more time periods. Using fewer lags results in a smoother model—useful if the lagged variations are random errors instead of seasonal effects. Diagnostic tools can help you determine if variations are random or part of a seasonal pattern, but it helps if you understand the fundamental business data being examined.

## Trends

As an analyst or business person, trends in time series data are important. An overall increase in sales would indicate that the average sales level is increasing each year. Likewise trends in stock prices or interest rates would signal long-term changes, which would give the wise investor the opportunity to make money. Trends can be linear or nonlinear. Linear trends are relatively easy to estimate using linear regression. Nonlinear trends are trickier because you might have to identify the degree of nonlinearity. Typical polynomial equations and log-linear equations can also be estimated with standard computations and regression techniques. Techniques for using regression to estimate trends are covered in the linear regression section of this chapter. As a hint, a simple approach is to include time ( $t$ ) as a variable, where time can be a simple numerical sequence:

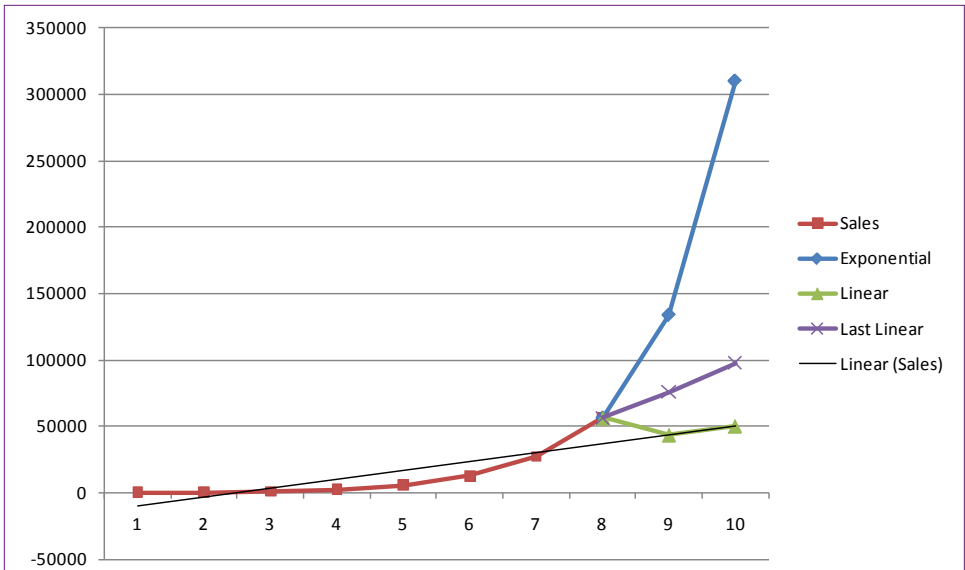
$$Y = b_0 + b_1 t + b_2 \text{ seasons} + b_3 \text{ others}$$



Other techniques are also possible, but it is generally best to stay relatively close to linear or cubic trends when forecasting time series data. Highly nonlinear patterns, particularly exponential growth, rarely exist for long periods of time. A small company might observe exponential growth for a year or two—because the initial base is so small. But, predicting an exponential growth beyond a couple of months is rarely going to work out. It is almost always better to be conservative and forecast a linear trend, even if it seems less accurate over historical data. Figure 8.12 shows the basic problem. The initial data follow an exponential growth rate, but there just is not enough data to be comfortable with an exponential forecast. Using an exponential trend, sales would leap from 56,000 in period 8 to over 310,000 only two periods later. Sure, it could happen, but without advanced knowledge of the industry and testing, such a forecast would be risky. At the same time, a simple linear trend using all of the initial data is likely to be too low—notice that the linear trend forecasts an initial drop in sales. A polynomial (cubic) equation might be a good fit, but it is going to forecast a relatively large increase in sales as well. An alternative is to use the last few periods to generate a linear forecast from that point. This decision and the selection of the starting point would have to be made by a human, not an automated algorithm; because it requires knowledge of the industry and examination of similar patterns from other organizations if they exist.

The interesting problem at this point is that in pure time series analysis, trends cause problems in the estimation of the seasonal and random data. The ARIMA technique described in the next section only works if the data contains no trend. Fortunately, that statement does not mean ARIMA is rarely useful. Instead, it





**Figure 8.12**

Dangers of nonlinear forecasts. An exponential trend would predict a five-fold increase in only two periods. Linear using all data is probably too conservative. Linear using the last few observations is probably a safer forecast.

means that you have to remove the trend before applying the ARIMA analysis. Most tools have an easy method to accomplish this task. If you see a trend in the data—usually by plotting it—the tools can analyze the difference instead of the raw numbers. So, if you see a linear trend, you can specify a single difference step to compute:  $z_t = y_t - y_{t-1}$ , and then base the analysis on the  $z$  series instead of the original. If the pattern is nonlinear, you might need to take a second difference. In extreme cases, you can convert the data using logarithms:  $z_t = \log(y_t)$ , and possibly difference those values if you still see a trend. The details of the differencing process are explained in the ARIMA section. The point is that you need to recognize and often test for trends in the data before trying to estimate seasonal component.

## ARIMA

**Auto regressive integrated moving average (ARIMA).** The name makes it seem like a difficult topic, but the name also clearly defines what the tool does: combine auto regression and moving average time series estimation into one package. Besides, everyone just uses the initials ARIMA. The methodology is sometimes called Box-Jenkins after the two authors who developed the process for analyzing time series data. The clearest way to understand the combined approach is to write the mathematical model:

$$y_t = \theta_1 y_{t-1} + \theta_2 y_{t-2} + \dots + \theta_p y_{t-p} + \varepsilon_t + \psi_1 \varepsilon_{t-1} + \psi_2 \varepsilon_{t-2} + \dots + \psi_q \varepsilon_{t-q}$$

The equation states that a series value  $y$  at time  $t$  is generated from two parts. An auto-regressive component that pulls information from prior  $y$  values, and a moving average component that pulls information from prior error terms—which

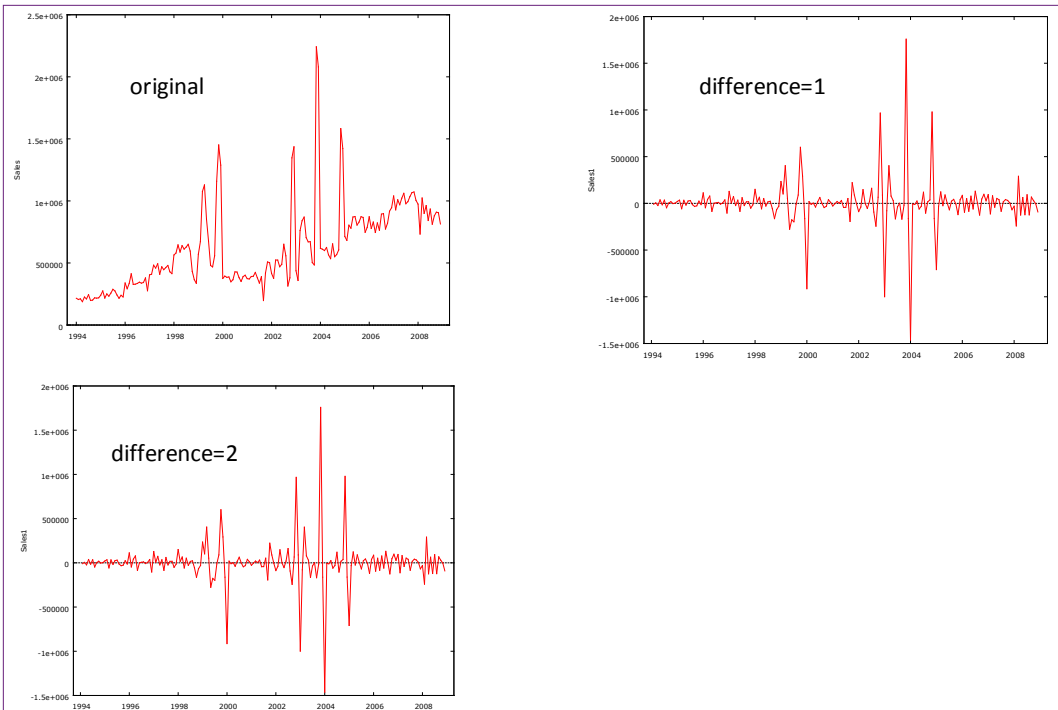
could represent seasonal variations. This version of the model is usually abbreviated ARMA(p,q) where p represents the highest lag for the auto-regressive terms and q the highest lag for the moving average terms. Note that the “I” is missing because it is not included in this model and will be explained shortly. The interpretation of the equation is straightforward. Each value in a time series is affected by the values that came before it and the variations from the average are also influenced by the variations in prior periods. The goal of ARMA is to estimate the values of the coefficients to determine the weight that each prior value plays in determining future outcomes. That is, the coefficients define the time series pattern.

One important catch with estimating the ARMA equation is that it can be estimated only if there is no trend in the data. In mathematical terms, the time series must be stationary—where the series revolves around a fixed mean. The concepts of stationary and identifiability are covered in advanced textbooks on time series. Most tools automatically notify you if the conditions are violated, so you do not need to be able to verify them manually. Still, you do need to know that if a trend exists in the data, it needs to be removed. The most common solution is to define a new series  $z_t = y_t - y_{t-1}$ . If the trend is linear, this simple differencing usually eliminates the trend and the coefficients can be estimated. If the trend is nonlinear, you might need a second differencing—where the z values are subtracted. In extreme cases, you might need to convert the original y values by defining a new  $\log(y)$  variable—which often needs a single differencing to be usable. Because differencing is the most common approach, it defines the integrative aspect of the model: ARIMA(p,d,q) is defined with p the number of auto-regressive lags, d as the number of differences, and q as the number of moving average lags.

### *Differencing to Remove Trend*

Most traditional ARIMA tools require supervision and the analyst must specify the p, d, and q values. So, how do you know which values to choose? The d value is usually the easiest because it is almost always 0, 1, or 2. For a quick approach, simply plot the time series and see if a trend exists. If there is no apparent trend up or down, it will not be necessary to difference the data, so d equals 0. If there appears to be a linear trend, choose d equal to 1. If the trend appears nonlinear, perhaps as in a cycle, set d equal to 2. Avoid jumping automatically to the highest value. If you did not choose enough differencing and a strong trend remains in the data, it is unlikely that the ARMA equation can be estimated and the tool will generate an error. So increase the value of d and try again.

The lag limits p and q are trickier. Some tools use statistical methods to help estimate the appropriate values automatically. Others follow the Box-Jenkins methodology and ask you to examine two charts to help determine the appropriate values. An additional complication exists. In the equation, p and q are the highest values of the lags, and all intermediate lags are estimated as well. Often, the time series will not have enough data to estimate every possible lag value. However, you still might need some higher-order lags. For example, some time series can be estimated with p and q set to 2 or 3. However, many time series also exhibit seasonal patterns—particularly annual events such as holiday sales. Using monthly data, these patterns would arise at lag values set to 12. Because the values at lag periods 4-11 are likely to be minor, you typically do not want to estimate those lags. Consequently, most ARIMA tools allow the analyst to select potential lags by entering specific numbers, such as 1, 2, 3, 12; skipping the intermediate values that are unlikely to be relevant. Of course, the analyst needs to know or guess which values might be important.



**Figure 8.13**

Differencing series to remove trend. The original series appears to have a linear trend. One difference has removed the trend, leaving a flat line with variations. A second difference operation does not appear to improve the pattern.

Figure 8.13 shows the time series of total sales by month for Rolling Thunder Bicycles. The original data plotted over time clearly shows a trend that needs to be removed. A series that computes the first difference appears to remove the entire trend, leaving variations along a horizontal line. There might be a slight trend remaining at the right-end of the chart. As comparison, a second difference is also plotted, and it appears to be the same as the single difference. Hence, a single difference should be sufficient

### *Autocorrelation and Partial Autocorrelation*

The Box-Jenkins approach emphasizes the importance of the **autocorrelation function (ACF)** and the **partial autocorrelation function (PACF)**. These two functions are used to evaluate whether the model will be solvable and to help identify the appropriate values for  $p$  and  $q$ . Autocorrelation is the correlation between any two time series observations separated by a lag of  $k$ . The function is computed and plotted for several (perhaps 20) values of  $k$ . The correlation is computed as the covariance between all points separated by  $k$  time periods, and divided by the variance of  $y$  to obtain a value between  $-1$  and  $+1$ . The values are estimated from the formula where  $\bar{y}$  is the mean of the  $y$  values:

$$r_k = \frac{\sum_t^{n-k} (y_t - \bar{y})(y_{t+k} - \bar{y})}{\sum_t^n (y_t - \bar{y})^2}$$

The partial autocorrelation is loosely the correlation between two points separated by  $k$  time units without the effects of any intermediate observations. It is computed for  $k > 1$  to be:

$$r_{kk} = \frac{r_k - \sum_{j=1}^{k-1} r_{k-1,j} r_{k-j}}{1 - \sum_{j=1}^{k-1} r_{k-1,j} r_j}$$

Where  $r_{kj} = r_{k-1,j} - r_{kk} r_{k-1, k-j}$  and  $r_k$  is the sample autocorrelation at lag  $k$ . Fortunately, tools exist to compute and plot both functions with respect to the lag values. The charts are called correlograms.

Figure 8.14 shows a sample ACF and PACF from a hypothetical time series. First, observe that the ACF dies down relatively quickly, while the PACF simply cuts off after 2 or 3 lags. Some systems will display confidence interval bars to help determine if the correlations at each lag are significantly different from zero. To understand the charts, first note that both functions either die down or cut off. If one of the two (or both) failed to die down, it would indicate that the series was not stationary and still had a trend that needed to be eliminated through differencing.

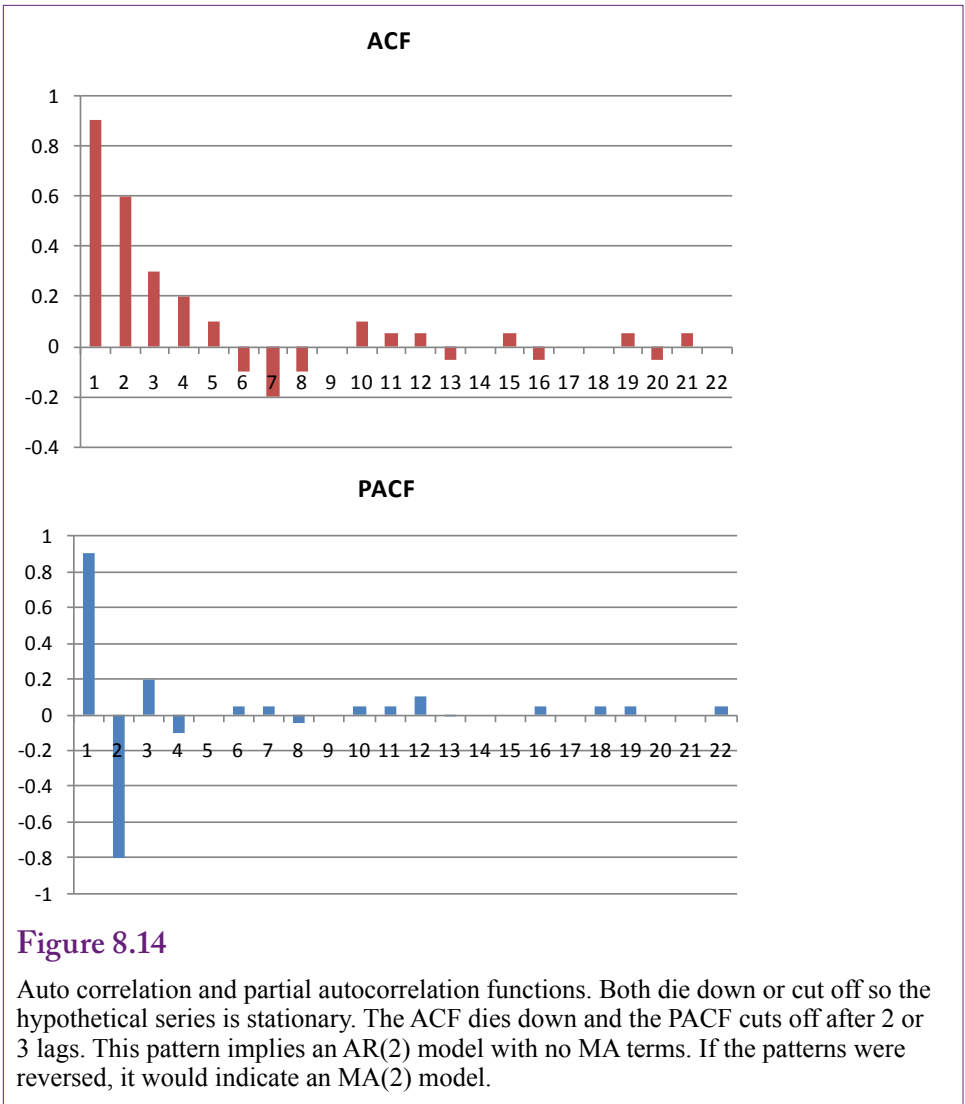
The most important aspect of the ACF and PACF is that they provide an indication of the lag values to be tested for the AR and MA components of the model. The key is to look for the point where either the ACF or PACF cuts off. In this example, the PACF cuts off at lag 3, with lag 2 significant, while the ACF dies down. This pattern (AR die down, PACF cut-off) is representative of an AR(2) series, so the ARIMA model could be run with  $p=2$ ,  $d=0$ , and  $q=0$ . If the pattern were reversed where the PACF died down and the ACF cut off, it would be an indication of an MA(2) model, or ARIMA(0,0,2).

In many cases, the PACF or ACF will have additional spikes at later lag points. In particular, it is common to see a spike at lag 12 in monthly data to indicate an annual seasonal pattern. If these spikes are widely scattered, they can usually be specified individually in the ARIMA definition, such as  $p=1, 2, 3, 12$ . If both the ACF and PACF exhibit cutoffs, it is likely that the model will need lags for both the AR and MA components, such as ARIMA(3, 0, 3).

The initial estimate of the AR and MA lags does not have to be perfect. You just need decent estimates to provide a starting point for the estimation. Even with tools, such as Microsoft Time Series, the estimation process works best if you can provide a reasonable starting point. The estimation results will provide additional information to evaluate the significance of each lag coefficient.

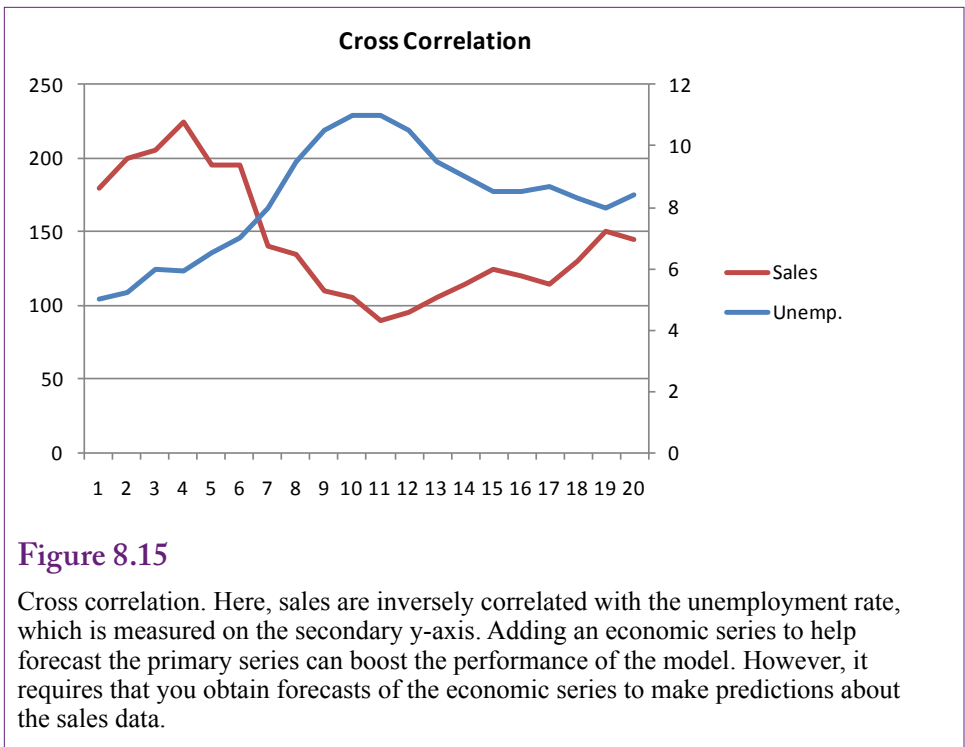
## Cross Correlations

The ARIMA process is probably the most common method used to forecast time series. In addition to its long-term forecast capabilities, it provides measures of the lag and seasonal effects which provide useful information to explain the underlying patterns. However, ARIMA is designed to work only with a single series of data at a time. It estimates purely internal patterns within that series. In some business problems, particularly in finance, it is clear that changes in one series will affect the outcome of the series being investigated. The business cycle or general economy provides a classic example. If the overall economy is doing poorly, people have less income, more people are out of work, and they are less likely to buy expensive products. To analyze business sales, it might be useful to match the sales pattern to the economic pattern. These **cross correlation** models are useful



when a model exists that explains how some attributes affect the predictable series. Economic examples are common, and the technique is often used in financial forecasts. It can also be useful when physical or biological relationships exist. Essentially, cross-correlation combines traditional economic modeling with time series patterns. In fact, regression tools are probably the easiest method for evaluating combined cross-correlation and time series models. Regression can estimate the coefficients on the independent attributes as well as the values of the autoregressive and moving average terms.

Figure 8.15 shows a hypothetical example of Sales inversely correlated with the unemployment rate. By economic theory, sales at most firms will be correlated with the economy—as a measure of consumer income. It might be easier to understand the relationship if real GDP or income were used instead of the unemployment rate. However, the U.S. Bureau of Economic Activity (BEA) only releases quarterly and annual data for GDP accounts. Unemployment, inflation, and interest rates are available monthly which provides more data for better analyses.



One additional warning about government data is critical when working with time series. Many government time series are provided as **seasonally adjusted**. Seasonally adjusted data has been averaged to remove seasonal patterns. Fortunately, the online Web-based tools make it relatively easy today to obtain either the raw data or the seasonally adjusted series. You simply have to set a check box, but you must decide which version of data you want to use for each specific situation. If you choose the raw data, then the estimation will attempt to apply any seasonal patterns in the reference series to your data. In the example with the unemployment rate, it might be best to use seasonally adjusted data. Unemployment rates are strongly affected by school graduations and those seasons might not match typical sales. If the problem is using unemployment as a proxy for the overall state of the economy and consumer income, the seasonally adjusted values would be a better economic measure. If the problem truly needs to rely on the number of unemployed workers available, then the raw data would be better. The point is that the answer depends on the specific problem being studied and the analyst needs to make the decision to match the problem.

A second type of problem often crops up in business forecasting where cross correlation is useful. Remember that time series forecasts require a relatively long history of data—preferably at least 50-100 observations. So how do you forecast a series for a new product or a new business region? Will you have to wait four years to gather enough monthly data to make a forecast? The solution is to assume that the new product or new region will follow a pattern similar to an existing product or region and use the cross-correlation as the basis to forecast sales for the new region.

One of the drawbacks to building a cross-correlation model is that to predict the dependent variable (e.g., Sales), it is also necessary to predict the series for each of the independent variables. So, instead of making one prediction, you must first predict every other series that is influencing the outcome. If the independent series consist of common economic data, it is usually possible to find at least short-term predictions made by several organizations that track general economic activity. If the independent series consist of internal data, the series must be predicted using separate models or separate time series analysis of each series. Before embarking on building a cross-correlation model, first decide if it will be easier or harder to forecast the other series that will be used in the model. Trading one problem for a harder one is going to cause problems.

Models that use extensive numbers of cross series will probably benefit by being estimated with least squares regression. Other tools, such as Microsoft Time Series, include the ability to add external attributes to the forecast method. Microsoft BI embeds these attributes into a decision tree solution and provides different estimates of the ARIMA model for each leaf on the tree. Details are explained in the tools sections.

### Evaluating Models

It should be clear by now that time series models are difficult to automate. Most tools require supervision by the analyst. Ultimately, the analyst must choose among several model variations. In the ARIMA model, the selection of  $p$  and  $q$  ultimately requires judgment. Is there a way to evaluate the resulting models?

Several overall measures have been defined for evaluating ARIMA models. Box-Jenkins proposed the  $Q$ -statistic which measures the remaining autocorrelation of the residuals after the model has been fit:

$$Q = (n - d) \sum_{i=1}^K r_i^2(\hat{\theta})$$

The value for  $K$  is somewhat arbitrary—it is chosen to see if the first  $K$  autocorrelations together account for too much variation.  $K$  is often chosen to be 12, or perhaps 24 or 36 for large models. The number of observations is  $n$ , the number of differences taken is  $d$ . The  $r^2$  terms are the square of the autocorrelations in the residuals separate by the specified lag ( $i$ ).  $Q$  follows a Chi-square distribution with  $K - n_p$  degrees of freedom, where  $n_p$  is the number of parameters estimated in the model. If  $Q$  is too large the model is rejected on the grounds that too much autocorrelation remains within the residuals.

Another common measure is a variation of the RMSE:

$$s = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n - n_p}}$$

Models with smaller error values ( $s$ ) are better. Check this value when comparing models and pick the model that has a substantially lower value of  $s$ . Most tools report this measure, the Chi-square statistic might not be reported automatically. Other writers have proposed numbers that have similar properties in general. For instance, the **Akaike information criterion (AIC)** becomes smaller as the AR variance decreases; so smaller values are better. Schwarz proposes a similar



measure (**Schwarz criterion**) that is sometimes reported. The Schwarz measure is sometimes known as the Bayesian Information Criterion (BIC) because of the methodology he used to derive the measure. Most tools report at least some of these measures so the analyst can compare models to see which one generates the smallest errors. Judge [1985] provides the derivation and interpretations of these measures, but as long as they are computed by the software tool, the values can be compared across different models without worrying about the differences in interpretation.

The main challenge with ARIMA is to get the correct selection of AR and MA lags ( $p$  and  $q$ ). Each different collection represents a different model. Whenever a model is evaluated, care must be taken to ensure that the model actually makes sense. Are there significant coefficients? Do the autoregressive and seasonal terms fit the known facts? Look at the residual plots. Do substantial trends or correlations remain in the residuals? Try a forecast to see if the model values match the actual patterns.

## Data

---

**What types of data can be used in time series analysis?** The short answer is that a single attribute is evaluated at a defined time interval. For example, sales totals are computed for each day, week, month, or quarter. The key is that there can be only one attribute and the time interval has to be constant. Some tools support cross correlations where one series can affect a second series. In these cases, the series must contain exactly the same time observations—the same interval, and the same starting and ending points. The data itself should be continuous data. Typically it is a subtotal such as sales value or quantity. Almost any measurement data will work, including temperature, distance, and weights.

The time interval must remain constant for the entire series, and there must be enough observations to estimate several parameters. Common recommendations are at least 50-100 data points. The catch is that it takes time to record the observations. Fifty points is four-years of data collected monthly. It might be tempting to use more detailed data—weekly or even daily—just to collect enough data points. The drawback to detailed data is that it can be harder to interpret the results. Daily data might work if the business exhibits patterns during the week, such as low sales on Mondays and high sales on weekends. But, the daily values will not provide information about seasonal patterns (such as July versus December). The key is to decide what types of patterns are likely to be important and use that choice to select the time interval.

### Attributes and Observations

Only a single attribute will be used for most analyses. The data is stored with the time values in rows. Traditional tools work well with just the single column of data for the series. Microsoft Time Series requires a second column to provide an identifier or key value. Each observation row must be assigned a unique key value. For time series data, this value is typically either a simple integer or it is a combination value for the date interval. Microsoft tools only use the value to sort the data and display cutoff points, so the intervals do not have to be continuous.

Most time series data is aggregated, and the values can be computed using a GROUP BY query or an OLAP cube. The requirement in Microsoft BI of using a single key column often requires some work to define a column that concatenates date parts. For example, assume the database contains a SaleDate column that lists

sales by day. Analyzing the data by month will require the creation of a column that combines year and month and then computing the subtotal by that column. The catch with Microsoft BI is that the time identifier column should be numeric. However, that objective is somewhat easier to achieve than it first sounds. The simplest way to create the time key column is to define a new column:

`Year(SaleDate)*100+Month(SaleDate) As YearMonth`

The Year and Month functions return numeric values. Multiplying the year by 100 shifts it two places to the left to leave room for a two-digit month. The resulting column contains values such as 200801, 200802, 200901, and so on. These values are sorted correctly by year first and then month. However, note that values cannot be subtracted from each other to obtain a time distance. Subtraction will work within a given year (200803 - 200801 = 2 months from January to March), but not across years (200901-200812 is way more than one month). Because Microsoft BI uses the column only for sorting, subtraction is not an issue; just remember not to use the column for any other purpose.

Microsoft BI can use other attributes to determine how they influence the predictable variable. To include other attributes in the problem, they will have to be treated as additional columns in the query. Most importantly, the time identifiers (rows) must exactly match those of the predictable series. If the data comes from other internal tables, it can usually be computed as subtotals the same as the primary series. If the data is retrieved from external (e.g., government) sources, it most likely will need to be assigned the same time key value used for the primary series. At that point, a JOIN query can match the rows correctly between the imported comparison series and the predictable series.

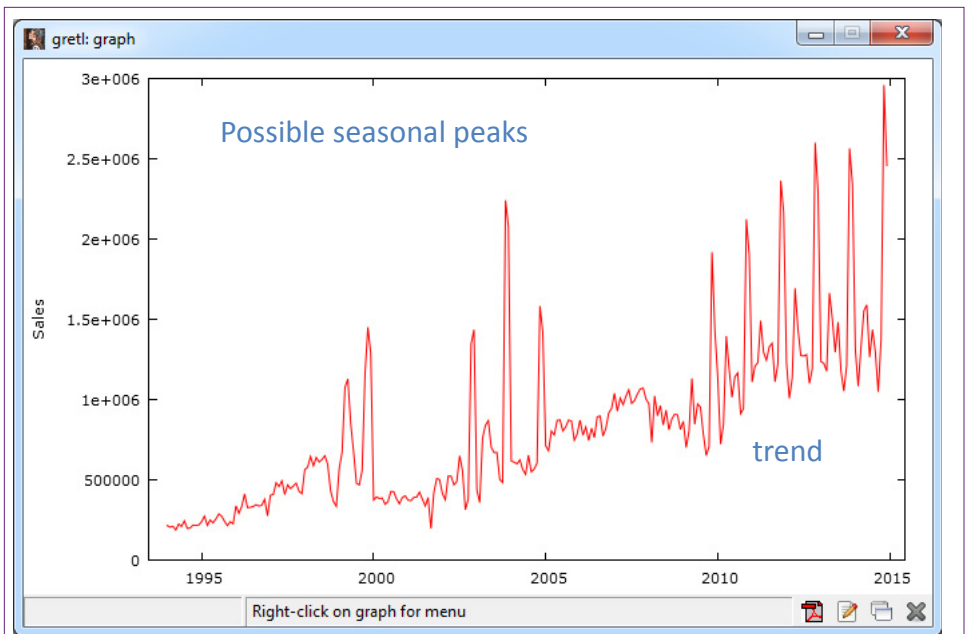
## Missing Data

A time series should not contain missing observations. For instance, with monthly data, it cannot skip some months. If a series has many missing points, it is better to aggregate up to a wider time interval. For example, if daily data has many holes, aggregate and perform the analysis with weekly or monthly data. If a series has a small number of missing values, they can be imputed through a variety of methods. The most common are: (1) average the two nearest values, (2) copy the nearest older neighboring value into the missing value, (3) compute an average of all data for that time slot (e.g., quarter 1 or specific month). All of the methods have some drawbacks, which is why series with many missing values should not be used. Microsoft BI provides options similar to these to automatically replace missing observations. However, a query should be used to count the number of missing values first to decide if the number is small enough to reduce the impact on the results.

## Traditional ARIMA Estimation

---

**What results does traditional ARIMA provide?** The model section of this chapter provides a detailed description of how ARIMA works. Most tools that implement the traditional version follow the Box-Jenkins process. Microsoft Time Series has a version of ARIMA but it is mixed with a proprietary algorithm so it is a different process described in the next section. The traditional Box-Jenkins process essentially provides tools to plot the data, create ACF and PACF charts, and estimate the ARIMA model according to lags and differencing specified by the analyst. The analyst needs to organize the data into a series, check the model



**Figure 8.16**

Rolling Thunder Bicycle sales by month. Notice the upward trend and the potential seasonal peaks. Some of the stronger peaks might be difficult to predict.

for trends, then try to find the best combination of auto-regressive and moving average lags for the model. The lag structure provides the key information about seasonality. The model itself and most tools make it easy to forecast future values for the series.

## Goals

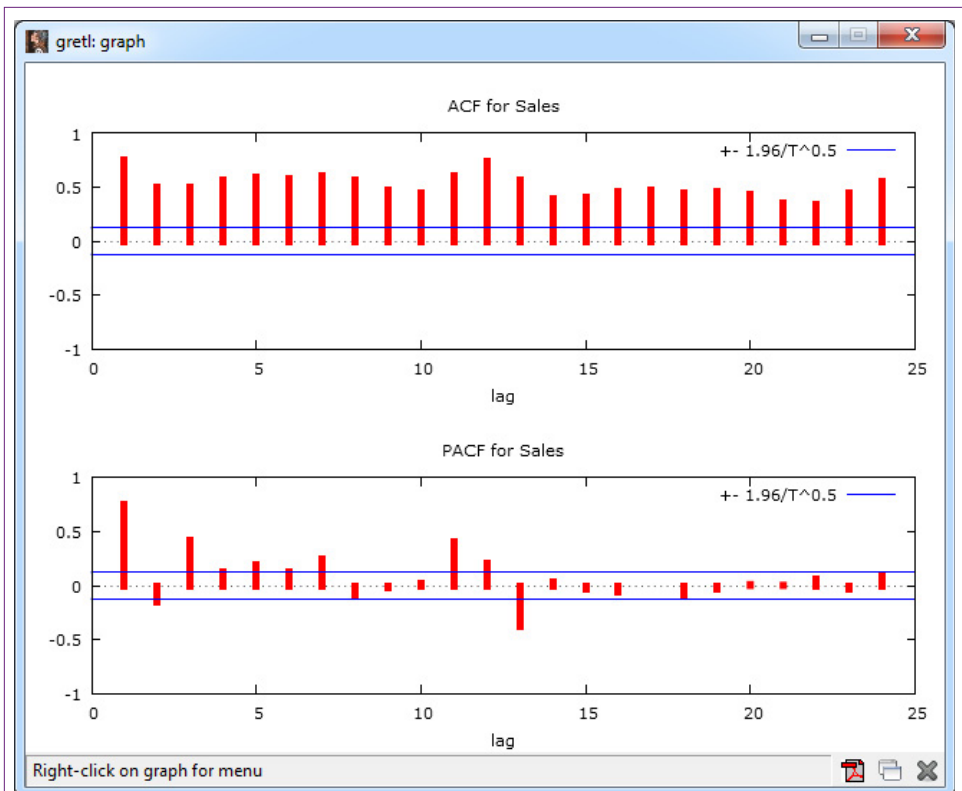
The main goal of traditional ARIMA is to identify the patterns inherent within a time series set of data. The patterns are defined in terms of auto-regressive terms that transfer information from prior values of the series and moving average coefficients that transfer data from prior values of variations—particularly seasonal variations.

The data consists of a simple series that is organized by time. Each row represents one time period and the time periods must be uniform (days, weeks, months, quarters, years, and so on). Most tools allow the addition of a column that holds the time definitions, making it possible to sort the series correctly (early to later). Typically, the data is created from a database query or OLAP cube because it represents subtotals by time period.

## Tools

To illustrate the process, create a SQL query for the Rolling Thunder Bicycle company that computes total sales by month:

```
SELECT YEAR(OrderDate)*100+MONTH(OrderDate) As YearMonth,
SUM(SalePrice) As Sales
FROM Bicycle
GROUP BY YEAR(OrderDate)*100+MONTH(OrderDate)
ORDER BY 1
```

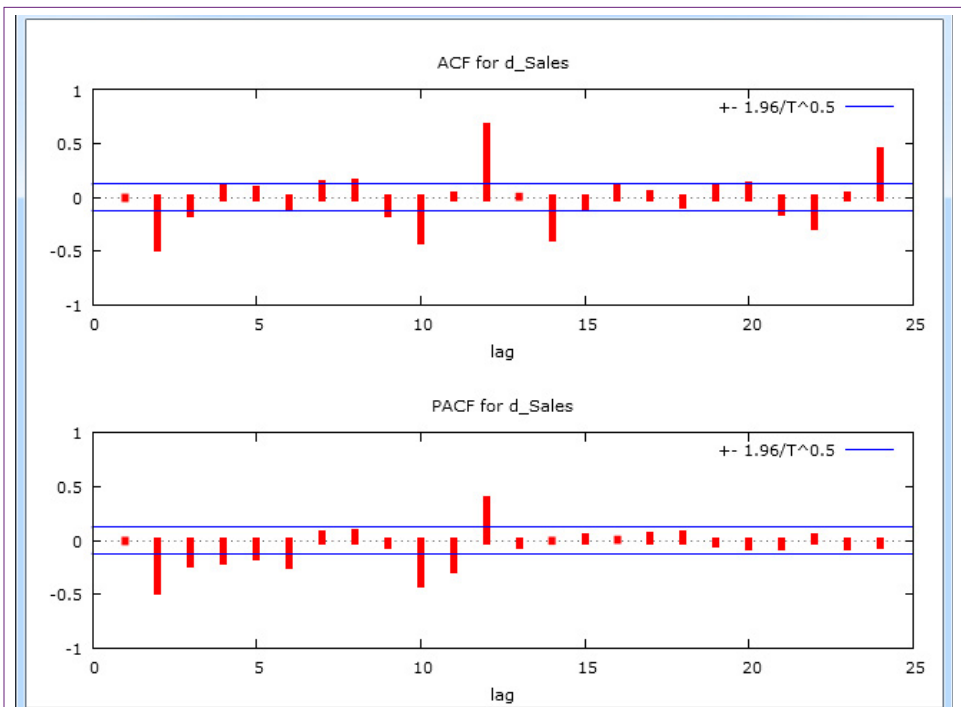


**Figure 8.17**

Initial correlogram. Notice the strong peaks in the PACF suggesting some important auto-regressive terms. But the ACF does not die down or cut off so the series is not stationary and needs to be differenced before trying to estimate the ARIMA model.

In SQL Server Management Studio, the results can be saved as a CSV file by right-clicking the query results and choosing the Save As option. You might want to use WordPad to add the column names to the top of the file. A CSV file can be imported into a standard tool that supports ARIMA such as gretl, which is free and relatively easy to use. The gretl package does not use the YearMonth column directly, so you need to specify that the file contains time series data that is monthly and begins in 1994:01. All of these options are specified when the file is opened.

One of the first steps to take with any times series analysis is to plot the data against time. It is critical to look for trends, but just exploring the data visually helps identify possible seasonal patterns. Figure 8.16 shows the plot for Rolling Thunder. First, notice the upward trend. This trend is good for business, but will need to be removed to analyze with ARMA. The trend appears to be linear, so a single differencing should solve the problem. Second, notice the potential seasonal peaks. Exploring the detailed data should show relatively higher sales in the last couple months of the year (holiday sales), plus more sales in the spring months. The five high peaks shown in the data will likely be hard to predict. They are probably related to economic or bicycle issues (such as the introduction of full suspension mountain bikes).



**Figure 8.18**

ACF and PACF for differenced data. Series appears to be stationary. The PACF dies down with additional peaks at 10 and 12. The ACF has peaks at lags 2, 8, 10, 12, and 14. The model leans towards MA, but has a complex structure.

The correlogram shown in Figure 8.17 reveals similar information to the raw data plot. Because the ACF does not die down or cut off, the series is not stationary and cannot be estimated with the raw data. At least one difference will be needed to remove the linear trend. The differencing can be handled directly within the ARIMA tool. However, it would be useful to look at the new correlogram first to see if a single difference is going to be sufficient. More importantly, the ACF and PACF based on the differenced data are needed to provide initial guesses for the AR and MA lag structure.

Gretl makes it easy to define a new series, so use Add/Series difference or create: `Sales2=diff(Sales)`. Figure 8.18 shows the ACF and PACF for the new, differenced data. The data for the differenced series should also be plotted to ensure the trend has been removed. The PACF appears to die down after a few lags, but has some additional peaks at lags 10 and 12. The ACF has interesting peaks at lags 2, 8, 10, 12, and 14. The overall structure seems to be relatively complicated. It seems to lean towards an MA structure, with important lags specified by the ACF, but there are likely to be a couple of important AR terms as well—indicated by the PACF peaks. The peaks at 12 indicate important annual/seasonal elements.

## Results

Some of the lag values are difficult to explain, but the first pass model should probably be ARIMA(4, 1, 3) with an additional MA coefficient estimated for a

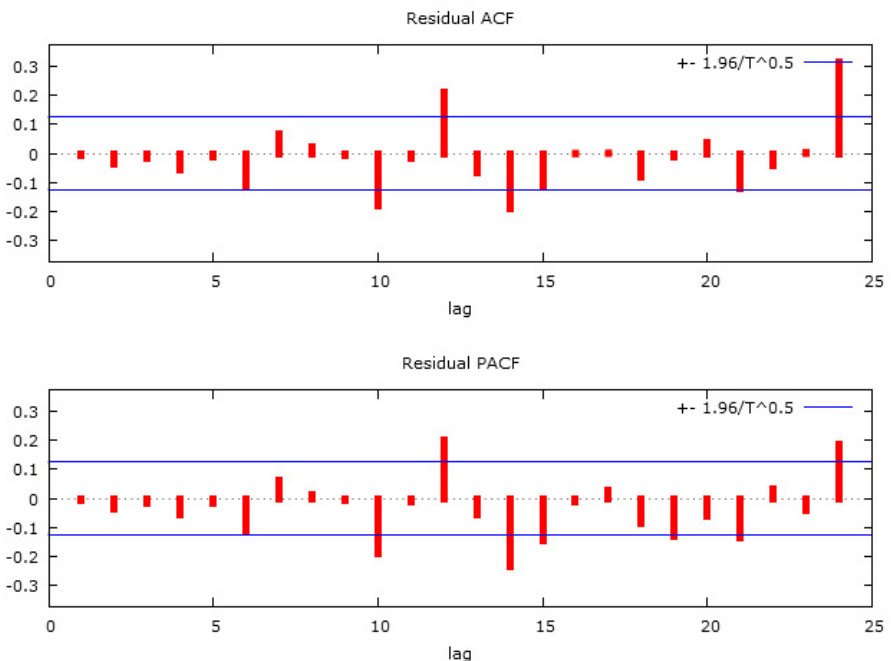
	Coeff.	Std. Err.	Z	P-value	
const	6607.7	7364.5	0.8972	0.3696	
phi_1	-0.1559	0.0991	-1.5730	0.1157	
phi_2	-0.6170	0.0981	-6.2910	0.0000	***
phi_3	-0.0234	0.0777	-0.3009	0.7635	
phi_4	-0.2196	0.0764	-2.8750	0.0040	***
theta_1	-0.0594	0.0794	-0.7484	0.4542	
theta_2	0.0363	0.0734	0.0494	0.6213	
theta_3	-0.4407	0.0659	-6.6850	0.0000	***
theta_12	0.5127	0.0517	9.9160	0.0000	***
Akaike criterion:	6930.493				
Schwarz criterion:	6965.747				
Hannan-Quinn:	6944.680				

**Figure 8.19**

Results from initial model ARIMA(4, 1, 3) and MA 12. Notice the significance of the AR 2 and 4 lags and the MA 3 and 12 lags.

**Figure 8.20**

ACF and PACF for residuals. Major peaks have been reduced, but both the ACF and PACF show statistically significant effects at lags 10, 12, and 14.



	Coeff	std. error	t-ratio	p-value	
const	5589.04	757.496	7.378	0.0000	***
phi_1	0.0255	0.0795	0.3208	0.7484	
phi_2	-0.2282	0.0779	-2.9290	0.0034	***
phi_3	-0.0000	0.0626	-0.0005	0.9996	
phi_4	-0.0917	0.0504	-1.8180	0.0690	
phi_12	0.6707	0.0492	13.640	0.0000	***
theta_1	-0.3340	0.1031	-3.2410	0.0012	***
theta_2	-0.2690	0.1139	-2.3610	0.0182	**
theta_3	-0.2348	0.0963	-2.4370	0.0148	**
theta_10	-0.1499	0.0567	-2.6410	0.0083	***
theta_14	-0.0124	0.0611	-0.2032	0.8390	
Akaike criterion:		6863.546			
Schwarz criterion:		6905.851			
Hannan-Quinn:		6880.571			

**Figure 8.21**

Results from new model ARIMA(1 2 3 4 12, 1, 1 2 3 10 14 ). Notice the new terms are significant and the diagnostic measures have improved.

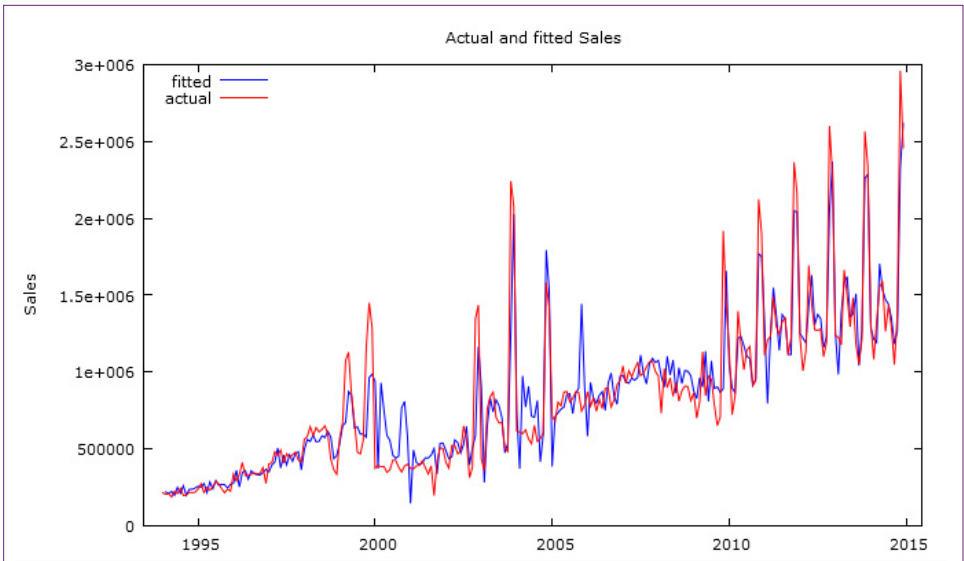
12-period lag. Figure 8.19 shows the results of that estimation. Notice the significance of the AR 2 and 4 lags along with the MA 3 and 12 lags. Some basic model evaluation numbers are also reported. They are useful when comparing this base model to other variations.

Is this model good enough now? The easiest way to answer that important question is to look at the correlogram of the residuals. Figure 8.20 shows the ACF and PACF for those residuals. First, notice that the major peaks have at least been reduced. However, both the ACF and PACF have effects at lags 10 and 14 that are significantly different from zero. Both are also negative, so they are likely to be secondary effects from the seasonal data. That is, people appear to buy more bicycles at a couple of times in the year (holidays and spring), and consequently, they pull back even further than expected just before and just after these two big events. Because the events are appearing in both diagrams, it is not clear if they should be estimate as AR or MA terms.

As a first pass you can try estimating the additional lag effects as MA terms but the estimation might not succeed. Also, the lag at 12 should probably be modeled as an AR term, bringing the full seasonal information forward. So the new model is ARIMA(1 2 3 4 12, 1, 1 2 3 10 14). Figure 8.21 shows the estimated coefficients. Notice the new lag effects are significant (except for MA 14) and the diagnostic values have improved from the original model. Consequently, this model is better than the initial one. Examine the residual correlogram to double check—no significant effects remain in the residual ACF or PACF. You could try running the model with the 14-lag in the AR side instead of MA to see if it makes a difference. (It does not so it could just be dropped, which slightly improves the model evaluation statistics.)

Once the model has captured the main effects, it should be compared to the actual data. Figure 8.22 shows a basic time series plot of the actual values versus



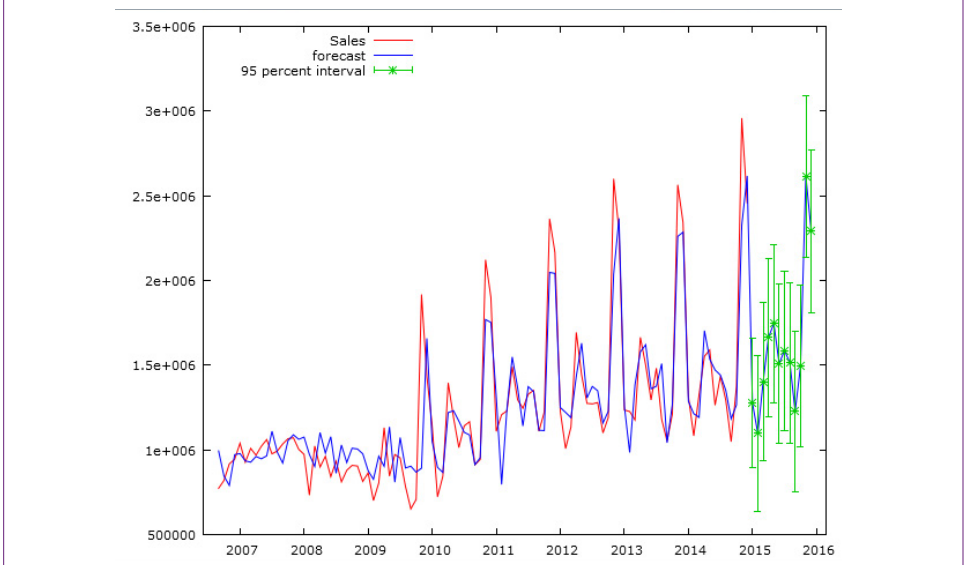


**Figure 8.22**

Plot of actual (red) versus values predicted by the model (blue). Overall the model appears close, except for an over estimation in 2001 and 2009. These drops are probably due to external factors.

**Figure 8.23**

Forecast into the future (12 months). The blue line is hard to see but it is the central wavy line. The green bars indicate the 95 percent confidence interval for the prediction.



2015:01	1276797
2015:02	1097255
2015:03	1402733
2015:04	1663312
2015:05	1746264
2015:06	1510111
2015:07	1587235
2015:08	1515373
2015:09	1228551
2015:10	1494301
2015:11	2617069
2015:12	2290510

**Figure 8.24**

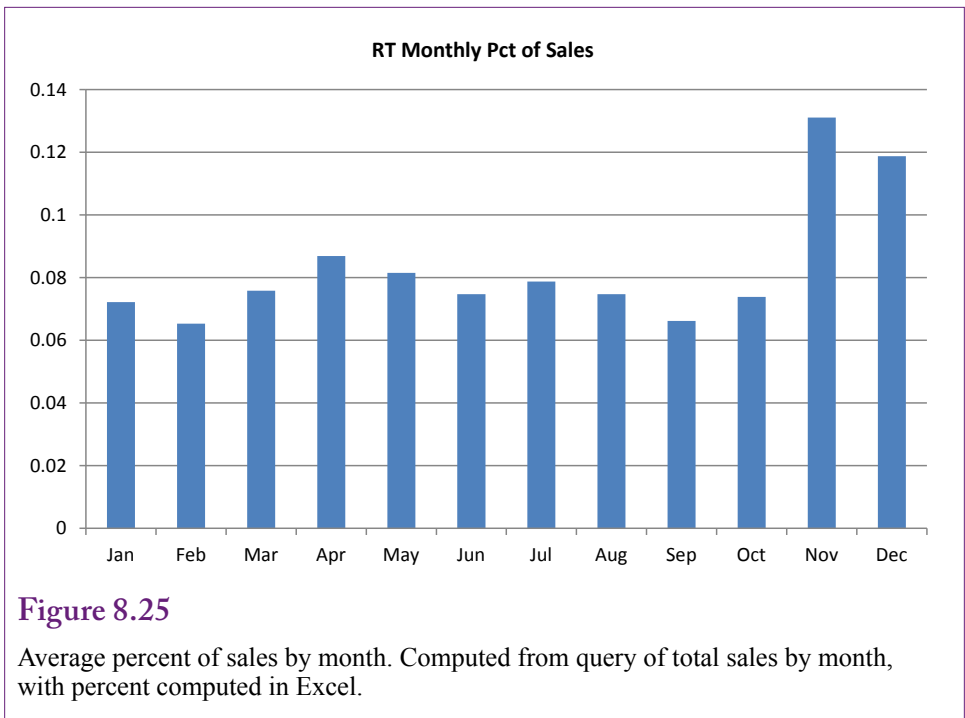
Forecast values for 2015.

the values predicted by the current model. Overall, the model prediction appears to match the actual values fairly closely—surprisingly even for the major peaks. However, the predictions for 2001 and 2009 seem to be substantially higher than the actual sales for the months in that year. These drops in actual sales is most likely due to external factors. Any guesses? Do you know history? Plausibly the attacks of 9/11 played a role, but the other major effect that year was the dot-com e-commerce crash, which also caused a slight economic recession. Similarly, the housing crash of 2008/2009 probably reduced sales for a couple of years. Remember that the ARIMA method relies solely on internal patterns. Deviations similar to this one provide interesting points to look for external events that might have influenced the data.

## Forecasts

The ARIMA model is well-designed for forecasting patterns into the future. Most ARIMA tools have options to generate forecasts and *gretl* is no exception. From the main model results, *gretl* has a menu option to generate a forecast using that model. Sticking with the default options leads to a standard forecast. Figure 8.23 shows the forecast for 12 months past the original data. The wavy blue line appears to copy the pattern over the prior three or four years with variations for seasonal effects. The green bars indicate the 95 percent confidence interval for each prediction. Notice that the range is relatively large—indicating that actual sales could be quite different from the expected forecast. Also notice that the confidence interval increases as the forecast moves further out in time. This effect is rational because more random events can affect patterns over longer periods of time. It might be tempting to discount the wide range of possibilities, but remember that the model does not evaluate external economic effects, and remember the changes missed in 2001 and 2009.

Figure 8.24 shows the actual data forecast by the model for 2015. The numbers could be used by managers to make purchasing decisions, change marketing plans, or develop long-term plans to expand or contract the business. The seasonal information is particularly useful for staffing decisions at various times through the year. The ARIMA tools also provide estimates of the standard errors; however, these numbers are quite high in this situation because of the high variability in data in the past years. These numbers will be useful to compare with forecasts created by other techniques in later sections of this chapter.



### Seasonality Evaluation

Another of the key strengths of traditional ARIMA is the visibility of the seasonal effects. These are easiest to see from the coefficients on the moving average terms. Specifically, the value of 0.6707 for the AR(12) term is positive and quite high. It indicates a strong seasonal effect—whatever happened 12 months ago is likely to repeat each year. To see the effects by month, create a query that computes the total sales by month for all of the years. Transfer the data to Excel to compute the overall total and divide to get the rough percent of sales by month. A more time-consuming approach is to compute the percentage of sales by month within each year and then compute the average of the percentages. This latter approach is more accurate and is a good exercise for the reader. Anyway, Figure 8.25 shows the primary sales months of November and December. Spring months of March, April, and May also appear to have slight gains.

Look again at the values for the MA lag coefficients. The MA coefficient are all negative. And only the AR(1) and AR(12) coefficients are positive. With monthly data, this value means strong seasonal effects exist. Now realize that the coefficients for MA(1), MA(2), and MA(3) are all negative. Yet, these are not auto-regressive coefficients, so the negative value needs to be interpreted carefully. Look again at the percentage sales values by month. Notice that months with big jumps (November and March) are preceded by months with low sales. The moving average coefficient applies to the variation within the month, not to the overall value. Hence, when the preceding three months are lower than average, the MA coefficients reveal that the sales in the target month (e.g., November) are even higher (negative coefficient times negative variation). This pattern means that consumers are essentially holding back on purchases during certain months and then buying at specific times—particularly for November sales. The negative values also mean that sales fall off after the big months.

The negative values for MA(10) and MA(14) are a little more difficult to explain. On face value, they would imply purchase decisions are made 10 and 14 months in advance. For example, customers who plan to purchase a bicycle in November 2010 have already made some decisions about that purchase in October 2009 and January 2010. The January effect of 10 months might be understandable, but the October effect takes a stretch of imagination. It could be an artifact of the data; or it could be a statistical result of the extremely low sales in October and January—note that it is small and not significantly different from zero. In any case, the important point is that the ARIMA coefficients clearly picked up these variations, but the annual percentage chart makes it easier to explain to managers.

## Microsoft Time Series Estimation

---

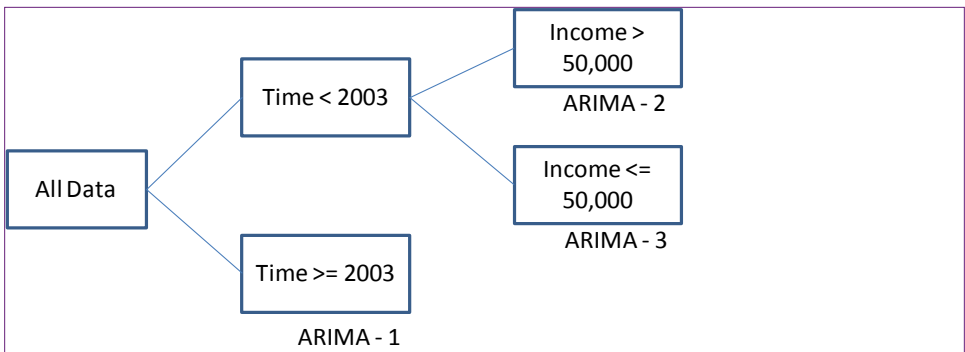
**How does the Microsoft Time Series estimation work?** With SQL Server 2008 BI, Microsoft modified its time series estimation method by adding a version of ARIMA. The tool attempts to reduce reliance on analysts by using statistical tools to guess the lag structure. More importantly, the default estimation method (MIXED) also relies on a proprietary model called **auto-regressive tree with cross prediction (ARTxp)**. ARTxp is used for immediate (1-5 period forecasts). It uses a decision tree approach to try and determine which time periods affect the outcome. It also emphasizes cross correlation, so it encourages the addition of other series that might provide information to predict the original series.

With an emphasis on prediction instead of evaluation, the Microsoft results can be challenging to interpret. However, it is easy to incorporate other attributes into the analysis. Still, as shown in the Results section, the forecast results can be highly variable depending on the model and any parameter hints.

### Goals

The Microsoft approach focuses on the prediction of the time series, hopefully with minimal intervention by the analyst. Microsoft uses two models estimated independently to forecast data. It is possible to restrict the process to a single model, and users of the enterprise version of SQL Server can control the mixture rate. However, the default is to use both models (MIXED), so it is important to understand both methods.

The ARTxp method is a decision tree approach that can be thought of as a piecewise linear regression model. The tool searches for change points that indicate a different effect on the outcome variable. Figure 8.26 shows that these change points can be created from various points within the time series or from external, cross-correlated series. Within the time series, the argument is that a time series might exhibit a different pattern while the series is declining compared to when the series is increasing. If so, the ARTxp method creates a node and estimates new model coefficients for each segment. When mixed with the ARIMA process, each leaf node has a separate ARIMA model. Forecasting entails finding the matching decision tree leaf node and applying its model. In the simple example, if the data to be forecast is earlier than 2003, and the Income level is greater than 50,000, ARIMA model 2 would be used to make one forecast. The node also contains ARTxp parameters to make a separate forecast. Depending on the mixture parameters, the two values are averaged to obtain the final forecast. By default, short-term forecasts within about 5 periods of the ending data are predominantly made with the ARTxp model. Longer-term forecasts lean more heavily on the ARIMA



**Figure 8.26**

Microsoft ARTxp goal. Find split points where a model change can improve the prediction. Mixed with ARIMA, each leaf node has a separate ARIMA estimate.

model based on a parameter that sets the weighted average. Predictions that are farther out in time rely more on the ARIMA model because it is less volatile.

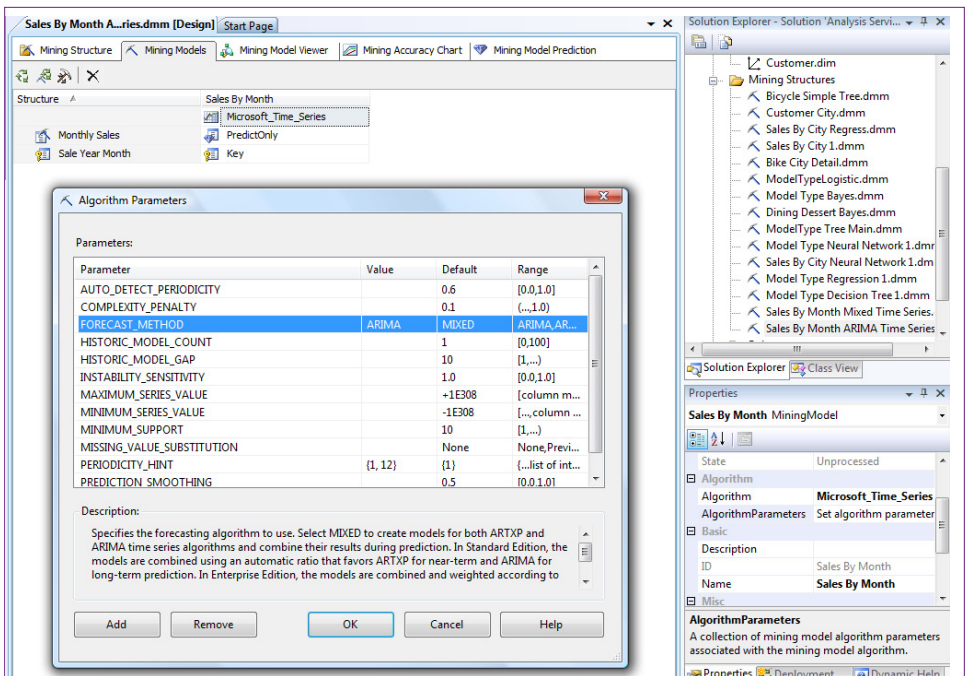
The ARIMA method used by Microsoft is also slightly different from the traditional model. As proposed by Box-Jenkins, seasonal models can be made easier to estimate with a multiplicative seasonal structure. The seasonality ( $s$ ) first needs to be defined. For example, it would be 12 in a monthly model and 4 in a quarterly model to estimate annual effects. Then, a separate **seasonal ARIMA (SARIMA)** model is estimated based on lags for the seasonal data, specified separately as  $P, D, Q$ . The seasonal lags ( $P$  and  $Q$ ) are defined as multiples of the seasonality. Typically, they are small values (1 or 2 at most). For example, a value of  $P=1$  would generate a **seasonal auto-regressive (SAR)** estimate of one season (or 12 months). A value of  $Q=2$  would estimate a **seasonal moving average SMA(2)** lag equal to 24 months. Differences are made at the seasonal level, so  $D=1$  leads to subtracting all values 12-months apart. It should never be necessary to go above 1 difference for seasonal components. The model is typically written as:

$$\text{ARIMA}(p, d, q) \times (P, D, Q)$$

The model is multiplicative, where the seasonal terms are multiplied by the unit terms. Consequently, the seasonal terms should be kept small or the model ends up with a huge number of coefficients to be estimated. A common seasonal model is based on two levels of exponential smoothing:

$$\text{ARIMA}(0, 1, 1) \times (0, 1, 1)$$

Microsoft also complicates the results somewhat by estimating up to 8 times the number of lags specified. The high-end value of 8 is often used for the unit ARIMA, and rarely for the higher-order seasonal component. That is, if periodicity is specified as MA(1) or ARIMA(0, 1, 1), the tool will actually estimate coefficients for MA(1), MA(2), ... MA(8). Consequently, it is seldom necessary to specify a base ARIMA with a parameter greater than 1 because the other values will be estimated anyway. Most commonly, if you know that the data is monthly, you would specify the periodicity parameter as: {1, 12}.



**Figure 8.27**

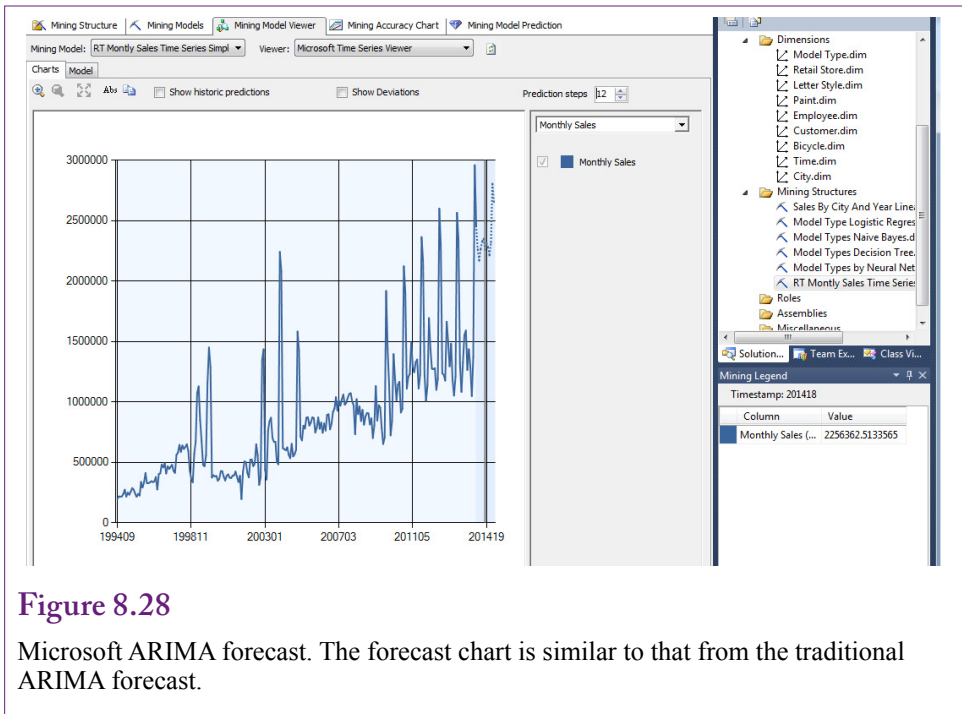
Microsoft time series set to ARIMA. Select Microsoft Time Series in the Mining Model tab. In the properties window, click the button under the Algorithm Parameters option. Enter ARIMA for the Forecast\_Method and {1, 12} as the Periodicity\_Hint.

## Data

Time series data for the Microsoft model is usually computed with a GROUP BY query or the subtotals are pulled directly from an OLAP cube. The data must have a time key column that uniquely identifies the rows so they can be sorted in the proper order. The column should be numeric but the data values are not important. A common approach is to combine year and month numbers, or quarter numbers for quarterly data). A named query can be created quickly. The query for Rolling Thunder Bicycles is:

```
SELECT YEAR(OrderDate)*100 + MONTH(OrderDate) AS
SaleYearMonth, SUM(SalePrice) AS MonthlySales
FROM dbo.Bicycle
GROUP BY YEAR(OrderDate) * 100 + MONTH(OrderDate)
```

Any additional attributes that will be used for cross correlation must match the time key data. If the data is imported, it should be imported with a similar key column so the named query can use a JOIN to match the values correctly. Also, remember that there should be no missing data. Microsoft Time Series does offer a parameter with several options to control how missing data values will be replaced. However, even this option should not be used if there are too many missing observations. Instead, aggregate the data to a higher level.



## Tools

For comparison to traditional ARIMA, try running a model with just the ARIMA component first. The ARTxp or MIXED version that blends in Microsoft’s proprietary tree method can be run second—it complicates the interpretation of the results.

Create a new mining structure using Microsoft Time Series and select the data source with the named query containing the sales totals by month. Follow the wizard to create the model. Choose the MonthlySales as the predictable column. It does not need to be selected as an Input column. The SaleYearMonth column should automatically be selected as a key time column. As shown in Figure 8.27, the ARIMA specification is made as a parameter change. On the Mining Models tab, select the Microsoft\_Time\_Series entry. Switch to the parameters window in the lower-right corner. Highlight the Algorithm Parameters entry and click the ellipses button to open the parameter edit window. Find the Forecast\_Method row and enter ARIMA as the Value. Find the periodicity hint and enter {1, 12} to indicate the series contains monthly data. Process and Browse the model.

Figure 8.28 shows the results can be displayed as a chart. The default forecast is 5 periods, but it can be increased by setting the number of prediction steps to 12. At first glance, the chart forecast appears similar to that created with traditional ARIMA. Switch to the Model view to see the estimated coefficients.

Because the model was restricted to simple ARIMA, only one tree node is displayed in the model view. Select that node and examine the Mining Legend. The ARIMA model is displayed in the legend, but it is difficult to read. Figure 8.29 shows the model reformatted and labeled by hand. The multiplicative seasonal component (P, D, Q) is listed after the “X” symbol. It is labeled with (12) to indicate the seasonal length and the algorithm determined that one was estimated



```

ARIMA (
    {1,0.23311},          AR(1)
    1,                   difference
    {1,-0.14026})       MA(1)
X (
    {1,-0.39775,5.34085E-02}, SAR(12)
    0,                   S difference
    {1,-6.72065E-02})  SMA(12)
Intercept:7225.834

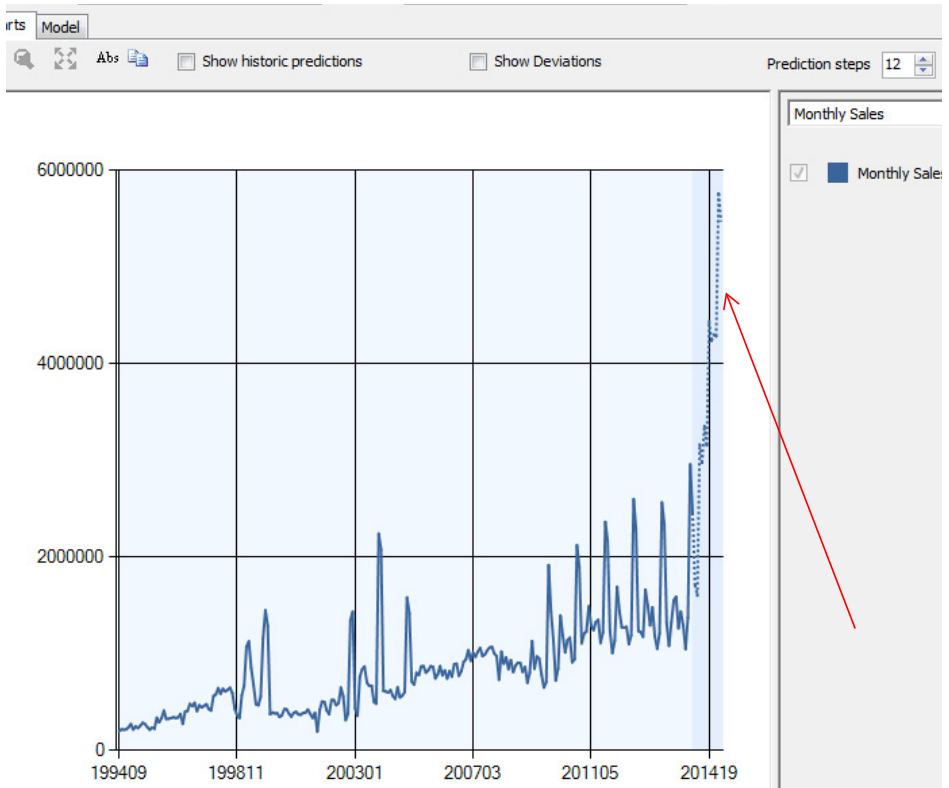
```

**Figure 8.29**

Microsoft ARIMA coefficients. The legend display is hard to read. The coefficients here are displayed on separate lines and labeled by hand.

**Figure 8.30**

Microsoft MIXED forecast. The model predicts a large drop in sales for the next year—both immediately and throughout the year. Based solely on internal sales, this forecast seems unrealistic.



ARIMA (	
{1,-1.45534E-02},	AR(1)
1,	Diff.
{1,-0.74364}) X	MA(1)
({1,0.27151,	SAR (6)
4.86931661E-02},	SAR(12)
0,	Diff.
{1,0.767121})(6) X	SMA(6)
({1,0.10754,	SAR(4)
0.09174323},	SAR(8)
1,	Diff.
{1,2.832889E-02})(4)	SMA(4)
Intercept:7293.6173	

**Figure 8.31**

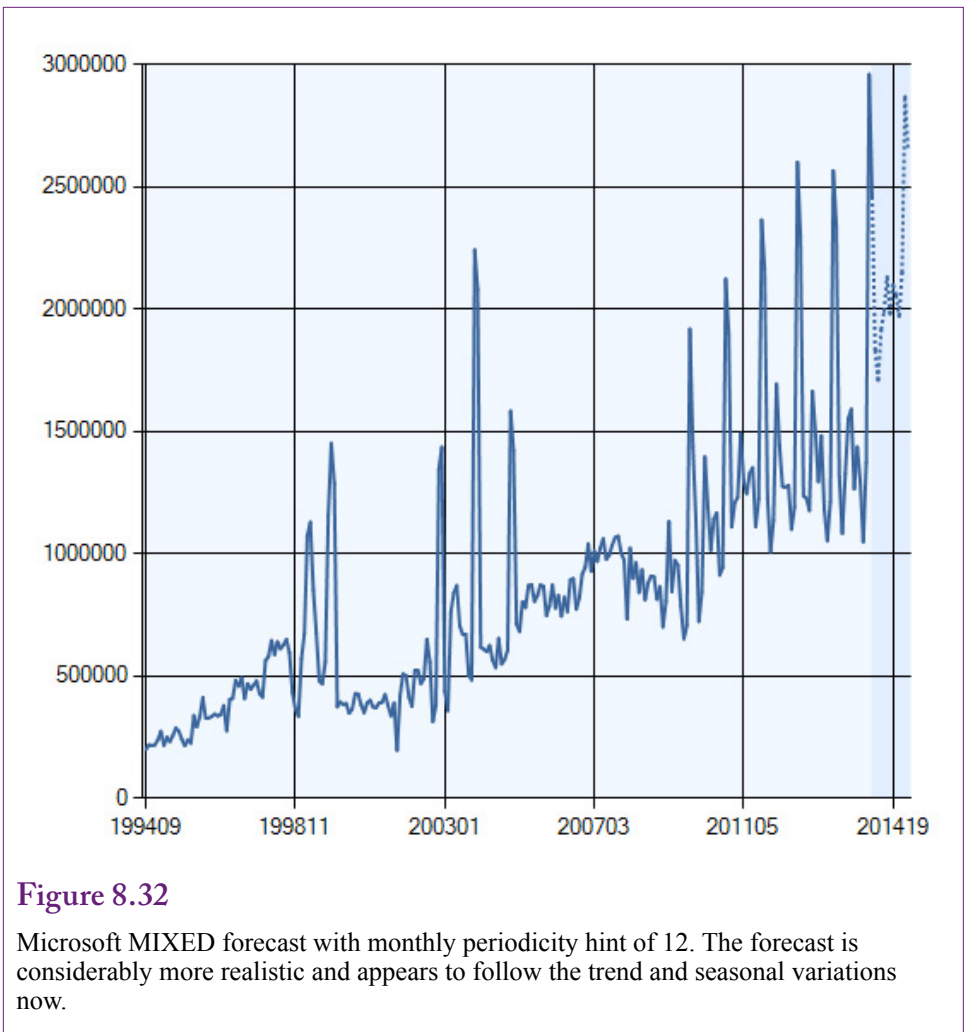
ARIMA coefficients for root node Note the seasonal auto-regressive terms. The structure of the model was selected automatically to use a 4 and 6 month seasonality lag.

at 12 months. The coefficients are different from those computed with the traditional ARIMA model. Note that *gretl* can estimate a seasonal ARIMA model, and when it is specified similar to this version, the coefficient values are closer to these results.

The goal is not to see if Microsoft ARIMA can produce identical results to traditional ARIMA tools. The point of the example is to ensure that the model can be run and the results retrieved and interpreted. The more common method of using Microsoft Time Series is to stick with the default model: Mixed and let the tool choose the lag structure. In other words, the least amount of supervision by the analyst.

Create a new mining structure, choose the same data source and follow the default prompts to create a new Time Series model based on the mixed ARTxp/ARIMA process. Figure 8.30 shows the result as a chart. Even before checking the coefficients and estimates look at the forecast for the next 12 months. The model predicts that sales are going to jump precipitously—even continuing through the year. Clearly, an outcome that predicts a three-fold increase in sales in a couple of months is unlikely to be realistic—based only on the data available from this internal list. Something is wrong with the model. It is possible to dig into the model in more detail, but the simplest solution is to remember that all of the default model parameters were used. So the biggest problem is that the system does not know that the data values are monthly.

To see what is happening, look at the Model structure which is displayed as a tree diagram. Select the root node (or one of the end nodes) to see the model for that slice of data. Although cut off in the figure, the ARIMA coefficients can be found by scrolling the Mining Legend. Figure 8.31 shows the values for the same future node. First, check the seasonal terms and recognize that the algorithm guessed at SAR(6) and SAR(4) which covers half-year, and slightly more than quarterly effects; but it did not estimate any annual effects. Remember that the algorithm does not know the series contains monthly data.



The answer is to tell the tool that the data has 12 months in the year. Build a new time series model. After the wizard has finished, open the Mining Models tab and select the Microsoft\_Time\_Series entry. Open the Properties tab and click the ellipses button on the entry for Algorithm Parameters. Scroll to find the Periodicity\_Hint and enter a value of {1, 12} to specify monthly data. This parameter is the only thing that will be changed from the default specification. Process and Browse the model to see the results.

## Results

Figure 8.32 shows the charted forecast results for the slightly modified model. Compare the forecast to the one in Figure 8.30 computed with the default periodicity selected automatically. Telling the algorithm that the main series holds monthly data made a huge improvement in the forecast. So you should always plan on specifying the overall periodicity of the series. Of course, if you enter values that do not match the data, the results are likely to be even worse.

```

ARIMA (
    {1, 0.23311},    AR(1)
    1,              diff.
    {1,-0.14026})  MA(1)
X (
    {1,-0.39775,    SAR(12)
    0.0534085},    SAR(24)
    0,              diff.
    {1,-0.06721})(12) SMA(12)
Intercept:7225.8342

```

**Figure 8.33**

ARIMA coefficients for the root node with the periodicity hint. Note the new seasonal terms for 12 months and 24 months.

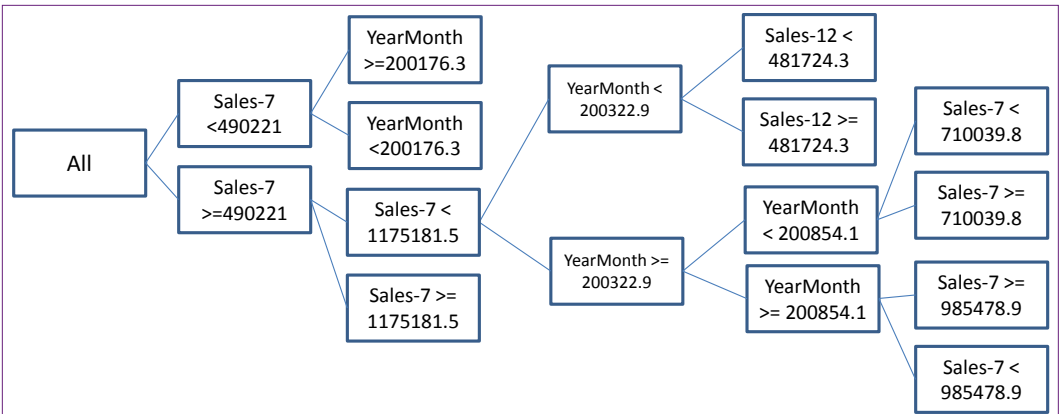
The difference lies in the ARIMA model—which did use the periodicity hint. Figure 8.33 shows the new coefficients for the root node displayed from the default model. Because of the periodicity hint of monthly, the new ARIMA model contains seasonal moving average terms for 12 and 24 months. The misleading auto-regressive values for lags 4 and 6 months are gone. Why is this model better—in the sense that the forecast does not go crazy? First, notice that the SMA term is negative in this model, compared to the positive values for the SMA terms in the initial model. This negative value provides a damping effect. The same holds true for the SAR terms. The main AR(1) term is positive and strong in this model, which would lead to higher forecast values, but the other terms tend to dampen the effects.

### ARTxp Model

The ARIMA results are actually just a small portion of the results from the Microsoft time series model. They are a useful starting point because they are similar to the traditional ARIMA model. However, the Microsoft approach downplays the role of the ARIMA model and focuses on decision-tree approach. As shown in Figure 8.34, the tool creates a tree structure by splitting nodes where it is possible to improve the results. At this point, the model contains only the key column (YearMonth) and the data column (Sales). Still, the tool found several points that justified splitting the model. However, the tree is stronger when additional data is added. This mechanism makes it relatively easy to add correlated series including economic data.

The ARIMA model coefficients remain the same for each node, but the forecast value is modified within each node based on the other terms. Figure 8.35 shows that each node has a separate equation. For example, for the node on the right side (Sales-7 < 710039.8) the additional equation is:

$$\text{Monthly Sales} = 395688.159 + 0.499 * \text{Monthly Sales}(-12)$$



**Figure 8.34**

ARTxp decision tree results. The tool searches the associated data for breakpoints where the model results can be improved by adding a node to a tree branch. This model has only the key column (YearMonth) and the sales data itself.

**Figure 8.35**

Node differences. Select a node to see the equation for that item on the tree. The ARIMA remains unchanged but each node has an additional equation.

Term	Coefficient	Histogram
Monthly Sales(-12)	0.499	

Monthly Sales = 395688.159 + 0.499 \* Monthly Sales(-12)

ARIMA equation:  
 ARIMA ((1,0.23311204599679),1,{1,-0.140261553333308}) X ((1,-0.397745131343896,5.34084824481309E-02),0,{1,-6.72065385064266E-02})(12)  
 Intercept:7225.83418153909

Time	Sales	StDev
201413	1831835	115390
201414	1709642	113679
201415	1914923	110646
201416	1991393	77833
201417	2130782	105124
201418	1980396	102778
201419	2098216	100685
201420	2056840	102464
201421	1971231	106954
201422	2148194	91530
201423	2868532	96244
201424	2647986	93432

**Figure 8.36**

Forecast values formatted with Excel. Note the Time data is useful only for sorting.

## Forecasts

The forecasting process with Microsoft Time Series is different than in the other tools. In particular, a time series forecast does not need source data, so there is no need to set up a table of values, or to enter data for a singleton query. If only one or two values are needed, the forecast chart displays the graph and provides a way to obtain one value at a time. Simply select a point on the chart with the mouse and the corresponding predicted value is displayed in a table on the right. However, this process is cumbersome for multiple values.

To obtain a set of predicted values, and to get their standard deviations, you need to enter a special MDX query. This query needs to be entered in the BI processor, not in the SQL Server query processor. Switch to the Mining Model Prediction tab. Click the icon to switch to SQL, which opens an edit window in the bottom half of the screen. The simple query to get the predictions uses the **PredictTimeSeries** function:

```
SELECT FLATTENED PredictTimeSeries([Monthly Sales], 12) As
Forecast
FROM [RT Monthly Sales Time Series ARTxp 12]
```

The query is straightforward, with the only drawback of having to type the table name and column name by hand. Be sure to specify the FROM section with the name of the model you are using in brackets. The PredictTimeSeries function needs only two parameters: The column name and the number of periods (e.g., 12) to be forecast. Forecasting a single value is always risky—it is always good to include the standard deviation of the forecast value to obtain the accuracy of the forecast. This query is slightly more complex because the standard deviation needs to be computed for each forecast value:

```
SELECT FLATTENED
(SELECT *, PredictStdev([Monthly Sales]) As [StDev]
FROM PredictTimeSeries([Monthly Sales], 12))
As Forecast
FROM [RT Monthly Sales Time Series ARTxp 12]
```

Figure 8.36 shows the output from the query. The actual output is a little messier. These values were formatted in Excel to make them easier to read. Notice that the Time variable is useful only for sorting. To restore the business meaning, the numbers would have to be manually changed to 201501, 201502, and so on. The standard deviation values can be used to compute 95 percent confidence intervals to indicate the accuracy of the forecast.

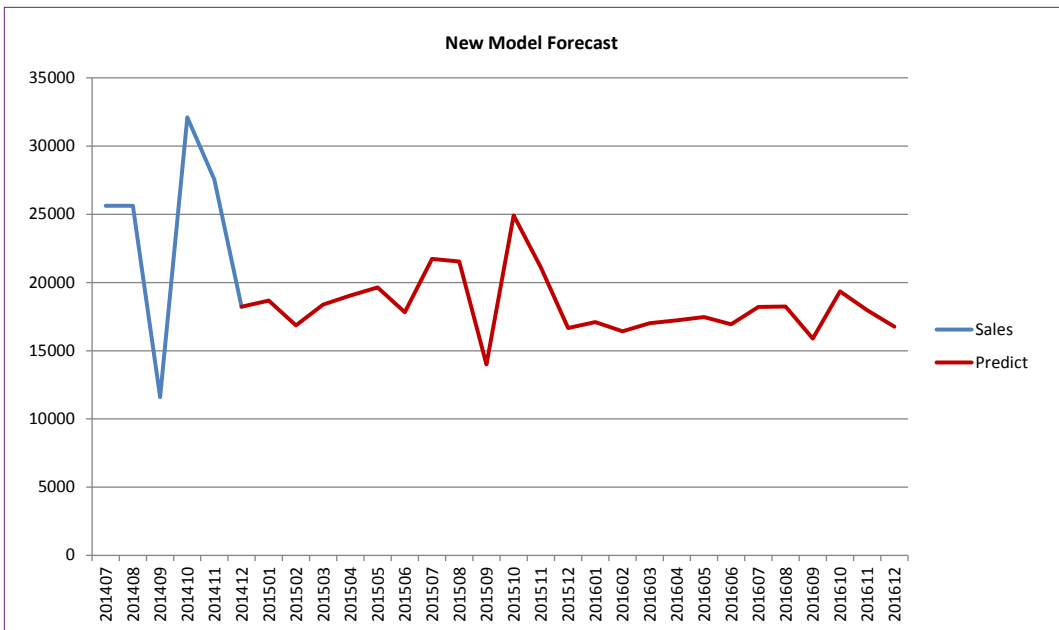
The MDX language includes additional functions to return the variance (PredictVariance) and the decision tree node that matches the conditions (PredictNodeID). The language also has powerful tools to create forecasts by changing some of the input data. An even more interesting tool is to predict a new short set of data based on the patterns established from the longer series. For instance, what if RT managers have introduced a new model type and have sales data for only a few months. This data would not be sufficient to establish a useful pattern on its own. Instead, the sales patterns from the entire company can be applied to this new data. That is, take the trend and seasonal information from the overall sales and apply it to the data from the new model. The syntax is a little tricky, but it is such a useful trick it is worth examining:

```
SELECT FLATTENED
    PredictTimeSeries([Monthly Sales], 24, REPLACE_MODEL_
CASES)
FROM [RT Monthly Sales Time Series ARTxp 12]
PREDICTION JOIN
    (SELECT 201407 AS [Sale Year Month], 25621 As [New Model]
    UNION SELECT 201408 AS [Sale Year Month], 25621 As [New
Model]
    UNION SELECT 201409 AS [Sale Year Month], 11592 As [New
Model]
    UNION SELECT 201410 AS [Sale Year Month], 32119 As [New
Model]
    UNION SELECT 201411 AS [Sale Year Month], 27559 As [New
Model]
    UNION SELECT 201412 AS [Sale Year Month], 18225 As [New
Model]
    ) AS t
ON [RT Monthly Sales Time Series ARTxp 12].[Sale Year
Month] = t.[Sale Year Month]
    AND [RT Monthly Sales Time Series ARTxp 12].[Monthly
Sales] = t.[New Model]
```

In this query, the PredictTimeSeries function specifies a prediction of 24 periods and it uses the REPLACE\_MODEL\_CASES parameter to ensure new data is used as the starting point of the prediction. This new data is provided manually by the PREDICTION JOIN subquery. Notice that the new data includes the Sale-YearMonth entries that match up to the existing values for the end of 2014.

Figure 8.37 shows the results of the forecast. The data was copied and pasted into Excel to generate the chart. Notice that the new model sales data form the starting point. The forecast applies the overall seasonal pattern to that initial data and generates a forecast specifically for the new model. Of course, it is possible that the new model sales will be radically different—particularly if a special advertising campaign is created. Nonetheless, the ability to apply the overall seasonal pattern to the small set of data is a useful technique.





**Figure 8.37**

Existing sales pattern applied to a new model. The new model starting values were entered manually but then the existing pattern is applied to yield a forecast that follows the seasonal patterns.

### Seasonality Evaluation

The ARTxp model provides only a limited amount of information about seasonality. Recall that the node results included an AR term for a 12-month lag, indicating that a portion of sales from 12 months back will automatically be included in a prediction for the current month.

The ARIMA model is more powerful at identifying seasonal patterns—much the same way it is used in the traditional regression approach. The objective is to evaluate the coefficients on the various seasonal terms. Because Microsoft time series relies on the seasonal model, the analyst can focus on the seasonal P and Q terms. In the final monthly model estimated here, the results returned two seasonal terms:  $SAR(12) = -0.398$ ,  $SAR(24) = 0.053$ , and  $SMA(12) = -0.067$ . The 12-month term represents a one-year lag and indicates that a seasonal pattern does exist. Each year, sales in a specific month will be determined somewhat by the sales that occurred one year ago in that month. The interesting twist is that the sales will be negatively impacted by the sales from one year ago in that month. But the positive  $SMA(24)$  coefficient buffers that value by adding sales from two years ago.

### Cross Correlation and Linear Regression

**How can the effects of other series be incorporated into the prediction?** The question of multiple time series falls into the subject heading of dynamic systems. It is a complex topic that can involve detailed mathematical

models. Few data mining packages handle the complex tools by default, and if you really need these high-end tools, it would be safest to consult with an expert. For example, the ARMA process has been extended to vectors. Data sets with multiple time series can also run into issues involving simultaneous equations, and problems with fitting multiple models to the data. So, at least read all of the literature on a specific tool before attempting to use it. On the other hand, it is possible to get basic estimates for models that incorporate time series and multiple input variables. Just remember that these tools only scratch the surface, and this section is merely an introduction.

## Goals

The basic goal is to predict future values for a time series. The time series can exhibit trend, auto-regressive, and seasonal effects. It can also be affected by other attributes. Economic models are fairly common examples. From a business perspective, sales can be heavily dependent on overall economic data. For example, in economic terms, bicycle sales are most likely a normal good—which means that as consumer income increases, consumers will potentially buy more high-end bicycles. Certainly, if an economy dips into a recession, sales of high-end bicycles are likely to decline. If it is possible to predict overall economic trends, this information could be used to improve forecasts for specific products, such as custom bicycles. Of course, any external attributes that are added to the model must be predictable by some process. Fortunately, most broad economic measures are important to many people, so it is possible to find short-run forecasts by experts. These forecasts are not always perfect, but they are available and are probably as good as possible given the random nature of the world.

As usual, a secondary goal of including additional attributes is simply to learn more about the data. If some series do affect sales, just that knowledge can help improve decision making. Similarly, if some attributes do not significantly affect the outcome, many decision problems become simplified.

## Data

The predictable variable should remain the same as for basic time series analysis: A single column of observations at regular time intervals. It will usually be necessary to include a key column that defines the sequence of the time data. As with simpler models, this column is often a composite number consisting of the year and month for monthly data. Quarter or day numbers can be used for different intervals. If linear regression is going to be used to analyze the data, the data set should also contain simple columns containing the year and the month number. These columns will become separate attributes to measure trend and seasonal effects. Any data for cross correlation should be held in separate columns—being careful to match the time increments of the original series.

Economic data is often used to estimate effects on internal data. Standard economic series are readily available from government Web sites. Several U.S. government agencies provide tools to find and download economic data. A few international organizations provide data on other nations, but the U.S. data is available free of charge—some organizations charge fees to download data. The U.S. government maintains a Web site that serves as an index to the various agencies. To find data, start at [www.fedstats.gov](http://www.fedstats.gov). One catch to federal data is that it is often seasonally adjusted. It can be difficult to find raw data for some government series. In many cases, seasonally adjusted data will be acceptable for cross

correlation analysis. For example, comparing national income to sales will work even when the economic income data are seasonally adjusted. The sales data will already contain any seasonal variation which can be measured, so any correlation with the economic income data would represent basic income effects.

For the Rolling Thunder Bicycle sales example, it is possible to download disposable personal income and a price index (inflation) measure. Both of these measures are available as monthly data, and both are seasonally adjusted. To be used with Microsoft BI, the data series need to be downloaded and imported in to SQL Server. It is often easier to copy-and-paste the data into Excel first and then save the two series as a CSV file which can be imported into a new table in SQL Server. While the data sits in an Excel worksheet, you can add the time stamp information for year and month. Later, this column makes it possible to join the economic data to a query that computes total sales by month. The U.S. Bureau of Economic Analysis reports standard national income and product account (NIPA) data. For the bicycle sales case, Table 2.6 reports Disposable Personal Income as monthly data. Table 2.8.4 reports the personal consumption expenditures (PCE) price indexes on a monthly basis. Both series can be found at the [www.bea.gov/national/nipaweb](http://www.bea.gov/national/nipaweb) site. Find each data set separately, set the range for 1994/01 through 2014/12 to get the data that matches the sales for Rolling Thunder Bicycles. The data can be downloaded and saved as a CSV file or copy-and-paste the data into Excel and save the data as a CSV file. SQL Server can import a CSV file and create a new table to hold the data. Be sure to include a YearMonth column that is in the same format as the SaleYearMonth column used to compute the sub-totals of sales by month (such as 201201).

## Tools

Ignoring high-end complex tools, the most common method of incorporating cross correlation into time series analysis involves some version of linear regression, or a modified version of ARIMA. Microsoft Time Series uses the decision tree approach to emulate a regression effect. Simply including another data series as an Input attribute leads the decision tree model to look for node points where the values produce a different effect on the results. The decision tree essentially estimates a piecewise-linear model to find the node values that are significant factors.

Box and Jenkins, who defined the ARIMA approach, also defined a method to incorporate external data into the prediction. Some tools, including *gretl*, support this approach. With this approach, as with Microsoft ARTxp, the analyst simply needs to import the new data series and add the attribute as an Input variable to the model. The tools do most of the work automatically.

It is also possible to use linear regression directly to build a linear model of trend, seasonality, and cross correlation effects. The key lies in creating the correct set of X-attribute values. These attributes would be easier to create if SQL Server supported the Lag function, because a common solution involves using  $y_{t-1}$  as an input variable. Linear regression requires the most amount of supervision and setup, but it also provides detailed control over the model.

### *Microsoft ARTxp*

Adding cross correlation to a Microsoft time series model is relatively easy—once the data is imported and available for analysis. The ARTxp process was specifically designed to handle multiple input attributes. The model simply factors them into decisions about creating tree nodes as split points and the results are available in the Mining Legend.

By now, you should have downloaded and imported the Income and price index from the BEA database into SQL Server. Assume the columns are in a table called DPI with YearMonth as a primary key. The challenge now is to build a named query that (1) computes the subtotal of sales by month and (2) joins the result to the imported DPI data. The process is somewhat tricky because data sources do not handle one-to-one joins, and named queries apparently cannot reference other named queries. The complication arises because the SalesByMonth calculation must be performed first to create the SaleYearMonth column which is then joined to the YearMonth data in the DPI table. One solution to the problems would be to build the SalesByMonth as a saved view within SQL Server itself, then join the DPI table to the result either within SQL Server or in the data source. However, the entire calculation and join can be handled as a named query with the little trick of using a temporary table:

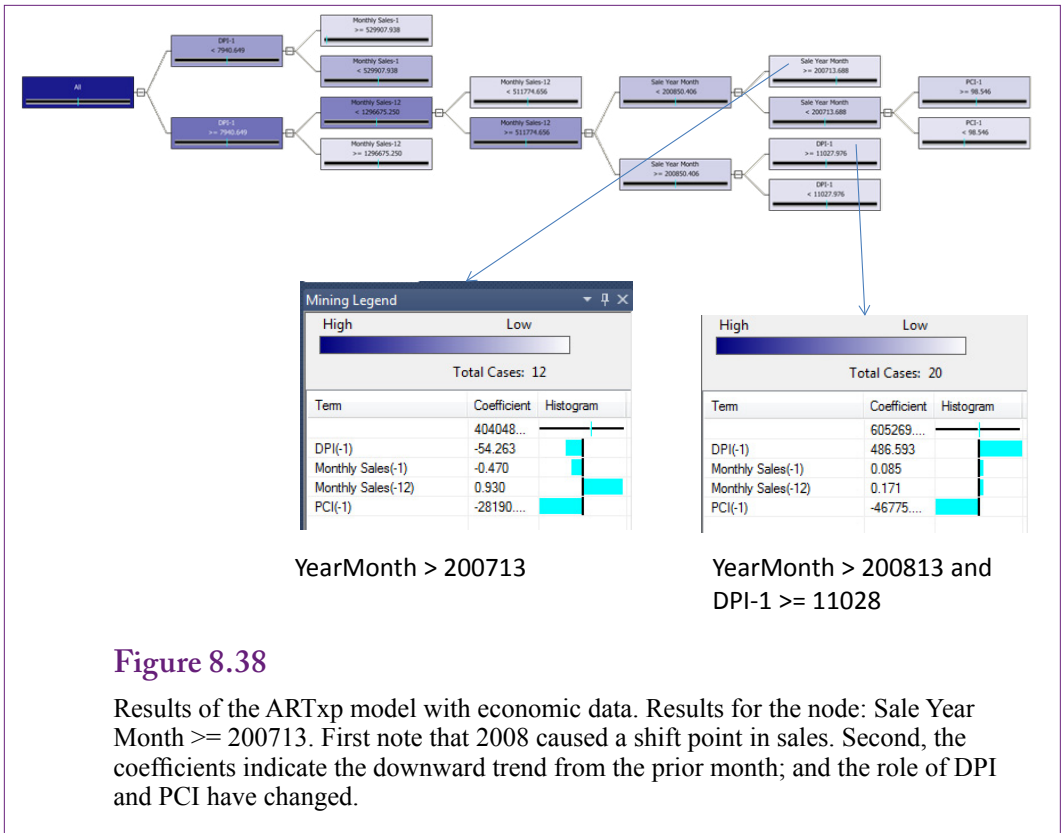
```
SELECT t.SaleYearMonth, t.MonthlySales, dbo.DPI.DPI, dbo.
DPI.PCI
FROM dbo.DPI INNER JOIN
(SELECT YEAR(OrderDate) * 100 + MONTH(OrderDate) AS
SaleYearMonth, SUM(SalePrice) AS MonthlySales
FROM dbo.Bicycle
GROUP BY YEAR(OrderDate) * 100 + MONTH(OrderDate)) AS t
ON dbo.DPI.YearMonth = t.SaleYearMonth
```

The inner SELECT statement handles the computations to get monthly sales and this output is assigned to an internal temporary table named t. The outer select joins those rows to the DPI table based on the year/month columns and displays the results. Examine a couple rows of the named query to see that it works correctly.

After the data is defined, building the mining model is straightforward. Create a new mining model using Microsoft Time Series, based on the new named query. Work through the wizard, selecting Monthly Sales as the predictable column, [Sale Year Month] as the time key, and add DPI and PCI as Input columns. After the wizard finishes, remember to return to the Mining Models tab and specify the Periodicity\_Hint as {1, 12} to indicate monthly data. Select the Microsoft\_Time\_Series entry, and click the ellipses button in the properties for Algorithm Parameters. Process the model and Browse the results. The ARIMA model results should be the same as before—they are based solely on the predictable series. Although, the dataset might be slightly different because real-world data might not exist yet for the 2013 and 2014 years.

Figure 8.38 shows the results for the node that handles all months after 2007. Check the Mining Legend for details about the node. The fact that the change point is 2008 is important alone—that was a key year in the housing market crash. The coefficients indicate the declining trend with the negative coefficient on sales for the prior month. That shift altered the effect of both income (DPI) and inflation (PCI). In essence, the values measure a structural shift in the underlying economic impacts.

At this point, an analyst for the company would look at the values for all of the other nodes. Note there is another sale month change after 2008, which is further split based on values of the DPI. Fortunately, that node (not shown here) reveals that the coefficient on DPI turns positive after 2008 and once DPI exceeds a value of 11028. The point is that these nodes providing the starting point for deeper



**Figure 8.38**

Results of the ARTxp model with economic data. Results for the node: Sale Year Month  $\geq 200713$ . First note that 2008 caused a shift point in sales. Second, the coefficients indicate the downward trend from the prior month; and the role of DPI and PCI have changed.

understanding of the data. Analysts who lived and worked through 2008 might remember the impact of the U.S. housing market crash. The key is that it does show up in the model results.

### *ARMAX Addition of Series to ARIMA*

Many tools that support traditional ARIMA also include the option to analyze the impacts of other data on the main time series. Gretl has a simple option to specify other series as inputs, and then estimates an ARMAX model based on the approach defined by Box-Jenkins. The new columns are evaluated as single points. The model does not automatically test these columns at different lag points.

The data from the U.S. BEA Web site needs to be integrated into the existing sales data. Because the sales data is stored in a simple CSV file, the easiest approach is to edit that file in Excel, download the two new columns and paste them into the main CSV file. Estimating the model is almost identical to the traditional ARIMA model. The trend needs to be differenced out and the lag structure needs to be identified from the ACF and PACF charts. These lags should be similar to the values found in the initial model with the single series, but they can be tweaked in the new model if necessary. Set up the model as before, using AR terms of 1 and 12, a single difference ( $D=1$ ), and MA terms of 1, 2, 3, and 10. Then in the gretl model definition window, move the DPI and price index variables to the Input box. Run the model.

	coefficient	std. error	t-ratio	p-value	
const	-8710.8	8019.9	-1.086	0.2774	
phi_1	0.0286	0.0812	0.3529	0.7242	
phi_12	0.5701	0.0591	9.6520	0.0000	***
theta_1	-0.2700	0.0950	-2.8420	0.0045	***
theta_2	-0.4881	0.0802	-6.0870	0.0000	***
theta_3	-0.0854	0.0892	-0.9572	0.3885	
theta_10	-0.1565	0.0552	-2.8350	0.0046	***
DPI	220.61	187.1	1.1790	0.2385	
PCI	37251.3	42794	0.8705	0.3840	

**Figure 8.39**

ARMAX (gretl) results with the DPI and price index series. The ARIMA structure is similar to the base model. The DPI coefficient is negative but not significant, and the price index (PCI) is positive but not significant.

Figure 8.39 shows the coefficient estimates for the new model. The lag structure is similar to the base model. The new series (DPI and PCI) do not appear to have significant impacts on the monthly sales data. Both are positive but not significant. The inflation measure is also not significantly different from zero. If the PCI coefficient really were positive, it would likely indicate that the value of sales increases during times of inflation. Because the sales value is measured in nominal prices, such a result would simply indicate that managers raise prices during inflationary times, so total sales increase. For this reason, any model that examines price changes should probably look at quantity of sales instead of just value.

### *Linear Regression*

Linear regression has been used for many years and several variations have been developed to handle time series and dynamic data. The theoretical details are beyond the scope of this book, but issues involving lags, auto-regression, and other complications can be found in almost any econometrics book. The techniques used in this section are relatively common methods to handle basic estimation of trends, seasonal effects, and evaluate the impact of other data on the main series. In some ways, bringing in multiple series can cause the most problems. Consider the basic problem of trying to estimate a demand curve by evaluating data collected over time that compares quantity sold to sale price. Economic theory reveals that this relationship should almost always be negative. Yet, in many cases, if this data is estimated with a simple regression, the coefficient will be positive. The problem is that too many other variables will affect the quantity sold, and at least some of these need to be added to the estimation. In econometric terms, it is a problem with simultaneous equations. The observed data consists of changes in the equilibrium point or intersection of the demand and supply curves over time. The model needs enough data to measure changes in both the demand curve as well as the supply curve. Then special estimation techniques are needed to separate out the impacts on both curves. Again, these topics are beyond the scope of this book. Just remember that when you try to measure effects from multiple time series, you should seek the assistance of an expert.

With time series data, it is highly likely that values in one period affect values in the next. This auto-regressive model is common in economics and several methods exist to estimate the effect. One of the easiest approaches is to define lagged variables of the dependent variable. That is, create new X-variables that are lags of the Sales variable and estimate the coefficients directly:

$$\text{Sales} = b_0 + b_1\text{Sales}(-1) + b_2\text{Sales}(-2) + b_3\text{Sales}(-3) + b_4\text{Sales}(-12)$$

The analyst can choose the lag variables, similar to the way AR terms are selected in the ARIMA model. The regression results will return a test of the significance of each coefficient.

Trend patterns are even easier to estimate. Simply include a measure of the time variable—often the year is used to measure annual changes. These values are easily generated in SQL with the Year function when the sales totals are created. Using a generic (-i) to represent all lag values, the model becomes:

$$\text{Sales} = b_0 + b_1\text{Sales}(-i) + b_y \text{Year}$$

Seasonal data is also relatively easy. It is tempting to use a Month column similar to the concept of the Year column. This approach will work, but the interpretation is different from a true seasonal measure. It would indicate whether any month affected the results, with no way to identify which month or season made a difference. To determine which specific months made a difference, it is necessary to create dummy variables—essentially one for each month. A **dummy variable** has a value of zero or one. The value is one if the desired attribute (month) is true, otherwise it is zero. So, define 12 new variables. In gretl, an example is: `M01=(Month==1)`. This statement defines a new variable (M01) and sets each observation to 0 except when the Month value is 1 (January). However, when using dummy variables in a linear regression, a specific problem arises. If all 12 dummy variables are included in the regression, it is not possible to include the constant term. Because only 12 months exist, all of the data is covered and defined by those 12 dummy variables. The constant term is then a linear combination of the 12 variables and linear regression does not work if some variables are perfectly correlated. The solution is to drop either the constant term or to drop one of the 12 dummy variables. In most cases, it makes sense to keep the constant term and drop the first dummy variable (M01 for January). Nothing is lost. The impact results for January can be obtained as a linear combination of the other months and the constant term. The regression model now looks something like:

$$\text{Sales} = b_0 + b_1\text{Sales}(-i) + b_y \text{Year} + b_{m2}M02 + \dots b_{m12}M12$$

Finally, it is possible to add terms for the other series. These variables will hold the same interpretation as they would in a traditional linear regression model. The model can now be written as:

$$\text{Sales} = b_0 + b_1\text{Sales}(-i) + b_y \text{Year} + b_{m2}M02 + \dots b_{m12}M12 + b_d\text{DPI} + b_p\text{PCI}$$

Of course, it is also possible that the exogenous variables (DPI and PCI) do not have an immediate effect on sales. Particularly for income, it is possible that the impact is delayed. People might choose to make a purchase a few months in advance (particularly for the year-end holidays). Even if income drops in the month of the sale, consumers might complete the purchase because the decision has already been made. So, perhaps the real effect of income occurs a month or so before the sale. This dynamic situation can be evaluated by including lagged values of the DPI and PCI data. It should be clear that the regression approach is flexible enough to accommodate relatively complex models.



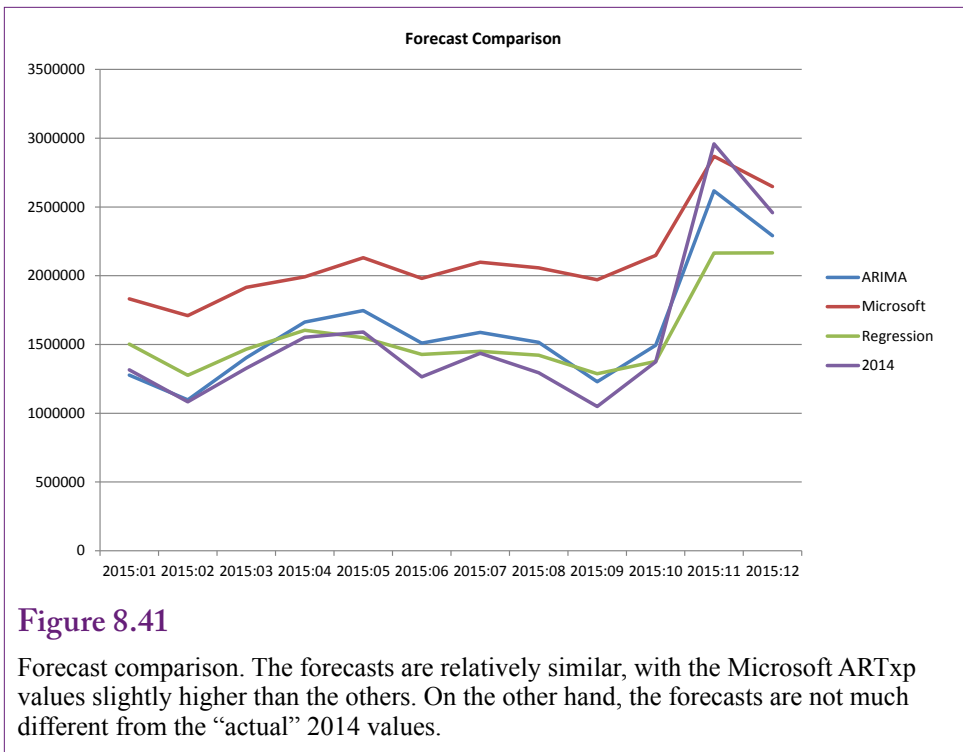
	coefficient	std. error	t-ratio	p-value	
const	-173894000	64787700	-2.684	0.0079	***
DPI_1	-105.5	181.089	-0.582	0.5610	
DPI_3	-137.9	186.6	-0.739	0.4608	
PCI_1	7500.1	45595.8	0.1645	0.8695	
PCI_3	7349.5	44302.2	0.1659	0.8684	
M02	51363.2	78077.0	0.6579	0.5114	
M03	105546	81738.7	1.291	0.1982	
M04	136553	78115.6	1.748	0.0821	*
M05	78413.0	73504.1	1.067	0.2874	
M06	98425.6	73524.4	1.339	0.1823	
M07	123598	76786.4	1.610	0.1092	
M08	98079.3	78434.4	1.250	0.2127	
M09	68710.6	77535.8	0.8862	0.3767	
M10	146903	78651.6	1.868	0.0634	*
M11	438455	82203.4	5.334	0.0000	***
M12	130437	83798.5	1.557	0.1213	
Year	87203.4	32756.3	2.662	0.0084	***
Sales_1	0.6043	0.0709	8.527	0.0000	***
Sales_2	-0.3251	0.0829	-3.925	0.0001	***
Sales_3	0.1356	0.0826	1.642	0.1023	
Sales_4	-0.02148	0.0734	-0.3051	0.7606	
Sales_12	0.2702	0.0615	4.395	0.0000	***

**Figure 8.40**

Regression results. Notice the significant seasonal (month) terms. Notice the positive trend (Year). The AR lags on the dependent variable loosely match the results from the other tools. The external series (DPI and PCI) do not have a significant impact on sales.  $R^2 = 0.7799$ .

The process of obtaining all of the lag and dummy variables depends on the specific tool used. Most, such as gretl, SAS, and SPSS have functions to define lags and new variables with one command. The interactive version of gretl even has a form that makes it easy to specify the number of periods to use for each lag.

Figure 8.40 shows the results of the regression. The  $R^2$  value indicates the regression results are reasonable—however, time series data with auto-regressive terms often have high  $R^2$  values, and 78 percent is not exceptional. Check the significance on the seasonal (monthly) terms. The results loosely match those from the ARIMA model. April, and November are important months. However, July and October show strong coefficients here, but not in the ARIMA model. The catch with the dummy variable approach is that the January results are incorporated into the values. July and October usually have lower sales than other months, but they are higher on average than January. The coefficient on the Year variable is significant and positive, so there is an upward trend to the data in the



**Figure 8.41**

Forecast comparison. The forecasts are relatively similar, with the Microsoft ARTxp values slightly higher than the others. On the other hand, the forecasts are not much different from the “actual” 2014 values.

amount of about \$87,000 a year. The AR lags on Sales indicate that prior sales are important—particularly for the past two months. One month back has a strong positive influence on sales for the current month, but this boost is mitigated slightly by the negative effect from two-months prior. The significant coefficient on the 12-month lag also indicates the presence of a seasonal effect.

Finally, notice that the DPI (income) and PCI (inflation) terms do not significantly affect the sales results. Even the lagged values are not significantly different from zero. A couple of the lagged coefficients appear strong, so perhaps if more data is acquired, more variables are added, or variability declines, these variables might influence total sales. The key point is that it is relatively easy to incorporate external data and still measure time series effects with linear regression.

## Comparison

This chapter covers several tools that try to accomplish the same task: Identify patterns in time series and use them to forecast future values. Many data mining tools for time series rely on ARIMA as the foundation to estimate trend and seasonal effects. Still, variations in the methods lead to differences in the coefficients and in the forecasts. Figure 8.41 shows the forecasts for the first four months of 2015 using the three basic tools covered in the chapter. The values are all relatively close, except the Microsoft ARTxp forecast numbers appear higher than the others. The ARIMA forecast has a bit more variation than the Microsoft forecast. Note the slight increase over the “actual” 2014 values for the regression and ARIMA forecasts. Overall, they are showing minimal trend effects. And although the ARTxp forecast seems too high for most of the year, it is the only one that seems to pick up the full forecast for the months of October and November. Regression

Lag	ARIMA/MA	Microsoft/MA	Regression/AR
1	-0.3340	-0.1403	0.6043
2	-0.2690		-0.3251
3	-0.2348		0.1356
4	-0.0917		-0.0215
10	-0.1499		
12	0.0124	-0.0672	0.2702

**Figure 8.42**  
Seasonality comparison. Because of the dissimilarities the results are difficult to summarize. However, the primary 12-month lag values are except for Microsoft's ARTxp.

and ARIMA are both predicting that those months for 2015 would be lower than the observed values in 2014. The bottom line is that no model is “perfect.”

Figure 8.42 is more confusing. It compares the seasonality results from the three models. The values are not completely comparable because the techniques are somewhat different. Specifically, the linear regression approach estimates AR terms instead of MA values. Still, it would be nice if the seasonal effects were similar across the models. The big difference is that the 12-month lag coefficient is positive except in Microsoft's ARTxp model. Even including the SAR(12) and SAR(24) terms does not resolve this difference. Although not shown in the table, the ARTxp AR(1) term of 0.233 is larger than the AR(1) term of 0.026 from the ARIMA model. So the ARTxp model is responding more strongly to recent events. The negative coefficients for the 2-period lag are interesting—they are also somewhat incorporated into the Microsoft 1-period lag. They imply a consistent damping effect against the other attributes. In essence, when RT has a good sales month, the next month is almost always going to be lower. But that effect represents the seasonality—the data indicates that consumers tend to have a couple of targeted purchasing months.

## Summary

Time series data is common in business. Managers commonly make forecasts based on patterns over time. Basic questions include the total level of sales, peaks and troughs in weekly and seasonal patterns so that adequate staffing can be provided and forecasting consumer needs for the next holiday period. Time series data often exhibit patterns—including long-term trends or growth over time, seasonal patterns such as holiday sales or weekly peaks, and cyclical changes that follow the general economy. The essence of time series forecasting is to evaluate a series and identify its patterns. These patterns can then be used to forecast the series into the future. Auto-regressive moving average tools are a common method for evaluating patterns—particularly seasonal patterns. These AR and MA terms are defined as coefficients that weight the effect of lags in the original data and the variations. Traditional Box-Jenkins/ARIMA tools provide auto-correlation and partial auto-correlation functions to help determine the lag structure of time series data. Trends are removed from ARIMA models by differencing the data—usually once for linear growth, twice for non-linear trends.

The Microsoft Time Series tool mixes ARIMA forecasts with a proprietary ARTxp tool that uses a decision-tree approach to find significant split points in the data. ARTxp focuses on auto-regressive lags to determine a piecewise linear model. Microsoft claims ARTxp is best at forecasting out a few time periods. For further forecasts, the tool mixes and eventually switches to a prediction based on an ARIMA model. The Microsoft forecasting tool has a useful trick to apply patterns in a larger series to a new series that is assumed will follow the same pattern—such as when introducing a new model or product.

Incorporating effects from business cycles (national income) or from other time series events requires tools that can handle dynamic models. Complex tools have been developed to analyze specialized data—particularly in finance—but this chapter covers only basic tools. Some ARIMA tools, such as gretl, support the addition of other series and can estimate coefficients to measure their correlation with the original series. Microsoft ARTxp evaluates the effects of other series when building a decision tree, but does not change the ARIMA model. Linear regression can be used to estimate trend, seasonal, and cross-correlation effects. Trend is estimated by including a time variable in the model. Seasonal effects are identified by adding a binary dummy variable for each season, less one if the constant term is included. Time series models generally include lagged dependent variables as well to capture dynamic effects and autocorrelation. Cross correlation or cyclical effects are evaluated by including other variables, where the resulting coefficients provide an estimate of the correlation.

When time series data is highly variable, the results from the different tools can vary. When multiple models are created that have different results it can be difficult to choose among the models to determine the most likely forecast. Various diagnostic measures can be used to compare models, but ultimately the final prediction will simply have a high variation. The bottom line is that forecasting the future is often difficult.

## Key Words

---

Akaike information criterion (AIC)  
 auto regression  
 auto regressive integrated moving average (ARIMA)  
 autocorrelation function (ACF)  
 auto-regressive tree with cross prediction (ARTxp)  
 chaotic  
 cross correlation  
 cyclical  
 dummy variable  
 moving average  
 partial autocorrelation function (PACF)

PredictTimeSeries Microsoft function  
 Schwarz criterion or Bayesian information criterion (BIC)  
 seasonal ARIMA (SARIMA)  
 seasonal auto-regressive (SAR)  
 seasonal effect  
 seasonal moving average (SMA)  
 seasonality  
 seasonally adjusted  
 time series  
 trend

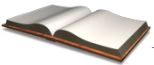
## Review Questions

---

1. What are the main components of a time series?
2. What is the difference between auto-regressive and moving average terms in a time series?
3. How does the ARIMA model handle long-term trend in a time series?
4. How are the ACF and PACF used to identify the preliminary lag structure of a times series?
5. What measures are often used to compare the performance of different time series models?
6. How is missing data handled in time series analysis?
7. How are forecasts made with the traditional ARIMA model?
8. How is seasonality identified with the ARIMA model?
9. How is Microsoft ARIMA different from the traditional ARIMA (ignoring ARTxp for now)?
10. What is the most important parameter to specify in a Microsoft Time Series model?
11. What does Microsoft's ARTxp provide that is not available in traditional ARIMA?
12. What is the purpose of cross correlation models and what problems might they create?

## Exercises

---



### Book

1. Set up and run the traditional ARIMA example from the chapter. Summarize the results and interpretation.
2. Set up and run the Microsoft Time Series analysis example from the chapter. Summarize the results and interpretation.
3. Set up and run the cross-correlation example from the chapter using Microsoft Time Series analysis. Summarize the results and interpretation.
4. Compute the average percent sales by month for Rolling Thunder Bicycles. As outlined in the chapter, compute the percentage of sales by month within each year and then average those values by month. Compare the results to the simpler method used in the chapter.



## Rolling Thunder Database

5. Rerun the time series analysis using quantity of bicycles sold instead of sale price as the primary series. Compare the results to those using prices.
6. Rerun the time series analysis using sale prices, but use quarterly instead of monthly subtotals. Compare the results to those using monthly totals.
7. Examine the time series of purchases of component parts from vendors. Is there a seasonal pattern?
8. Assume the company introduces a new model type (downhill or free ride) in 2014. Given the sales values listed in the following table, and assuming that the sales will follow the pattern of all mountain bikes, use Microsoft Time Series tools to estimate the sales pattern for 2015.

July	August	September	October	November	December
55,612	38,291	36,289	25,423	67,500	71,300



## Diner

9. Examine the total sales by week and identify any seasonal patterns that exist.
10. Examine the total sales by day of week and identify any patterns that exist.



## Corner Med

11. Examine total visits by month and identify any seasonal patterns and trends that exist.
12. Examine total visits by day of week and identify any patterns that exist.
13. Examine total revenue (charges) from visits by month and identify any seasonal patterns and trends that exist.



## Basketball

14. Choose one team and one season. Examine average points per game scored by that team by week and identify any time series patterns.
15. Choose one team and season and examine total fouls by that team by week and identify any time series patterns.
16. Choose one team and season and compute the average free throw percentage by week (total free throws made / total free throws attempted per week). Identify any patterns found.
17. Choose one player. Examine total points scored by that player per game by week in one season and identify any time series patterns.



## Bakery

18. Examine total sales revenue by month and identify any trends and patterns.
19. Examine total sales revenue by day and identify any trends and patterns.
20. Examine total sales revenue by hour and identify any daily patterns.



## Cars

21. Use the monthly car sales data to create a new database table. Identify any trends and seasonal patterns in the total sales. Source: Bureau of Economic Analysis: [http://www.bea.gov/national/nipaweb/nipa\\_underlying/TableView.asp?SelectedTable=55&FirstYear=2009&LastYear=2009&Freq=Month&ViewSeries=N](http://www.bea.gov/national/nipaweb/nipa_underlying/TableView.asp?SelectedTable=55&FirstYear=2009&LastYear=2009&Freq=Month&ViewSeries=N)
22. Use the data from the previous question and identify patterns and trends for the domestic and import manufacturers and comment on the two results.





## Teamwork

23. Assign one model type of bicycle to each team member. Use time series analysis to examine monthly sales of bicycles by model type. Are there differences in seasonal models (lags) for the different model types? Note, skip track and hybrid models because they have limited sales.
24. Assign one basketball team to each person in the group. Compute the average weekly field goal percentage for the team (total field goals made/total field goals attempted that week). Do a time series analysis by team and compare results obtained by all members in the group.
25. Using the Corner Med database create subgroups for each of the physicians. Examine total number of weekly visits handled by each physician separately and identify any patterns or trends. Compare the results for each physician and against the total.
26. Using the Bakery database, assign one product category to each person. Analyze total daily sales for the chosen category and identify any patterns and trends. Combine the results from each category and comment on the results.

## Additional Reading

---

Bowerman, Bruce L. and Richard T. O'Connell, 1979, *Time Series and Forecasting*, Duxbury: North Scituate Massachusetts. [A good introduction to the statistics of Time Series, particularly ARIMA. Undergraduate level.]

Box, G.E.P., and G.M. Jenkins, 1976, *Time Series Analysis: Forecasting and Control*, Holden-Day, San Francisco. [Classic description of ARIMA. Mathematical.]

Judge, George G., W.E. Griffiths, R. Carter Hill, Helmut Lütkepohl, and Tsoung-Chao Lee, 1985, *The Theory and Practice of Econometrics, second edition*, Wiley: New York. [A classic complete work on econometric theory for those who want to know how to handle problems that arise with regressions and time series introduction. Graduate level.]

# Specialized Tools

## How are complex or non-traditional analyses performed?

Several specialized tools can be used to analyze data. In some cases, the tools have to be purchased and installed separately, which makes it necessary to connect the tools to the data warehouse.

Problems involving geographic location are common in business, such as locating customers, stores, suppliers, and employees. Geographic information systems are useful tools to display and help visualize relationships based on location. Expensive tools exist, but relatively inexpensive and accessible tools are available from Microsoft (MapPoint) and some Web sites (Google Maps). GIS consists of more than just maps—it is the ability to display different levels of data to find patterns based on location.

Sequence analysis is similar to association, but refers to sequences of events, such as identifying which Web pages are most commonly viewed by surfers on a Web site, or what steps are most commonly followed by people assembling a product. The high-end versions of SQL Server (Enterprise and Developer) contain a tool for sequence analysis.

Multidimensional expressions (MDX) are used throughout SQL Server Analysis Services to define functions. They also underlie all of the analyses. Building a model is actually a process of defining the MDX required to run a specific analysis. Consequently, it is possible to customize the way a tool runs by creating or editing MDX expressions. MDX functions are also supported by other data mining tools.

---

**Chapter 9**      Sequence Analysis and GIS

**Chapter 10**     Multidimensional Expressions (MDX)

---

# Sequence Analysis and GIS

## Chapter Outline

Introduction, 435	Geographic Analysis, 459
Sequence Analysis , 435	Business Situation, 461
Business Situation, 437	Data, 461
Model, 438	Model, 462
<i>Classification, 439</i>	Microsoft MapPoint, 464
<i>Clustering: Business examples, 440</i>	Other Tools, 467
Data, 442	<i>Esri, 468</i>
<i>Attributes and Observations, 442</i>	<i>Google, 471</i>
<i>Continuous and Discrete Data, 444</i>	<i>Bing, 472</i>
<i>Missing Data, 444</i>	<i>Federal Government, 473</i>
<i>Web Log Data Files, 445</i>	Geographic Summary, 474
SQL Server Sequence Clustering, 447	Key Words, 474
<i>Goals, 448</i>	Review Questions, 475
<i>Data, 448</i>	Exercises, 476
<i>Tools, 449</i>	Additional Reading, 479
<i>Results, 452</i>	
<i>Prediction, 457</i>	
Other Tools, 458	
Sequence Summary, 458	

## What You Will Learn in This Chapter

- How are patterns over time or sequences of data or actions analyzed?
- What types of business problems are suited for sequence analysis?
- What are clusters of sequences?
- How is data organized for sequence analysis?
- How is sequence clustering used to analyze Web site traffic?
- What features are provided by other sequence mining tools?

### Houston Pawn Shops

A geographic information system (GIS) examines data related to location. Spatial Insights is a GIS consulting company and it used GIS tools to evaluate the market for check cashing and pawn shops in Houston, TX. The analysis begins with data on the locations of existing pawn shops and check cashing companies. It then uses Census demographic data to identify the areas with the highest potential demand for the services. Factors include population, median household income, percentage of renter occupied dwelling units, average household size, and population density. To combine the various factors, each was divided into quintiles (20 percent ranges), and each range was given a value from 1 through 5. The individual ratings were summed to provide a single score for each Census block. This index value can then be plotted on a color-coded trend map to show the areas with the highest potential demand. The supply side was mapped using the locations of existing stores by color coding each block based on the total square footage of existing retail stores. The supply and demand side values can be combined by reverse coding the supply and adding the two numbers. A color-coded heat map of the result highlights the areas that have the highest potential demand with the lowest number of existing stores. [Spatial Insights 2008]

Many business decisions and data are related to location. Government demographic data is useful in evaluating many of these problems, and several tools exist to display data geographically.

Spatial Insights, Inc., *Market Potential GIS Case Study*, 2008. [The PDF file contains the charts.] <http://www.spatialinsights.com/company/papers/downloads/MarketPotentialStudies.pdf>

## Introduction

---

**What tools exist for special purpose analyses for sequences of data and geographic problems?** Several specialized problems exist in business and other disciplines and interesting tools have been developed to provide useful information in these situations. The problems and tools are a little different from other forms of analysis but they can be useful in specific situations.

This chapter examines two specific types of problems: (1) Sequence data, and (2) Geographic or location-based data. The two topics are not related to each other, but both topics are relatively straightforward and can be examined in just part of a chapter. Sequence data has an interesting history, with most of the tools derived from genetic research into DNA and protein sequences. However, some interesting business problems are based on sequences—notably tracking user interactions on Web sites.

Many business problems are related to location—almost anything related to customers, suppliers, and competitors. Geographic analysis goes beyond simple mapping—it examines how data is correlated through location. For example, how are customer income and sales related through location—how much difference does a wealthier neighborhood make to sales? Many of the geographical tools are visual—emphasizing the ability to display the data on maps to make it easier to see the correlations.

The sequence data and tools are considered first in this chapter because the analysis is relatively complex. Microsoft SQL Server Analysis supports some types of sequence analysis so the examples focus on its capabilities. The geographical analysis follows in separate sections. The challenge to the geographical analysis is to get access to a tool. Microsoft MapPoint is used to demonstrate the basic mapping capabilities, but other tools exist including Google for Web-based charts and ESRI ArcInfo on the high-end.

## Sequence Analysis

---

**How are patterns over time or sequences of data or actions analyzed?** Several prominent examples of sequences raised important questions and led to the development of tools for analyzing the specific types of data involved. Two of the classic examples are: (1) DNA (and protein) sequences, and (2) Evaluating Web site logs to identify usage patterns. In the context of these tools, a **sequence** is a set of data or actions in a specific order. More importantly, the tools are designed to find patterns or groups of similar sequences. For instance, are there groups of customers who follow a common path when interacting with a Web site? Or, in the biology world, are some patterns of DNA (genes) correlated with specific outcomes or diseases?

The DNA example is interesting and many powerful tools have been designed specifically to attack the many biology questions involved. The results and background are not particularly useful in a general business context, but some business problems echo the same characteristics, so a couple of basic concepts are useful. DNA, the biological building block of life, consists of sequences of four molecules. The four nucleobases are cytosine, guanine, adenine, and thymine; commonly abbreviated as CGAT. Consequently, portions of DNA sequences are written using the four letters, such as: CCGATCGGTA, but the sequences contain millions or even billions of these characters. Researchers often need to compare multiple DNA sequences; for example, comparing sequences from two different

people. Or, if a group of people have a specific disease (such as a rare cancer), finding similarities among their DNA sequences that might have caused a susceptibility to the disease. A key element of analyzing DNA sequences is the need to begin at a specific point within the overall sequence, such as when comparing an individual gene. Many of the DNA tools are highly optimized for these specific tasks and are not directly applicable to business problems. However, some of the base concepts have been generalized into tools that can be used for other problems. Further, the DNA notation is convenient to illustrate concepts because each item consists of a single letter, so sequences are easy to write down.

The most common business example of sequences is analyzing patterns in Web site usage. A Web site consists of a set of pages and user browsing consists of moving through the pages in some relatively random order. Marketers and Web site developers are interested in learning if certain types of people follow similar paths through the Web site. For example, do people who end up making a purchase follow some path that leads them to make a purchase; or do those who do not make a purchase somehow miss a critical page? Stop for a minute and see how this problem is similar to the DNA situation. Each person has (or follows) a sequence of items and researchers want to find similarities in those sequences for groups of people.

Sequence mining tools follow two general topics: (1) **classification** and (2) **clustering**. Technically, a third version of tools combines both topics into one method. Classification is used to help identify a sequence. It can be based on the entire sequence but more likely uses a subset. For example, a researcher might want to know if a specific subset exists within the sequence; such as searching for a specific gene pattern in DNA or identifying whether a Web site visitor worked through a set of checkout pages. Detailed classification can involve identifying or specifying the starting location of a subset. Beyond simple pattern matching, sequences raise difficult questions about how close a pattern must be to constitute a match. In particular, gaps are an important topic. For example, does the sequence CAGTC contain the subset CC? The answer depends on the length of the **gap**, or spacing between items, that the researcher is willing to consider. With a zero-length gap (or no gap), the answer is “no” the sequence does not contain the CC subset. But, if a gap of at least 4 units is allowed, then the sequence does match. Longer patterns also create issues with partial matches. If a pattern contains 50 items and a match is found for all but one entry, should it count as a success, or does the researcher need an exact match?

Most of the business sequence mining tools use a version of clustering which is conceptually similar to the basic cluster tools. Simple clustering tools compare items or people based on attribute measures. They use a distance measure to find items that are close to each other and distant from items that belong in a different cluster. In terms of sequences, the tools search for patterns that are similar across a group of observations. With most business mining tools it is also possible to include static dimensions if they are available—such as customer income or location.

Clustering requires a distance measure function and much of the work in developing tools relies on developing better distance measures. Distance is useful as a measure of closeness—instead of relying on exact matches. For example, the **Levenshtein distance** or **edit distance** is an interesting concept for many sequences. Given two sequences,  $S_1$  and  $S_2$ , the edit distance is the minimum number of edit operations needed to convert  $S_1$  into  $S_2$ . It is easiest to see in terms of

simple strings or words. For example, converting the sequence of letters “dollar” to “holler” requires two steps: change “d” to “h” and “a” to “e” so the edit distance is 2. But converting “dollar” to “pound” requires at least 5 steps. The edit distance is not always the best measure of similarity. More sophisticated measures assign different weights to change, delete, and insert moves; and others are better at comparing from different positions or alignment. Common algorithms include BLAST, BLOSUM, PAM, FASTA, and Smith-Waterman. With most data mining systems, the choice of the distance function is made by the developers and you will rarely be able to change it. However, it can be one of the main differences between systems, so different tools can give different results. If you have specific data, you might need to test various tools to find one that uses a distance function best suited to the data.

## Business Situation

---

**What types of business problems are suited for sequence analysis?** Perhaps the most important question in this chapter is to know when sequence analysis can be used. Many concepts in business might appear to be sequence related, but they do not work well for common sequence analysis tools. In particular, much business data consists of time series, such as sales for each month. But sequence analysis on this data will not be very useful. (Instead, time series analysis is more useful.)

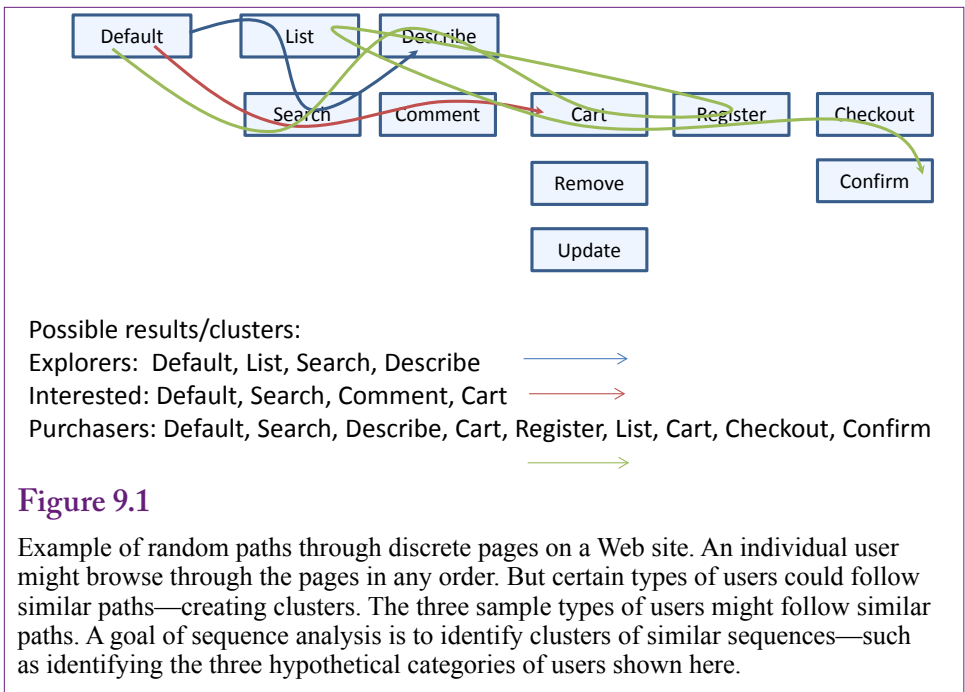
At a minimum, most sequence analysis tools require two things: (1) discrete items or events in order, and (2) individual tracks or random paths over that data. The second condition is used for clustering—finding groups that follow similar paths or sequences. Most business-based tools rely on clustering analysis. The alternative is classification analysis which requires defined patterns and outcomes; but these tools are more commonly found in biology applications.

The use of discrete data is relatively important. The data can be numeric or text, but continuous data rarely works. Even discrete items with too many options can cause problems. If tens of thousands of items can appear in a sequence, then it is unlikely that any two sequences will match. In some cases, you can **discretize** the data to create categories or **bins** that reduce the overall number of items. For example, a problem might create sequences by measuring the time between various events. Technically, time is a continuous variable, so it should be converted by defining categories based on intervals (short, medium, long, and so on).

The second condition for clustering is more restrictive—multiple random paths through the data. If the paths are not random, they are not very interesting. For example, a manufacturing company likely has a sequence for producing items, but the base sequence is usually fixed not random—all products go through the same steps in the same order. With a large, complex manufacturing system randomness might be introduced through other variables. For example, multiple machines or workers might exist at each step, so the sequence becomes a question of tracking which machine or person did the work, not the actual processing step. But even then, regression analysis might be more useful than sequence analysis.

Because of the importance of randomness in the sequence, most business use of sequence analysis is based on customer actions. Customers tend to generate random actions; but obtaining the data requires some method of tracking all customer actions. Web sites are one of the few places for accurately tracking customer actions. Two common practices for sequence modeling are (1) analyzing Web page sequences in browsing and purchasing, and (2) evaluating the sequence in which





**Figure 9.1**

Example of random paths through discrete pages on a Web site. An individual user might browse through the pages in any order. But certain types of users could follow similar paths—creating clusters. The three sample types of users might follow similar paths. A goal of sequence analysis is to identify clusters of similar sequences—such as identifying the three hypothetical categories of users shown here.

products are added to a shopping cart. It might be possible to identify similar actions in bricks-and-mortar stores, but the data collection is difficult and intrusive. (For example, a few grocery stores have experimented with hand-held scanners that shoppers carry through the store to record their own purchases as they load their carts.)

Sequence analysis could be applied to other business problems, just remember to focus on the actions and the desire to compare multiple random sequences. Focus on the potential results: sequence analysis identifies clusters of similar sequences. The clusters can include the sequence data and they can often include other, fixed data. For example, customer income, location, age, or other dimensions can be included in the search for clusters.

## Model

**What are clusters of sequences?** Identifying similar sets of users or customers is a common sequence mining problem in business. Figure 9.1 shows the basic concepts for a Web site project. With a Web site, a sequence consists of a set of pages that are browsed by an individual user—including the order in which they are viewed. As will be shown in the examples, Web servers automatically track this data in a log file—at least identified by user IP address. At a minimum, the data consists of files that show the user IP address, a time stamp, and the name of the page that was delivered to the user. The objective is to identify groups of users or clusters that follow similar paths. From a business perspective, these clusters should indicate different types of customers. To identify clusters, the system searches for multiple people who follow similar sequences. In the example, some customers simply browse the site for a couple of products, a second cluster searches for products and reads reviews, while a third cluster finds a product and makes a purchase.

$$P(S) = P(S_m | S_{m-1}) P(S_{m-1} | S_{m-2}) \dots P(S_2 | S_1) P(S_1)$$

**Figure 9.2**

Markov chain probabilities. From probability theory, the probability of any state can be computed as the chained probability of each previous item in the sequence. Starting from the right side is the probability of the starting point; which is multiplied by the probability of the second item given (|) the first item. Most Markov chains are restricted to looking at a specified number of entries, called the order (k).

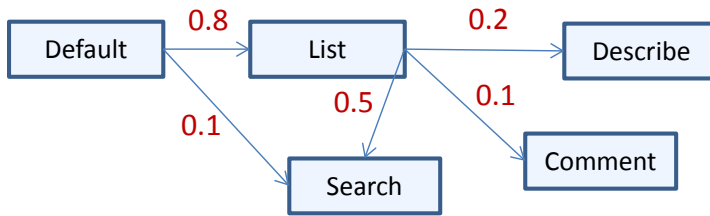
By identifying the various clusters, it might be possible to understand the customers and perhaps find ways to convert browsers into purchasers. Obviously, any group making a purchase will generate a sequence of pages using the shopping cart that will not exist with non-purchasers. If the only difference between these groups is the shopping cart pages, then the information available will be minimal. From a marketing perspective, the resulting clusters will be more useful if they reveal a difference in paths for the clusters before the purchase steps. Then it might be possible to compare the initial sequences of purchasers and non-purchasers to see which specific page or pages make a difference. For example, perhaps people who make a purchase often read reviews by other customers, while those who do not make a purchase skip (or never find) the reviews. In that case, it would be useful to make the review comments more prominent—such as including them directly on the product page without requiring a second step.

A relatively common method for analyzing sequence data is to estimate the values for a Markov chain. A **Markov chain** is a concept from probability and statistics—where any next state (or sequence item in this case) depends on the current state. The probability of moving from one state to another one is given by the **transition probability**. In the context of the Web site example, a person begins at the default page and works through several pages to get to a product description page. At any point, including this page, there is a probability that the person will move to the shopping cart (purchase the product), along with probabilities of moving to almost any other page. Figure 9.2 shows the basic mathematical concept of a Markov chain using conditional probabilities, which demonstrates how a sequence depends on all of the prior states or events in the sequence.

The transition probabilities estimated in a Markov chain are more useful from a business perspective. As shown in Figure 9.3, picture a user at a specific page in the Web site. The transition probabilities indicate the chance of the user moving to any of the next available pages. A key step in sequence analysis is to estimate these transition probabilities. These values are going to be different within each sequence cluster. People with similar paths will have similar (average) transition probabilities for moving to other items. Comparing these probabilities provides information on the differences across clusters which can help explain how various groups use the Web site differently.

## Classification

Classification of sequences or outcomes is a more complex task. Essentially, it requires matching sequences or patterns. As noted in the Model section, matching requires considering gaps and alignment issues. The technical steps are not important in business, but the researcher needs to determine whether gaps and alignment are critical issues for the specific problem.



**Figure 9.3**

Transition probabilities. Think about a user viewing a specific page such as the Product List page. There is some probability of the user moving to any other available page, such as 0.20 to get to a product Description, 0.50 to do a search, and 0.10 to read the product comments. A key step in sequence clustering is to estimate these transition probabilities.

Sequence classification problems in business are similar to other prediction tools. In most situations, the business has a set of outcomes and wants to determine which sequences are more likely to lead to a specified outcome. For example, a bank has sequences of loan payments for many borrowers. These values could be recoded as full payment, partial payment, no payment, or late payment; or perhaps in number of days late. The outcome of interest would be whether the borrower defaults on the loan. The sequence data mining tool would then look for patterns that are likely to lead to borrower defaults. If valid results are produced, the sequence patterns identified could be used to forecast problems with future borrowers.

One complication with sequence classification is that many of the techniques and tools were designed for DNA analysis and might need to be reconfigured to work with business data. For example, sequence classification is not supported in SQL Server Analysis Services. But, over time, more tools are likely to be made available.

### Clustering: Business examples

By far, the most common business example for sequence analysis is evaluating Web site logs. The two main tasks are (1) finding clusters of browsing sequences for different classes of users, and (2) evaluating the sequence in which customers add items to a shopping cart. The challenge with business examples is finding the appropriate data sets. The clustering tools need a set of discrete actions or events and a set of random tracks through those events. The requirement for multiple, random events rules out many potential business examples. For instance, most production steps are fixed and not random.

In many organizations, customer behavior is a key random element, so most business applications involve the analysis of customer actions. But collecting detailed data on customers can be a challenge—particularly in terms of maintaining privacy or at least anonymity. Web sites are one of the easiest ways to track detailed customer interactions. With SQL Server Analysis Services, it is also possible to include other dimensions on customers that are not part of the sequence data. For instance, firms might have income, age, gender, or customer category data. These elements can be used as part of the clustering computations to help differentiate sequences. For instance, browsing might be different for males versus females.

Sample DNA Data for individuals:

1. ACAGGTACTGGA...
2. CCAGGAATACG...
3. ACATTACTGAAG...

1. M, 32, ...
2. F, 45, ...
3. M, 62, ...

Each string represents a sequence for a single individual.  
The order is contained within the data string (left-to-right).  
Additional data might exist for each individual (gender, age, ...)

### Figure 9.4

Organizing data—DNA example. The DNA example is easy to visualize, partly because only four values exist for each item. The data consists of two components: (1) The sequence of items (shown as a string), and (2) Data about the individuals.

To add more customer dimensions, additional tracking data will be needed. A Web server does not exactly identify a user (or customer). Instead, it tracks usage by IP address. In most cases, an IP address is unique to a person for at least a short period of time. Over time, IP addresses are often recycled and used by other people; they cannot be used to identify individual customers. To track individuals, it is necessary to create accounts and have customers log in, then the associated ID value will have to be recorded along with the various events. From that point, it becomes possible to attach the other dimensions from the database.

It is possible to create other business applications for sequence data mining. However, the challenge is to find random data for the paths. It is also important to use discrete events or observations. For example, a manufacturing firm might be interested in tracking production sequences to see if some of them lead to higher error rates. In general, most production steps are fixed, but if the production events include different interactions by humans, then randomness will be added. However, in the end, the analysis will be evaluating the human (or possibly machine) randomness. The key is to understand exactly what is being measured and evaluated in the clusters.

Some trickier business examples can use timing as the random element. For example, perhaps customers go through only four or five steps; but those steps might be stretched out over time. That is, the events are fixed, but the time between them is random. For instance, an engineering firm might go through the same basic steps with each client, such as: define project, detail design, start construction, make changes, complete project. Because the steps are similar for each project, they do not provide a useful sequence for analysis. However, the time between the steps is more likely to be random so that data would provide more interesting results for sequence analysis. The time would have to be discretized to convert to a finite and smaller number of possible sequences.

## Data

**How is data organized for sequence analysis?** The main issues with the type of data have already been described: Discrete events or items where the order is important. Analysis also requires a collection of observations of these sequences that have a random element—essentially random tracks through the events or items. If the tracks are fixed instead of random there is nothing to gain from analyzing them because the goal of the analysis is to find patterns or clusters of similar tracks.

Collecting and organizing the data for sequential analysis is important because the tools are picky about how the data must be structured. Also, understanding the data requirements makes it easier to think about business applications that can benefit from sequence analysis. In some ways it is easier to think about the data by starting with the DNA tasks—partly because it is easier to write down and visualize the data. Figure 9.4 provides a simple example. A sequence for one individual is written as a string of letters where the order is defined by the position within the string from left-to-right. Additional data might exist for each individual and could be stored in a second table, and include things such as gender and age.

### Attributes and Observations

Business data is likely to be more complex than the DNA examples—each item usually consists of considerably more than four possible values, although the sequences will rarely be as long as a DNA chain. Business sequence data is also often stored in different formats—such as different rows in a table. For instance, in the common problem of analyzing a Web log, each page browse is stored as a single row in a table. Figure 9.5 shows some sample Web log data as it would be stored in a database table. Note that the Web logs are initially sorted by date and time. But, most importantly, each event is stored as a separate row within the table; and the rows from multiple users are intermixed. Without a customer login step, the only way to identify a sequence is to rely on the UserIP address. Within a period of time, a given user generally maintains a single IP address. The sample data in the table comes from a real Web log and shows three different IP addresses, so it contains the start of three sequences.

**Figure 9.5**

Sample Web log data. Some of the available data for each Web log event. Note that each event is stored as a separate row. Without a customer login step, the only way to create a sequence is the UserIP address. But the sequences are intermixed.

ID	Date	Status	Time	UserIP	Method	Bytes	URIStem	Bytes	Referer
1	01:28.0	200	250	207.46.199.39	GET	29247	/DBMS/...	344	NULL
2	03:11.0	200	46	173.192.34.91	GET	0	/	162	NULL
3	08:11.0	200	46	173.192.34.91	GET	0	/	162	NULL
4	11:16.0	200	140	62.212.73.211	GET	318	/robots.txt	327	NULL
5	11:17.0	200	140	62.212.73.211	GET	715	/petstore/...	364	NULL

PersonID	Seq	Item
1	1	A
1	2	C
2	1	C
1	3	A
2	2	C
3	1	A

1. ACAGGTACTGGA...
2. CCAGGAATACGG...
3. ACATTACTGAAG...

**Figure 9.6**

Organizing data—DNA example as a table. The table requires two keys: Person and the sequence order. Note that the data might be intermixed. Some tools will require the data to be sorted correctly before beginning the analysis.

For comparison, how would the DNA data be stored in a similar table? Figure 9.6 provides a partial answer, but the actual table would be considerably larger. Note that the table requires two key columns: (1) The identifier for the individual, and (2) a sequence order. In database terms, the data can be intermixed within the table. Most tools will require the data to be sorted correctly (PersonID, Seq) before beginning the analysis. The point of the example is that business data is typically stored in the data table format so business tools are designed to handle data in that format. However, it is more difficult to visualize the sequences in a table.

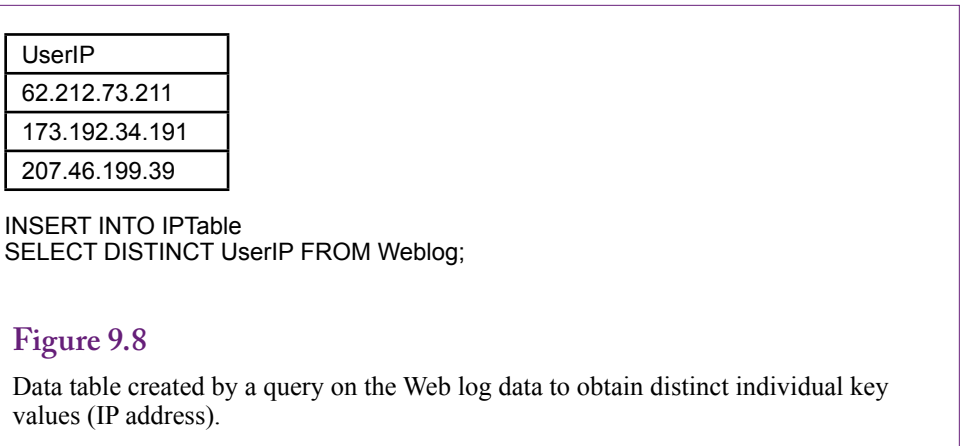
Additional dimensions on individuals would be stored in a separate table. It would have a single key column (PersonID) and columns for the dimensional data (age, gender, and so on). Figure 9.7 shows a sample table for the DNA data example. This table needs only one column in the key (PersonID) which is why it needs to be a separate table from the sequence data that used two columns (PersonID, and SequenceOrder). The two tables are linked by the PersonID column.

In the case of pure Web log data, the additional data regarding the individuals does not exist. However, many sequence analysis tools still require a base table

**Figure 9.7**

Data table for individuals. Only one column is needed for this key (PersonID) so it needs to be a separate table from the sequence data.

PersonID	Gender	Age	...
1	M	32	
2	F	45	
3	M	62	



for individuals. Essentially, this table needs to list each person one time. In this case, it is necessary to create the table by using a query on the Web log table:

```
SELECT DISTINCT UserIP FROM Weblog;
```

Note the use of the “DISTINCT” keyword to eliminate duplicate entries. This query essentially creates a table with a single column (UserIP) that contains each IP address a single time. Figure 9.8 shows the sample data. Ideally, the analysis tool will allow the use of the simple SQL query. However, if it requires a separate table, Figure 9.8 shows the addition of the INSERT INTO phrase that takes the selected data rows and inserts them into a table that was created with a single column.

### Continuous and Discrete Data

Sequence mining requires discrete data. In business terms, the sequences are usually events, such as browsing a Web page or selecting a product. The data is already discrete, and the researcher simply has to code the data so that the items will be recognizable in the evaluation of the results.

It is possible for business problems to use continuous data—such as using time intervals as a sequence. In these situations, the data will need to be converted to discrete bins; such as short, medium, and long time frames. A SQL Server CASE function can be used to define the categories. In more complex cases, a new table can be created with the category definitions and inequality-joins can be used to assign the category values to the data.

### Missing Data

Obviously, key columns cannot contain missing data. For the most part, sequences do not contain missing data—particularly in business exercises. For example, a clickstream generated from browsing a Web site simply moves from one page to the next. There is no requirement on the number of pages to be visited; so sequences can be varying lengths. For other problems that require a defined number of sequence items or that rely on position, missing data is handled by defining it as a new state. In the DNA example items would be allowed to take on five values: C, G, A, T, and missing; instead of just the four base values. However, no real rules exist for handling the missing entries. Instead, it is up to the specific tool,



and in some cases control over parameters to determine how to handle the missing values. For instance, they might be treated as gaps—that match any value. Or, they might be treated as a separate item. The difference matters most when using classification tools instead of clustering tools. For example, does the partial DNA string: CCA\_T match the string: CCAGT or CCATT or neither one? The answer has to depend on the specific problem and the researcher, so it should be controllable through parameters in the tools.

## Web Log Data Files

*Note: This section is only necessary if you want to read your own Web log files. It is not needed to load the sample databases for the book.*

Web logs are a common business problem analyzed with sequence mining tools. However, they have some special properties that need to be considered before they can be analyzed. The most important issue is that some standards exist for storing Web log files. In particular, the Web log files are almost always stored as plain text files. Each row represents one event, such as a page retrieval, and the detailed data are stored in a specific order on the row. The data is generally spread across multiple text files—where one file commonly holds data for a single day. The data mining tools, particularly SQL Server, require extracting the data from the text files and storing it in tables in a relational database. Fortunately, Web logs are used for several purposes so some tools exist to help with the conversion. Different tools exist for different servers, so additional research might be necessary to find a tool for a specific server.

Microsoft supports a parser tool that reads, extracts, and aggregates data from common Microsoft log files. The current version of the tool is “Log Parser 2.2” which has been around since 2005. It is available as a free download from Microsoft—just search for it by name. The tool is relatively powerful and can read common Microsoft files including Web logs, event logs, registry entries, and XML and CSV files. However, it runs as a command line tool without a graphical-user interface. This feature actually makes it more useful because it is easier to automate, but it can make it harder to learn the various options. More recently, Microsoft has added the “Log Parser Studio” which runs as a graphical interface on top of Log Parser. Only the Log Parser 2.2 will be used for the examples in this chapter.

The data files provided for this book have already been converted into database format so it is not necessary to deal with the original log files to follow the examples in the book. However, to perform a similar task on other log files, it will be necessary to convert them, so it is worth understanding the basic steps. Figure 9.9 shows the Log Parser command used to create a CSV file from text files generated by Microsoft IIS. The command needs to be entered on a single row but is split up to make it easier to read. The basic format is similar to a simple SQL SELECT command. The goal is to select the desired columns, but the data can be pulled from multiple log files and combined into a single output file. It is possible to add simple WHERE conditions, but they are not needed in this example. The example contains several special functions to clean up the data. First, the `OUT_ROW_NUMBER()` function is used to generate a unique ID value for each row. Second, the log files treat date and time as separate columns, but they will work better in the database as a single column. So the `TO_STRING` and `ADD` functions are used to combine the two columns into one. Finally, three columns can contain long character strings: User-Agent (browser), cs-uri-stem (page request), and cs(Referer). These values need to be limited to a set number of characters (512

```

"c:\Program Files (x86)\Log Parser 2.2\LogParser"
"SELECT OUT_ROW_NUMBER() AS ID,
ADD(TO_STRING(date,'MM/dd/yyyy'), TO_STRING(time,' hh:mm:ss')) AS dt,
sc-status,
time-taken,
REPLACE_CHAR(SUBSTR(cs(User-Agent),0,512), ',',';') AS User-Agent,
c-ip, cs-method, sc-bytes, cs-version,
REPLACE_CHAR(SUBSTR(cs-uri-stem,0,512), ',',';') AS CS_URI_Stem,
cs-bytes,
REPLACE_CHAR(SUBSTR(cs(Referer),0,512), ',',';') AS CS_Referred
INTO temp.csv
FROM *.log
ORDER BY dt"
-i:IISW3C -o:CSV

```

*Generated ID #*

*Combine date and time into one.*

*Limit length and change commas to semicolons*

*Specify Web log format as input and CSV as output*

**Figure 9.9**

Log Parser command to read IIS Web logs and convert to a CSV file for loading into a database table. Only common dimensions are selected. In the log, date and time are separate columns that need to be combined into a single column for the database table. Also, three of the strings are limited in size (512) and examined to convert commas into semicolons to prevent loading problems. The command must be entered as a single-row.

in this case) because the corresponding table in the database will be given a finite amount of storage. Additionally, the data will be loaded from an intermediate **comma-separated-values (CSV)** file and the data columns cannot contain extra commas. So the REPLACE function is used to convert commas to semicolons. In some cases, it might be more useful to simply truncate the data at the point a comma is encountered, which relies on the use of the INDEX\_OF function.

As a side question: How do you find the names of the columns (such as cs-uri-stem)? Usually, these names are header lines in the log files. However, it is often easier to issue a test statement of the form: SELECT \* FROM \*.log and then break the command after a couple of rows have been displayed; which will display the column names as a heading.

The remaining elements of the Log Parser command specify the input files (\*.log), the output file (INTO temp.csv), and the sort order (which is optional). The -i and -o options specify the format of the input and output files. The choices can be found by typing the Log Parser command without any parameters. Using the command-line version of Log Parser, you open a new command window (Start: cmd), navigate to the folder holding the copies of the log file (cd ...), and type the full command. With many large log files, it might take several minutes or more for the command to run and generate the CSV file.

Once the CSV file has been created, it can be examined inside Excel—although large files will be truncated. More importantly, it can be imported into SQL Server using the Bulk Load command. First create a table with the matching columns. Then issue the EXECUTE (N'Bulk INSERT ...) command. Figure 9.10 shows an example of the CREATE TABLE and EXECUTE commands that match the data from the Log Parser command. The columns in the table need to exactly match

```

DECLARE @data_path nvarchar(256);
SELECT @data_path = '<PATH>';
--
CREATE TABLE WebLogs
(
  ID int identity(1,1) NOT NULL,
  WebDateTime datetime NULL,
  SCStatus int NULL,
  TimeTaken int NULL,
  UserAgent nvarchar(512) NULL,
  UserIP nvarchar(250) NULL,
  CSMethod nvarchar(250) NULL,
  Bytes bigint NULL,
  CSVersion nvarchar(250) NULL,
  URISem nvarchar(512) NULL,
  CSBytes bigint NULL,
  Referer nvarchar(512) NULL,
  CONSTRAINT pk_WebLogs PRIMARY KEY (ID)
);
EXECUTE (N'BULK INSERT Weblogs FROM ''' + @data_path + N'temp.csv'
WITH (
  CHECK_CONSTRAINTS,
  CODEPAGE="ACP",
  DATAFILETYPE = "char",
  FIELDTERMINATOR= ",",
  ROWTERMINATOR = "\n",
  KEEPIDENTITY,
  TABLOCK
);');

```

**Figure 9.10**

SQL Server commands to create a table to hold the log data and the BULK INSERT command to load the table from the CSV file created from the text log files. Change the <PATH> entry to match the folder holding the CSV file.

the columns in the CSV file. Change the <PATH> command to be the name of the folder holding the CSV file, such as C:\Temp\ or wherever the file is stored. The BULK INSERT command then loads all of the CSV data into the database table. A similar process is used to transfer data files for this textbook because it is relatively fast and works with all recent versions of SQL Server.

## SQL Server Sequence Clustering

**How is sequence clustering used to analyze Web site traffic?** Microsoft SQL Server Analysis includes a sequence clustering tool that is useful for business analyses (as opposed to DNA). It can be used to analyze Web server logs to identify common patterns in usage. Remember that clustering works by finding groups of sequences that are similar to each other. SQL Server also supports the use of non-sequence dimensions in the clusters—behaving similar to traditional clustering. However, this data is usually only available if the Web site requires people to log in and then collects personal data on the users. The example used in this section does not include any data that can be traced to individuals.

To illustrate the high variability in real data, this section uses data collected from a live Web server. The Web server is used by the author to handle educational books and class content. The IP data was randomized so that it is not possible to track addresses back to individual users (even if it were possible to obtain network-address translation logs). Despite the relatively small number of total users (19,600 unique IP addresses), the log files generate a large amount of data—several hundred megabytes for five months of usage. This section shows the basic process of configuring the sequence analysis and briefly explains how to evaluate the various result screens.

## Goals

The primary goal of using sequence clustering on Web logs is to identify the main groups of users. Most organizations and Web sites develop traffic from various groups and it helps to understand the differences and similarities in how these groups use the Web site. For example, in a business site, some users will simply be browsers—perhaps searching for products or comparing prices. Hopefully, one group will be purchasers—people who buy products online. Possibly one group will be investors—looking for company background information. It is not necessary to know about these groups ahead of time—the clustering tool will automatically identify different groups and show how they behave differently in working through Web pages.

It will help to understand the results if the researcher knows the Web site—and the various pages. It will also be useful to have some initial concepts of the people who use the Web site. The clustering tool does not need this information, but the results can be difficult to follow if the page names are meaningless and the researcher has no background in the industry.

## Data

As explained in the prior section, the data is pulled from Web server logs. Web servers automatically record pages, date/time, and user IP address along with other tidbits such as the size of the pages (in bytes). Converting these text log files into a database table can be a problem—but the Log Parser simplifies and automates the task. Be sure to add a key column (ID) to the data and ensure that it is sorted by date, time, and IP address. Date/time by itself is not going to work as a key because Web servers are multitasking and can deliver multiple pages at essentially the same time. Also, the IP address cannot be used as a key in this sequence table.

Web analysis today has an additional catch. Large organizations often use relatively automated systems to manage their Web servers. These content-management systems consist of shell pages that retrieve data from a database to be displayed on a single page. For instance, a user might search and request data on a specific product. This data will always be displayed on a page that might be called “products.” The server log only records the “product” page—it does not record the data displayed on that page. Some systems deliver most content through these systems—including images and text. Likewise, as companies add more programming code to Web pages, it becomes more difficult for simple text log files to track the total user interaction. The log files will provide a broad overview of user actions, but perhaps not the detail wanted by everyone. Detailed Web site tracking requires the installation of tracking code directly on the pages. For example, Google Analytics provides several tools to track and display detailed usage on Web pages using this approach. However, these tools do not usually perform sequence clustering so both methods are often useful.

## Tools

Begin by loading the sample data into a SQL Server database (WebLogs). Because the data does not contain other information about customers, only one table is created. Then create a new project in Visual Studio using the Analysis projects. Add a connection to the WebLogs database and create a Data Source View that uses the WebLogs table.

SQL Server sequence clustering always requires two tables: (1) a Case table that contains a key value for each sequence—such as a CustomerID; and (2) the actual sequence item data in a “Nested” table. But, the sample data loads only the single (nested) table where each row contains a sequence item. The solution is to create a “Named Query” in the data source view that holds a list of unique UserIP addresses. The IP address will be used as an identifier for the each user. It is not perfect—IP addresses get reassigned over time, but it is the only identifier available.

In the data source view, create a named query and call it UserIPView. Add the WebLogs table and select the UserIP column. Modify the SQL so that it includes the DISTINCT keyword:

```
SELECT DISTINCT UserIP FROM WebLogs;
```

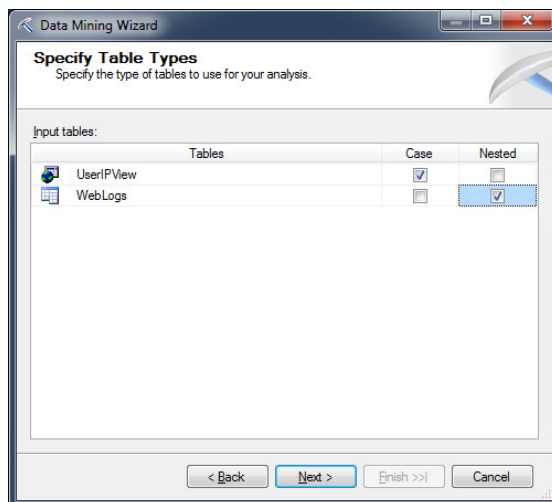
Because the WebLogs table contains the actual sequence data, the UserIP repeats. This new view extracts just the unique values of the IP address and treats the data as a separate table. After the view is created, you will have to right-click the new UserIP column and assign it as the logical primary key. Finally, drag the UserIP column from the WebLogs table and drop it on the UserIP column in the new UserIPView query to establish a relationship.

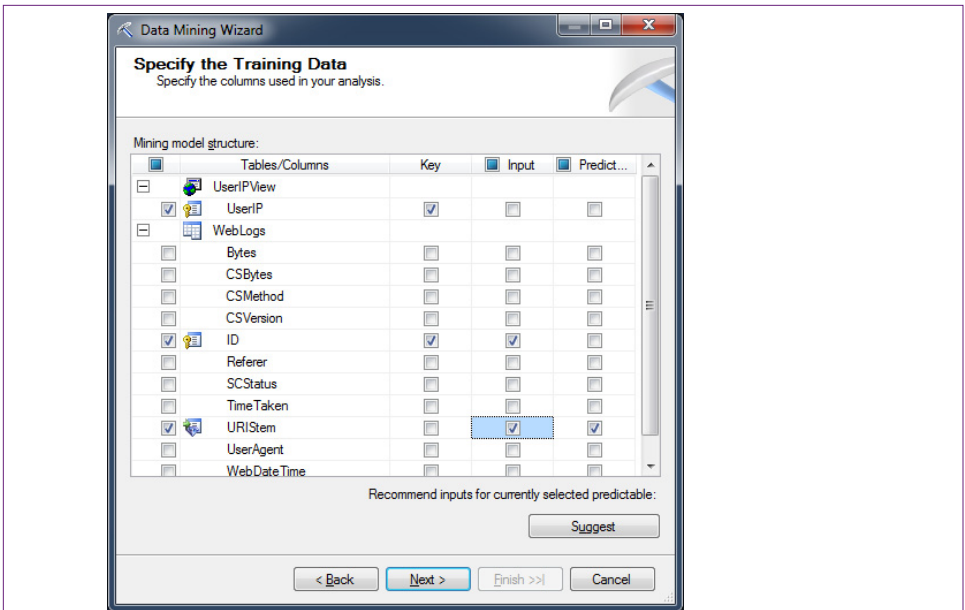
### *Configuring Microsoft Sequence Clustering*

With the data defined in both a Case and Nested table, the sequence cluster can be created as a new Mining Structure. Figure 9.11 shows one of the first steps in

**Figure 9.11**

Case and nested table selection in SQL Server sequence clustering.





**Figure 9.12**

Selecting key columns correctly. The UserIP is the only possible key for the UserIPView Case table. The generated ID column is the key (and input) for the Nested WebLogs table. The selection of the sequence items is critical. In this case it is the URISem column which contains the name of each page that was browsed. The URISem column is selected as Input and Predicted—never as key.

the process—setting the Case and Nested tables. Remember that the Nested table contains each sequence item on a new row. The Case table holds unique values for the user that generated the sequence. The Case table usually contains individual or customer data. This table or view can hold other non-sequence data such as age and gender if it is available.

One of the most important and trickiest steps is to correctly select the columns for the analysis. Figure 9.12 shows the choice of the columns for analyzing the Web logs. The UserIP column in the UserIPView table is the only possible key. In other situations, a CustomerID might be the key column. The key for the WebLogs data is the ID column which contains generated values. This column must be carefully created—it specifies the sort order for the sequence. Recall that it was created based on date and time. It would be tempting to use the WebDateTime column instead but it will not work because it includes duplicate data—the Web server delivered some pages at the exact same date and time because of multitasking. Also, note that the UserIP in the WebLogs table is not available because it was already selected in the Case table; none of the other columns track the sequence. The next step is to select the URISem column as the sequence item data. This column holds the name of the page that was requested so it represents the selected value—much like the letters CGAT in the DNA problem. This column is specified as the input and predicted column—but can never be chosen as the key. If additional data on customers is available in the Case table, these columns can also be selected as input and predicted columns in this step.

Parameter	Default	Range
CLUSTER_COUNT	10	[0,...)
MAXIMUM_SEQUENCE_STATES	64	0, [2,65535]
MAXIMUM_STATES	10	0, [2,65535]
MINIMUM_SUPPORT	10	[0,...)

**Figure 9.13**

Control parameters. These parameters control the processing for the sequence clustering tool. CLUSTER\_COUNT is treated as a hint not a fixed value. A value of 0 requests use of a heuristic to choose the number of clusters. The MAXIMUM\_SEQUENCE\_STATES parameter restricts the number of item values observed to those that are most common.

Those are the main steps for configuring the sequence clustering: (1) Choose the Case and Nested tables, (2) Specify the key columns in those two tables, and (3) Select the column in the Nested table that indicates the item value for the sequence. Processing the model is handled the same way as with other tools. Right-click the new data mining model and choose the option to process it. The processing could take a while to run—the sample data includes several million rows.

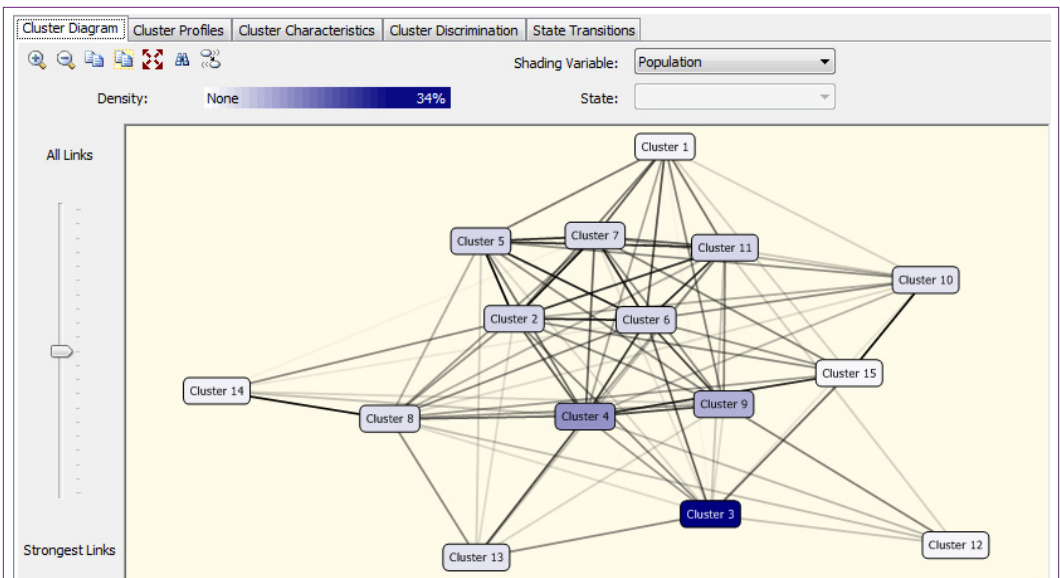
#### *Processing the Sequence Cluster Model*

In Microsoft Visual Studio, processing is initiated the same as with other tools: Right-click the model name in the explorer window and choose the option to Process the model. However, sequence clustering requires a hefty amount of processing power and time. The sample data on a fast machine with a high-speed drive is reasonably fast and takes only a couple of minutes at most. However, remember that the data is pulled from a relatively small server and consists of only five months of Web logs.

Despite the relatively small size of the server pages, when processed, the sequence cluster tool complains about the number of items (pages) and it generates a warning message that: *Cardinality reduction has been applied*. Look at the parameters for the sequence cluster model (Mining Models tab, right-click the Microsoft\_Sequence\_Cluster entry). Figure 9.13 shows the four parameters available to control the processing. The CLUSTER\_COUNT variable defaults to 10 but this number is treated as a hint not an absolute constraint. Setting the value to zero requests the use of a heuristic to determine the best number of clusters to use. The parameter MAXIMUM\_SEQUENCE\_STATES is more important in many situations. It limits the number of values to use for each item state. In the Web browsing example, the default value of 64 limits the number of pages to examine to 64. The system tries to choose the most active pages, so it discards pages that have few hits. This value can be increased to bring in more pages. However, Microsoft notes that values larger than 100 “can result in a meaningless model.” It is this parameter value that is triggering the warning message about cardinality.

For a first pass model, letting the system choose the most active pages probably makes sense. Researchers first want to examine the most commonly-used pages. However, it is possible that the lesser-used pages are important. For instance, per-





**Figure 9.14**

Cluster diagram. This diagram highlights the number of observations in each cluster—darker tints represent more observations. The link lines indicate how closely the clusters are related—heavier lines indicate closer relationships.

haps higher-end customers use specific short-cut pages. The solution to examining these other pages is to create a query that selects just the pages you want to investigate. For complex Web sites it can be helpful to build models that cover separate sections of the site.

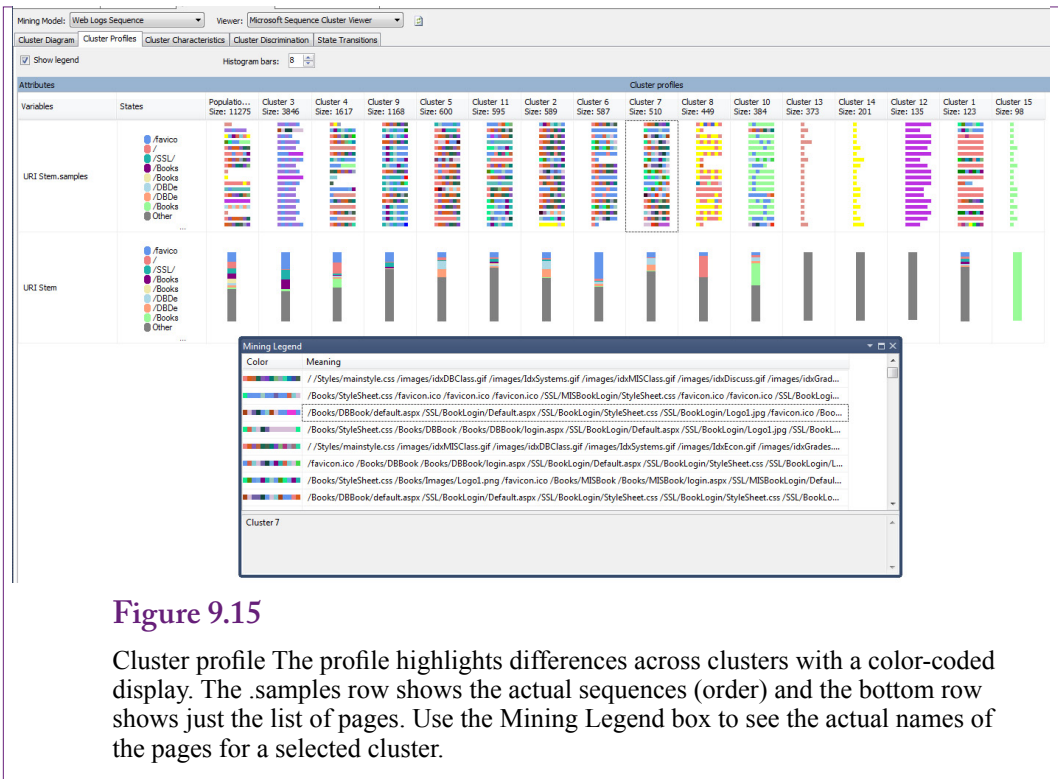
## Results

The results viewer in Microsoft Analysis Services is similar to that for standard clustering. Several graphical views are available along with a few tabular presentations. It is also possible to write queries to extract data directly from the Analysis database. However, the results for sequence clusters are a little more challenging to read than traditional clusters because the sequences can be relatively long. In one sense, sequence clustering needs to display data similar to traditional clustering—such as the choice of the Web pages displayed. But the results also need a way to show the order in which those pages were visited.

### *Clusters*

Figure 9.14 shows an example of the cluster diagram which provides an overview of the results. The main purpose is to show the number of clusters where the number of observations within each cluster are highlighted by the darkness of the tint. In the example, Cluster 3 has the most number of similar responses. The thickness of the link lines also shows how closely the clusters are related. Heavier lines indicate more overlap between clusters. SQL Server uses the expectation maximization algorithm which computes a probability that each item (sequence) belongs to a cluster so an identified sequence might be associated with multiple clusters.

The cluster diagram has an additional feature that is useful for understanding clusters in terms of Web pages. By default, the shading is set for the population



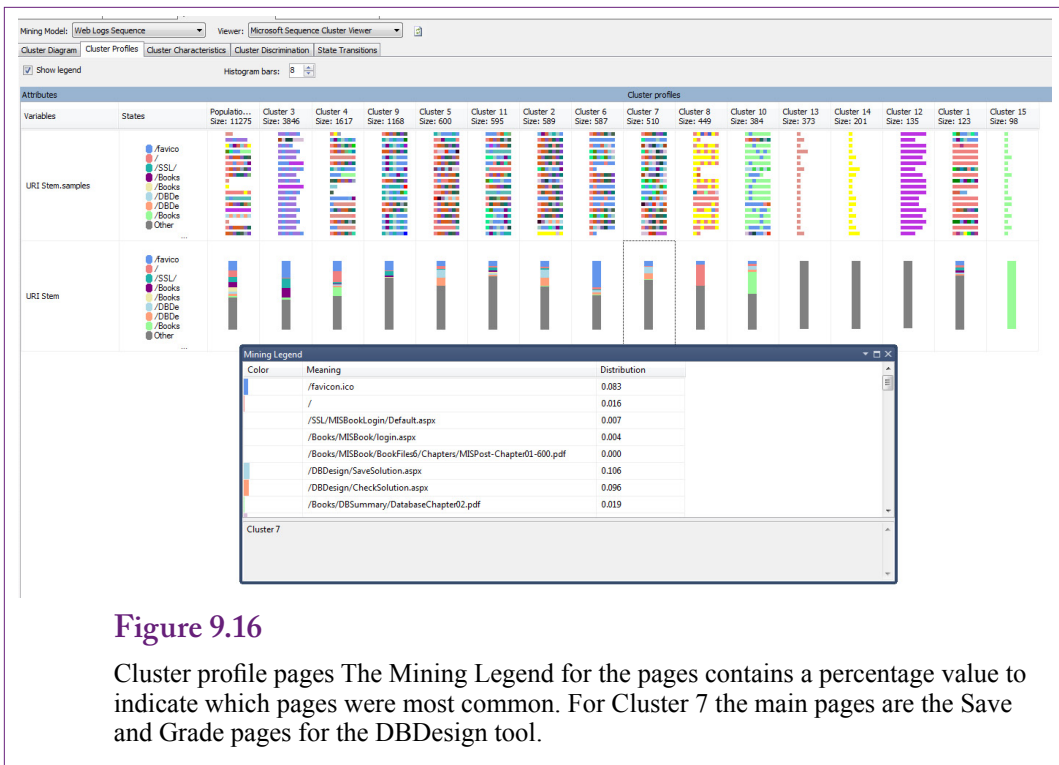
**Figure 9.15**

**Cluster profile** The profile highlights differences across clusters with a color-coded display. The `.samples` row shows the actual sequences (order) and the bottom row shows just the list of pages. Use the Mining Legend box to see the actual names of the pages for a selected cluster.

or number of sequences within the cluster, making it easier to spot the clusters with the most traffic. However, a drop-down box can be used to change the Shading Variable to the item values (URI Stem in this example). In this mode, a second drop-down box enables selection of individual pages (up to the limit of the 64 pages allowed). This trick then highlights the clusters that most heavily use a specific page. For example, choose the `/robots.txt` page and see which sequences most heavily use that page. The `robots.txt` file is a specific file aimed at search engines and it contains rules about what the robots should focus on and which pages they should ignore (voluntarily). Any sequences that heavily use that file are visits from search engine robots.

The three main result views focus on the individual clusters: Cluster Profiles, Cluster Characteristics, and Cluster Discrimination. The Profiles tab displays color-coded indicators for the sequences (URI Stem.samples) and the pages (URI Stem). The display shows multiple clusters on the screen to emphasize the differences across the clusters. On the other hand, the Characteristics tab focuses on a single cluster at a time—by showing the detailed list of items (pages).

Figure 9.15 shows the cluster profile for the sample data with the results highlighted for Cluster 7. Notice that the display contains two primary rows: (1) Top row labeled with `.sample` shows the sequences or page ordering, and (2) Bottom row shows just the list of pages. If additional customer dimensions were available and used they would be included in the bottom row analysis. Selecting a cluster in either the top or bottom row brings up the details in the Mining Legend box, which makes it easier to see the actual page names. Sequences are still difficult to see because they are usually too long to fit into the display box. It might be easier to read the results if the original data was recoded with shorter names.



As shown in Figure 9.16, the profile display for the pages contains some additional information—the percentage of hits for each page. For Cluster 7, the main pages are the Save and Grade pages from the DBDesign tool on the Web site. So this particular sequence represents usage of that tool. As these results are discovered, the clusters can be renamed to indicate the primary sequence usage.

Figure 9.17 shows more detail for the pages within a cluster. Notice the emphasis on the database book login, database design, and chapters 1 – 3 of the database textbook. These pages again indicate that Cluster 7 represents a group of students studying database design. Identifying the users of the clusters is a useful first step in understanding the results. However, it requires knowledge of the Web site and the various types of users.

Sometimes it is difficult to understand a cluster—particularly when clusters include seemingly unrelated pages. In these situations, it is possible that the model contains too many clusters; or that user sequences simply do not group together. The Cluster Discrimination view helps understand a cluster by displaying pages that are important to the cluster versus pages that are important to other clusters. As shown in Figure 9.18, the researcher can choose to compare two clusters side-by-side. However, the default is to compare the selected cluster (7 in this case) to its complement—things that are not in the chosen cluster. The drop-down box makes it easy to select a second cluster for situations where two clusters seem to contain similar data sequences.

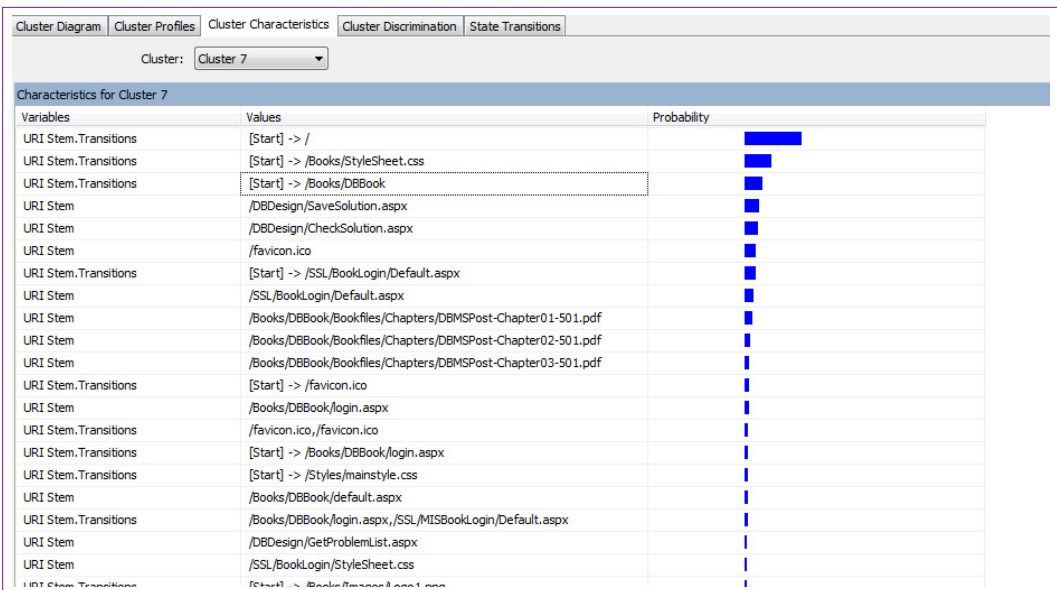
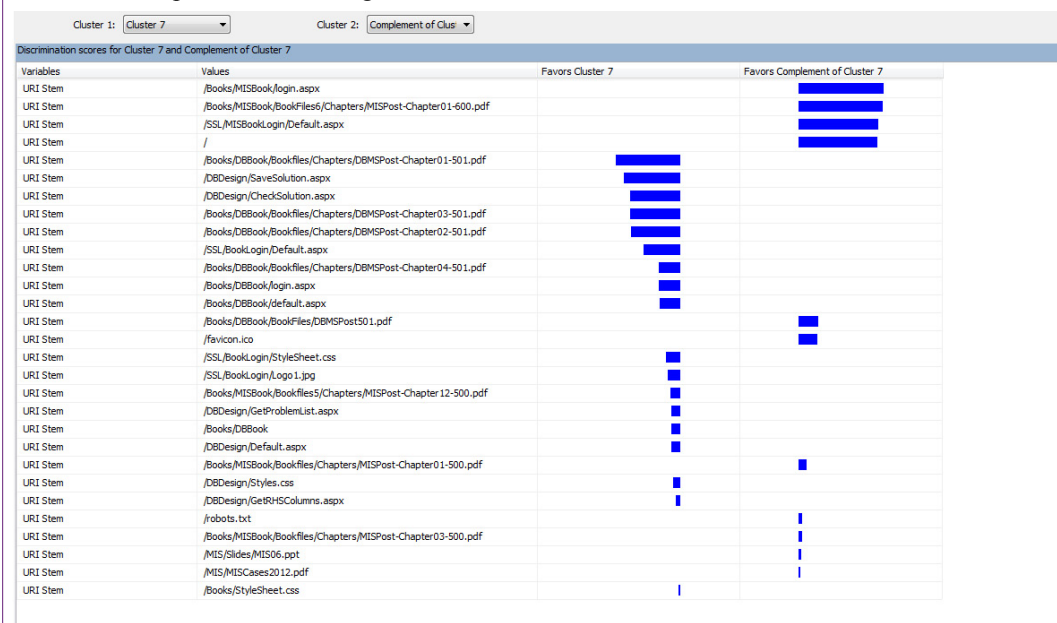


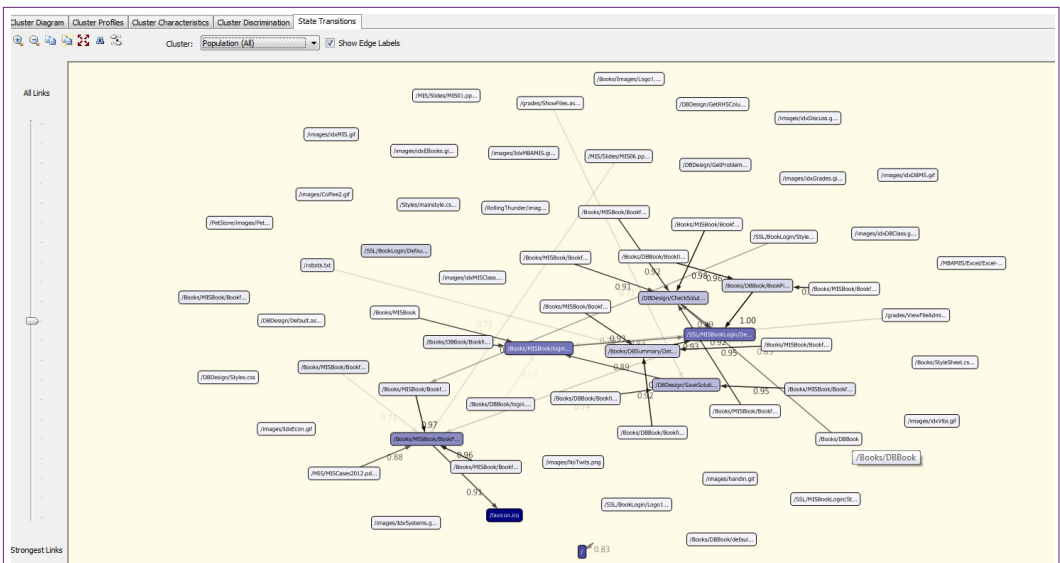
Figure 9.17

Cluster characteristics. The main pages for Cluster 7 are the database book login, DBDesign, and chapters 1-3 for the database book. So Cluster 7 represents students working on database design.

Figure 9.18

Cluster discrimination. Discrimination compares clusters side-by-side. The default is to compare a cluster to its complement—things not in that cluster, but the drop-down box makes it easy to pick a second cluster. Here, Cluster 7 emphasizes database concepts instead of the general MIS book.





**Figure 9.19**

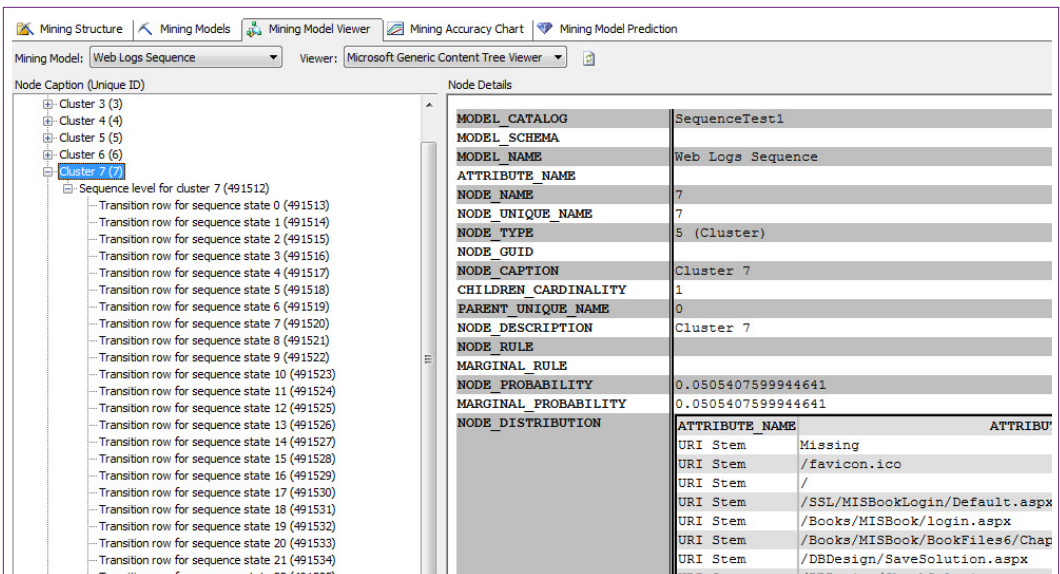
State transitions. Transitions are the probabilities of moving from one page to another. The result here is based on the population cluster or everyone. The drop-down box makes it easy to see how these transitions differ for each cluster or group of users.

### *State Transitions*

Microsoft Sequence Clustering uses a Markov chain to estimate parameters within sequences. A particularly useful parameter is the state transition or a probability associated with the next page to be selected. The result-viewer provides a graphical method to view these probabilities. Figure 9.19 shows the base diagram. This diagram is displayed based on the entire population cluster so it shows the page probabilities for a given page based on everyone. The drop-down box makes it easy to display the transition probabilities for any of the clusters. More than any of the other views, the transition probabilities provide an insight into the sequence or flow of the action. The diagram and the probabilities show how the pages are related to each other in terms of sequential flow. By switching cluster views, it becomes possible to see how each group navigates the Web site differently.

The detailed transition states are also stored within the Analysis database. The *Microsoft Generic Content Tree Viewer* retrieves these values and displays them in a table or list. To see the probabilities, use the Viewer drop-down-list to change to the generic tree viewer instead of the Sequence Cluster viewer. As shown in Figure 9.20, under each cluster is a list of the pages labeled sequence state 0 through 63. This state is the starting item or page. Selecting one of these entries creates a list of the target or next pages along with two main values: support and probability. The probability is the transition probability computed from the Markov chain for moving from the existing state (page) to the specified next page for individuals within the selected cluster. The support value is based on the number of cases in the training data that moved from the starting item to the targeted next item. However, the count value is modified by the probability of the cluster being selected so the support values are often fractions. Remember that a





**Figure 9.20**

Generic tree viewer. The tree viewer shows the transition probabilities overall and for each cluster. Choose a starting state in the left window (64 entries). The middle scrolling window lists the target states and associated probabilities for the selected cluster.

given item can be associated with multiple clusters—up to a level of probability. Even for a specified cluster, the transition probabilities form a large matrix—from each possible state (page) value to each possible target page. The tree viewer displays one row of the matrix at a time: Select a starting value and it displays each of the target values and the associated probabilities.

## Prediction

Prediction is not particularly useful, or supported, in sequence clustering. The transition state probabilities are the most useful for predicting behavior within a sequence. By themselves, the values are the conditional probabilities for each state. They are easily interpreted for a given a cluster or group, and a current page; they represent the probability of any moving to a specific page. Most Web sites are already designed to support certain types of flows. For instance, a product page supports search, then adding items to a shopping cart, followed by checkout pages. Users have the ability to choose options (such as return to add more products), so the transition probabilities show the chance that people within a given cluster will follow those paths. If some groups turn out to be more likely to follow detours, this information might be useful for altering the path options or finding ways to encourage others to follow them.

The clusters themselves are also defined in terms of probabilities. In the EM model, a group can be associated with multiple clusters—where the association is defined in terms of a probability. In traditional clustering, it is possible to enter values for dimensions and identify the clusters and probabilities associated with those specific values. This approach is difficult with sequences because it would be necessary to enter an entire sequence—not just a few dimension values.

## Other Tools

---

### What features are provided by other sequence mining tools?

The topic of data mining sequences is somewhat specialized. A few general tools exist, but almost all of them are commercial implementations somewhat similar to the Microsoft tools. Several add-ins also exist for the “R” statistical system to provide various tools for sequence analysis. However, most of the advanced sequence mining work has occurred in biology. In particular, the need to find patterns, align sequences, handle gaps, and other “messy” problems frequently occur in the analysis of DNA. Several specialized tools have been developed for these problems. Most are geared towards the DNA issues—notably a small number of items in the alphabet (CGAT), and potentially huge sequences. If a problem is encountered that has similar characteristics, it might be possible to define the problem in a form that can be handled by these specialized tools. For example, the U.S. National Institute for Health, specifically the National Center for Biotechnology Information) supports the BLAST (Basic Local Alignment Search Tool) tool (see the reference list for Web sites).

It is a little surprising that some of the DNA-specific features have seen little adaptation to business problems. However, the algorithms for biological research are highly optimized for their specific tasks in terms of performance and usability. The main task that they perform is comparing two or more sequences—in particular, the ability to search for a pattern from one sequence through a large collection of similar sequences. In DNA terms, a researcher might be interested in a specific pattern found in one or two sequences and then want to see if that pattern exists in other people.

In business terms, a close analogy might be found in terms of loan repayments. It is possible that a pattern of late or missing loan payments is an indicator that a person will file bankruptcy or fail to pay off a loan. If the sequence data existed for a large number of borrowers, it might be possible to search for similar patterns and then use those to identify future defaults. In a broader context, researchers have used similar techniques to examine young people moving through educational programs and into the workforce. So the tools have some value, but they can be difficult to apply to common business situations.

## Sequence Summary

---

The analysis of sequences is relatively new compared to most of the other tools. Much of the advanced research is taking place in biology in the form of DNA and protein-sequence analysis. These tasks are relatively specialized and the high-end tools developed for those tasks may be challenging to use in business research. Nonetheless, some business applications can benefit from analyzing sequences. A sequence is an ordered list of items or events. The ordering is an important characteristic of the problem. Additionally, the items must be discrete and relatively small in number of values. To be useful, the researcher needs a large collection of sequences that have random orderings.

The most common analytical approach to business sequences is to search for clusters or groups of people who follow similar sequences. A typical example is the paths taken by visitors to Web sites. A related typical example is the order in which products are added to a shopping cart; which generally also requires Web site data. Other examples can be found, but most of them involve customers to provide the random actions. Think in terms of states or events where there is a need to predict which event will occur next.



Sequence Clustering is similar to traditional clustering, where the goal is to find groups of actions that are similar to each other but different from other clusters. Using Microsoft Sequence Clustering, the key is that a Case table is needed that contains a single entry for each person or primary sequence. A second, Nested table contains the values of the sequence for each person where the value for each person at each point in the sequence is stored in a separate row. This nested table must have a primary key that provides the correct order of the sequence values. The column containing the sequence values must be specified as an input column, but never a key column. The tool limits the number of sequence values—by default to 64. This limit can be changed but Microsoft recommends that it be kept under 100. Problems with a large number of items, such as complex Web sites, should consider reducing the extraneous pages (such as GIF files) or perform multiple analyses on sections of the site.

The results of sequence clustering consist of two main components: (1) The cluster estimates, and (2) transition state probabilities. The cluster results are similar to other clusters but they need to be evaluated in terms of the items (pages) and the sequence of items. The transition probabilities are better at highlighting the dynamic nature of the sequence. They are derived from a Markov chain model and estimate the chance that a new item will be chosen next in the sequence given the current position. In Web site terms, if a person is viewing one page, the transition probabilities show the probability for each page that the person might move to next. These probabilities can be a general statement for everyone, or they can be found for each cluster or group.

## Geographic Analysis

---

**What role does location have in decisions?** Many people are familiar with the common mapping tools available on the Web, cell phones, and **global positioning system (GPS)** devices. These tools play a role in making some types of decisions, but business decisions are better supported by a more powerful **geographic information system (GIS)**. GIS tools include the basic mapping functions but focus on the ability to display layers of business, demographic, and technical data on top of the maps. In essence, the analytic decisions examine the question of how the business data elements are correlated due to location.

Increasing amounts of business data are available that are related to location. In the U.S., the federal government collects and distributes large amounts of demographic and economic data. For example, the demographic data available from the Census Bureau is tied to ZIP Code and often the more-detailed Census district codes. Consequently, basic information on consumers is readily available at many geographic levels. Also, several private companies collect detailed location data on businesses (particularly Google). Some sell the data and others sell geographic analyses using customized data sets. Local and state governments routinely use GIS tools to track services—such as utility (water) lines and crime incidents.

Many business decisions involve the use of location, such as siting for retail stores—which requires knowing the customer demographics of many areas. Building new factories raises similar issues of access to transportation (highways, ports, and railroads), as well as number, age, and educational level of potential workers. As more data becomes available, more complex questions can be addressed, including locations of specialty provider firms (such as machine shops), and environmental issues such as access to water, waste disposal, pollution levels, and so on.



**Figure 9.21**

Sample GIS color-coded layer. A GIS such as Microsoft MapPoint can display data within regions by color-coding the display. Here, Per Capita Income for 2007 is displayed at the County level. However, MapPoint can display only one color-coded layer at a time, so business data has to be displayed in a different form such as small charts or variable-sized circles.

Several governmental policy questions focus on geographic questions. For instance, health care potentially has many different regional differences. The Dartmouth Atlas of Health Care has some interesting charts online for comparing various national issues in health care. The term **geographic correlation** appears most commonly in health care research—where researchers test for differences in health care statistics across regions. The basic practice is to define a set of regions (such as hospital referral regions), collect data for various events within those regions and then compute and evaluate correlations for the various dimensions across the regions.

A GIS is different from a basic mapping system in that a GIS has the ability to display various data types on maps as layers. A **layer** is just one set of data tied to location. It might be displayed as a segment—such as a pipeline or travel path, a marker or pin, a small chart, or a color-coded region. Figure 9.21 shows a simple GIS presentation of per capita income at the county level, shown as a color-coded map. Microsoft MapPoint is used to display per capita income for 2007 at the county level. The data is from the Census Bureau and is built into MapPoint, making it relatively easy to display. However, MapPoint supports only one color-coded layer at a time. Any additional business data has to be displayed using small charts, pins, or variable-sized circles.

## Business Situation

---

**What types of business problems are related to location?** The location question is challenging to answer—at some levels almost any business decision has elements of geography or location embedded in the question. Selling almost anything to customers could be impacted by the customer's physical location. Even for online and digital sales, companies will want to track a customer's location—partly for taxation and currency issues; but also for longer-term questions of how to appeal to more customers in the same location.

The heart of a geographic analysis is that the data can somehow be tagged to a specific physical location. The location is not always specified down to an exact spot. As the availability of GPS data increases, more items can be identified to fairly tight locations. However, in many situations, location is identified at a wider level, such as by ZIP Code, county, or state. In fact, in most cases location at a specific point is too narrow. Rarely does a business need to know an exact spot—which might represent only a single person—instead, the area is more important because it represents a collection of potential customers, or employees, or suppliers. Along the same lines, location is typically viewed as a hierarchical concept—such as: Nation – State – County – ZIP Code. And managers often want to examine values at each level of the hierarchy—zooming in to see details when necessary.

In other cases, detailed local geography can be critical. For instance, consumers might be reluctant to cross a specific railroad track, highway, or river to shop at a store on the other side. A challenging aspect of geographical analysis is that local knowledge is often important for detailed analysis. At a higher level, sales by state or county are straightforward, but as the problem becomes more local, then additional information becomes useful to understanding the results. It is these areas where specialist firms hire people to collect detailed local data.

The bottom line is that many business problems are related to location. The key is to be able to collect the data with the associated location information. Sales are often easy to tag because individual stores or shipments automatically contain addresses. Even online sales generally collect address data. Likewise, purchases from suppliers and shipping data typically contain location information. Large companies that have production operations in multiple locations can also track geographical data on employees, production times, quality, and other measures that might be affected by geography or local demographic issues. Similarly issues in human resources that might seem to be unrelated to location can be affected by local cultures, such as overtime, sick days, and overall health costs. In some situations, data that might not seem to be related to location could be affected by other variables that are local—including culture, weather, income, and local demographics. In many cases, the only way to identify these effects is to look at the data and then search for explanations when unexpected differences do arise.

## Data

---

Probably the most important data source for geographical information is the U.S. Census Bureau. Census collects a vast amount of demographic and some economic data on a regular basis and all of it is geographically tagged. Almost all of the publicly-available data is available for download. Census also created some of the early mapping datasets through its TIGER mapping system—and much of this data is also available for download. Although, today, it is simpler to rely on

commercial software packages to draw the detailed maps—and they might use Census mapping data automatically. Census demographic data includes hundreds of series including “traditional” items such as age, gender, marital status, ethnicity, crime, and household size. It also includes economic and other data including income, employment, occupation, housing, industry data, commuting time, and weather. Census data is available at several geographic levels, including the Census District. The Bureau collects data at the household level but it is forbidden from releasing it until 70 years after it was collected. Interestingly, private companies have filled in some of the gaps. For example, Google routinely makes available the location of individual houses—including satellite photos; but Census (and the Post Office) is not allowed to provide this data. Another drawback to government-provided data is that release can be delayed—often for several years. For instance, the economic values from the Census might be two or more years out of date; which means that data collected every five years could be six or seven years out of date.

Companies also collect internal data—largely in terms of sales revenue and production costs. To handle this data geographically, it needs to be tagged or identified with location information. At the simple level, most data will have an address which includes the state or ZIP Code. ZIP Code level is probably the most detailed level available to most organizations. ZIP Codes are defined by the Post Office and loosely correspond to individual offices. Larger cities have multiple ZIP Codes that are organized around the physical Postal buildings. Smaller cities might include several cities within the same ZIP Code. The ZIP Code level is typically considered to be narrower than an individual city yet refers to a defined geographic area. The specific area is defined by the Post Office which defines the specific geographic region. GIS tools generally contain the mapping points for those regions.

The other way to tag data is to find a way to collect it along with GPS coordinates. GPS hardware prices have declined and are now embedded in many devices. However, collecting that data is a challenge—particularly in terms of privacy issues. Certainly, in-house data can be collected and stored by GPS-provided latitude and longitude. But data on customers can be problematic. Although cell phones collect GPS data, many people block this information (except to emergency providers—which cannot be blocked). Still, if a business knows the exact location of its stores or kiosks, then it knows the location of most customer interactions. Data at this level could be useful when examining detailed traffic patterns, and when sales are made at many small locations.

The Rolling Thunder Bicycle data will be used in the following sections to demonstrate some of the techniques of geographical analysis. In particular, the sales data is available at the city level. Although the data is simulated, it was generated with few specific patterns that can be investigated geographically. The database also contains valid Census data for several thousand cities which can be used to highlight comparisons.

## Model

---

The GIS approach is still commonly used to examine geographic data. The focus is on mapping the data to make it easier to see geographical patterns. For instance, sales by state might reveal stronger sales in southern or western states. This approach works well using color-coded shading when the focus of the analysis is a single data series. When multiple elements are being modeled, the maps can be-

come more difficult to create and read. For example, consider the Rolling Thunder Bicycle case and think about trying to display sales of bicycles. If the goal is to display just the total sales value of all bicycles, only a single series is needed and a color-coded region can display the relevant data. But what if you need to display sales by model type as well as location? Or what if it is important to display demographic data as well as sales, such as population or income levels? These additional dimensions are known as layers in geographic models.

In the old days when geographic data was largely handled manually, a layer of data was often created on a separate sheet—typically on acetate (clear plastic). Then any layer could be overlaid on the map—making it possible to stack up several layers to see multiple dimensions at the same time. Most GIS tools support a similar concept where additional dimensions can be added to any map. However, the issue of color-coded regions is a problem. Trying to display a second color-coded region on top of an existing one can make it difficult to see both sets of data. For this reason, few tools support multiple color-coded layers. It can work—it just requires careful consideration of the colors and intensities.

The issue of multiple dimensions is important in the analysis of most geographical data. In some cases, the effect of location is primary—perhaps sales near beaches or in the south are different from other locations. However, many situations require examining the correlation between multiple dimensions. For example, sales are likely to be correlated with per capita income or even population. Visually demonstrating these relationships requires creating multiple layers of data on a map.

A second type of correlation involves point data. A classic example is local data on crime statistics. For example, a police department could use push-pin markers to indicate the location of various crimes. One color could be used for drug busts, a second color for gun or knife attacks. If the two colors commonly appear together, it would indicate a correlation between the two types of crimes. It is relatively easy to plot these types of layers with most existing GIS tools. However, the point-location data is generally less useful for business applications because most businesses operate from a few fixed locations. But the tools could be helpful for companies with thousands of small locations (such as a coffee chain), or when tracking data through individual resellers such as convenience stores and other retail outlets.

The issue of data correlation is a critical element in geographical analysis. The GIS tools can make it easier to spot patterns, and they definitely make it easier to show others that a pattern exists. But it can be difficult to see correlations when the data dimensions are related to each other instead of directly to location. Examples are covered in the next section. But, the point is that sometimes traditional correlation or regression analysis is more useful. For instance, medical researchers rely on traditional statistics to analyze health data. The basic process is to examine the desired dimensions in terms of a specific region. The Dartmouth Atlas of Health Care defines hospital referrer regions (HRR) that correspond to major regional hospitals. Each hospital draws physicians and patients from a surrounding area. By collecting statistics within each region, it becomes possible to compare the data statistically.

Figure 9.22 shows a small example of correlation. Hypothetical data was created for sales and income. Both dimensions are plotted on the map with the income shown using region shading and the sales denoted by the size of a circle. Even with this small number of states, is it possible to see a correlation between



**Figure 9.22**

Sample correlation problem. The shading represents income and the circle size is defined as the sales. Is there a correlation between these two values? Perhaps, but it would be easier to see using traditional statistical tools. In this case, the simple correlation coefficient is 0.86 and a strongly significant regression coefficient from income to sales.

the two dimensions? Perhaps. But a simple statistical computation shows that the correlation is 0.86 and a regression analysis generates a strongly significant coefficient between income and sales. So, the statistical tests verify a correlation, and also provide a numerical measure and a test of the significance. The point is that there is a geographical relationship but it is actually due to a correlation between two dimensions, where one (income) is geographically related. The geographical aspect will become important later if the company wants to search for new sales territories—the data suggests focusing on high-income areas.

## Microsoft MapPoint

Several tools provide basic GIS support—many of them are expensive. Microsoft sells MapPoint for a comparatively reasonable price, and trial versions are available. The interesting feature of MapPoint is that it includes preconfigured Census demographic data. The Census data can be used along with your own data. The Rolling Thunder Bicycle data is useful for exploring MapPoint capabilities. The ZIP Codes in RT are accurate only to the city level, so it is better to start with the state data.

The first step in any analysis is to configure the data in the format needed for the tool. MapPoint (and other tools) make it relatively easy to read data in formats ranging from text files to databases. Because the RT data is already in a DBMS, the main step is to create a query to select and format the data. Microsoft Access or SQL Server are the easiest to use. Assume that managers want to explore 2012



```

SELECT Bicycle.SaleState, Sum(Bicycle.SalePrice) AS SumOfSalePrice
FROM Bicycle
WHERE (Bicycle.OrderDate Between '01-Jan-2012' And '31-Dec-2012')
GROUP BY Bicycle.SaleState;

```

**Figure 9.23**

RT sales by state for 2012. The query is straightforward and should be saved as a view or query so that the GIS tool (MapPoint) can open the query directly.

sales by state. Figure 9.23 shows the basic query to examine sales in terms of value (versus quantity).

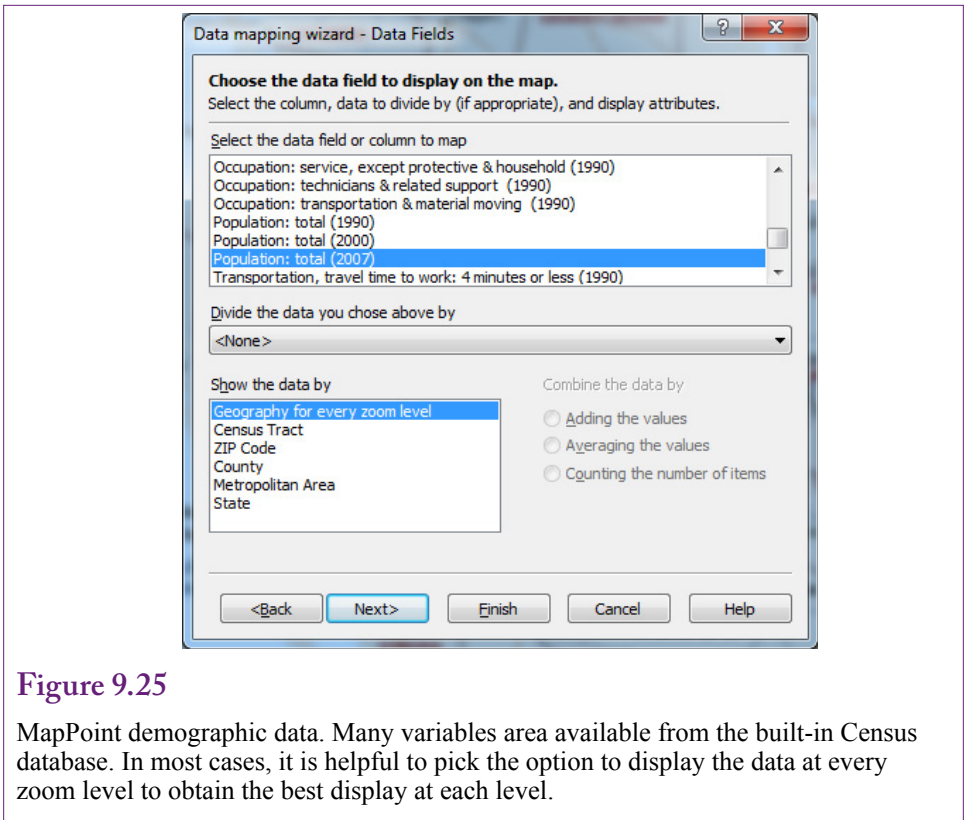
MapPoint has a Data Mapping Wizard that makes it straightforward to add data to a map. Set up the data in advance that contains a column with a location indicator. The location value should be at the most detailed level needed. MapPoint will provide any aggregation needed. For example, if data is available at the ZIP code level, use that and MapPoint will support aggregation to higher levels such as county and state. To begin, assume only one data series (sales) is going to be displayed. Start the wizard and accept the default choice of shaded area. Choose the option to link to the data (import is also acceptable but the link is dynamic). If the data is in Access, choose the Access option and find the database; otherwise pick the Link option and follow the prompts to connect to the database. Navigate and select the saved query. The wizard will automatically match the State column;

**Figure 9.24**

RT Sales by State for 2012 in MapPoint. The single dimension is relatively easy to see (once the road data is removed). But it is not immediately clear how the data are related to geography.







**Figure 9.25**

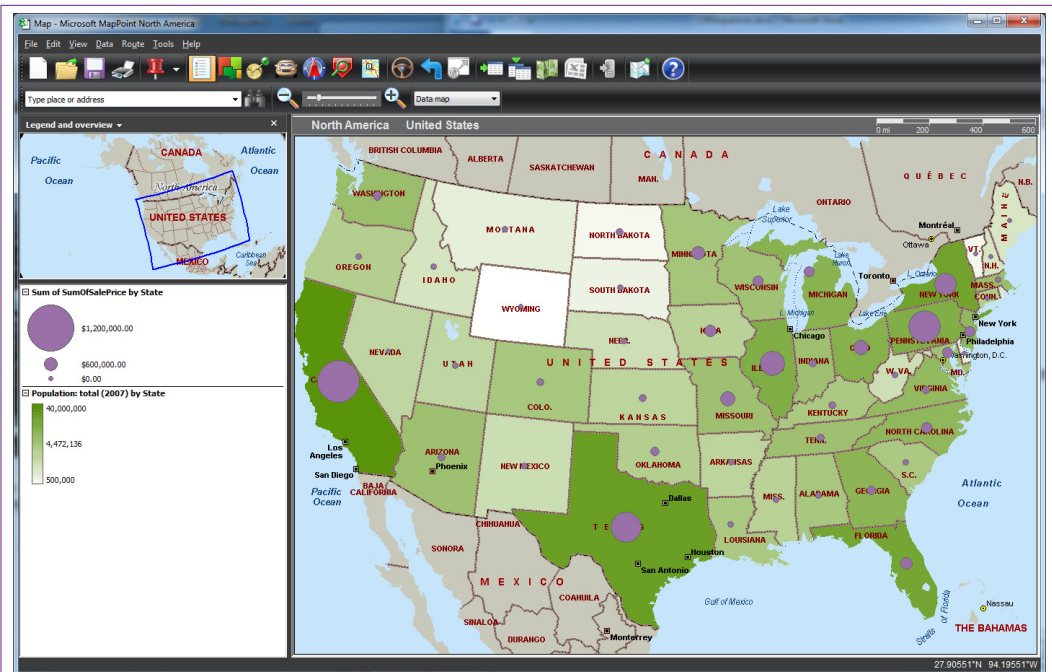
MapPoint demographic data. Many variables are available from the built-in Census database. In most cases, it is helpful to pick the option to display the data at every zoom level to obtain the best display at each level.

however, the database might include sales from Puerto Rico which has a code of PR, which is not considered a “state” by MapPoint (although it is a U.S. territory). So choose the option to skip the record.

Figure 9.24 shows the MapPoint chart. The chart is interesting, but it is not immediately clear how the sales data is related to geography. After a little thought, it appears that the states with the most sales are also the ones with the most population—which is not too surprising. So, it might be interesting to add a layer for population.

MapPoint has a large amount of Census data including population. The process is similar to adding your own data. However, MapPoint can use a shaded region for only one dimension. The second will have to be added using a different notation, such as a sized circle. So, it is necessary to think about which dimension should be shaded and which one as a circle (or other indicator). In this situation, the income can be shown as the background using the shading and then put the sales in the foreground. It is easiest to start over with a new map. Start the wizard but choose the option to add demographic data. Figure 9.25 shows the main step in selecting Population (Total 2007). It also shows the option to add the data for all zoom levels, which enables MapPoint to adjust the display for any point in the hierarchy.

After the demographic data has been added, restart the wizard to add the Rolling Thunder Bicycle data. Options for the display include Shaded Circle, Sized circle, Pie chart, Column chart, and push pins. The Sized circle is probably the easiest to see in this case. Figure 9.26 shows the resulting map and it is possible



**Figure 8.29**

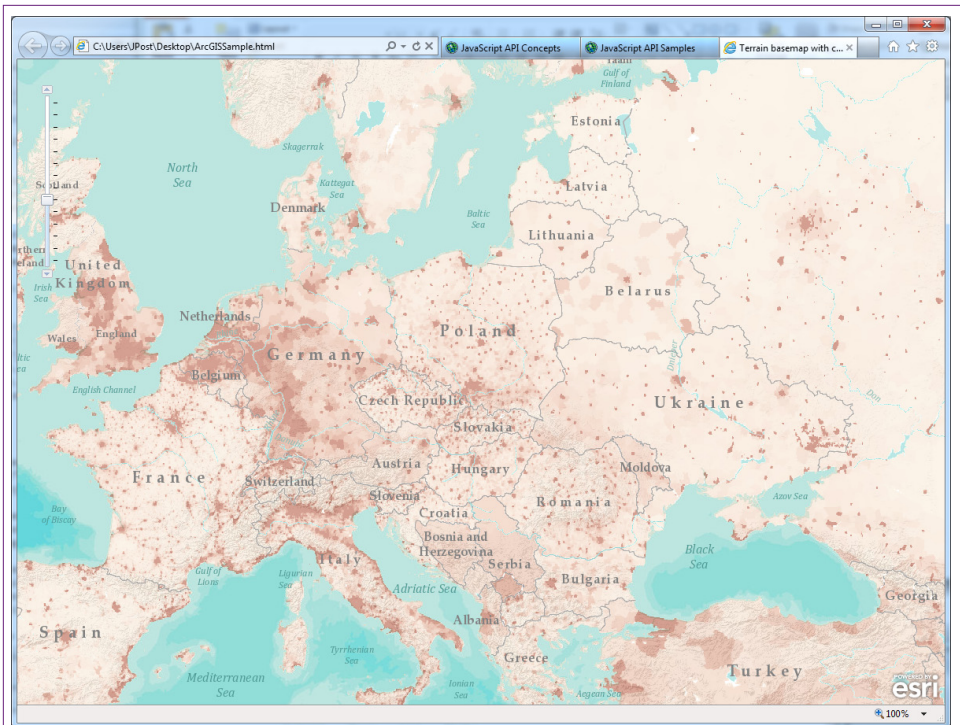
RT Sales versus population. The correlation between population and sales is relatively easy to see where larger circles appear in the states with higher populations (darker shades).

to see the correlation between sales and population. The correlation might be even easier to see if both layers could be presented as two differently-colored shaded regions, but MapPoint supports only a single layer as a shaded region. To support multiple layers might be useful but it requires colors and layers that support transparency so that the bottom layers show through to a certain extent; and it requires being cautious in its application. Note that the road layer was removed to make it easier to see the shadings. The individual layers can be edited by right-clicking the entry in the map key on the left side. The pop-up menu items support changing colors or even changing the data selection or display method.

Additional layers can be added using the same wizard, but beyond two or three layers, it becomes difficult to see the individual items. Multiple layers or dimensions work best by using push pins with a different color for each layer. This approach can highlight correlations between the dimensions by counting the number of times the colors appear together. But, it will work best at more-detailed location levels such as county, city, or ZIP Code. A comparison of only 48-50 states is usually not enough observations to highlight differences.

## Other Tools

**What tools exist for analyzing geographic data?** In particular, what online tools are available? Microsoft MapPoint was used in the earlier sections because it is available at a relatively low cost and is straightforward to use. However, several other tools offer additional features. The major commercial GIS



**Figure 9.27**

Esri ArcGIS map created with an HTML/Javascript file. The HTML file comes from the Esri documentation and uses Esri servers to load the base map and the population data layer. The country and names data are loaded as a third layer.

tool is sold by Esri, and it is heavily used by large organizations and government agencies.

Largely led by Google and its purchases and research in online tools, many people have turned to online GIS facilities. Google and Microsoft's Bing, the leading search engines, both have extensive mapping tools that can be integrated into other Web sites. Both are free to explore with documentation and sample code online. Both also charge for extended use of the services. Commercial Web sites with millions of visitors could end up paying substantial amounts of money if everyone uses the online mapping tools. However, a site with that many visitors could probably make money through sales or advertising. More recently, Esri has added online services as well—many of them can be used for free; and Esri tools provide detailed options that might not be available with the other tools.

## Esri

One of the earliest and most powerful GIS tools is ArcInfo by Esri. The current product name is ArcGIS and it is available as a standalone desktop standalone software tool or on online Web-based system. ArcInfo/ArcGIS is commonly used in government organizations. It can handle complex graphing and large datasets, but is relatively expensive. On the other hand, the company has been increasing access to free offerings online, including base maps such as USA topographic, ocean, and the National Geographic World map. The online system can be inte-

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=7,IE=9" />
<!--The viewport meta tag is used to improve the presentation and behavior of the
samples on iOS devices-->
<meta name="viewport" content="initial-scale=1, maximum-scale=1,user-scalable=no"/>
<title>Terrain basemap with custom data</title>
<link rel="stylesheet" type="text/css" href="http://serverapi.arcgisonline.com/jsapi/
arcgis/3.0/js/dojo/dijit/themes/claro/claro.css">
<style>
html, body { height: 100%; width: 100%; margin: 0; padding: 0; }
#map{padding:0;}
</style>
<script type="text/javascript">var djConfig = {parseOnLoad: true};</script>
<script type="text/javascript" src="http://serverapi.arcgisonline.com/jsapi/arcgis/?v=3.0">
</script>

```

### Figure 9.28

Initialization of the Esri HTML and Javascript. The first few tags simply define the location of the Esri server script and style files.

grated into Web sites to display basic location data such as a identifying a store location. More sophisticated data integration requires a subscription. Esri also has substantial amounts of geocoded data, including Census demographic data. But most of this data requires a subscription to use. Esri also has an “analytics” package to help integrate company data with the Esri databases. The free online version makes it possible to add up to 1,000 points to a standard Esri map and share that map with others or make it part of a Web site. The process is similar to that used by Google.

One of the nice features of ArcGIS is the ability to set the opacity (conversely the transparency) of a layer—allowing lower layers to show through. This feature also works with the online tools. Figure 9.27 shows a sample map created using Esri’s documentation. It contains three layers: The base map of continents and water features, population shown as shaded regions, and the political boundaries and names. The interesting feature of the map is that it was generated from a single HTML file using Javascript API commands to the Esri servers to obtain the base map and data layers. It is also possible to add custom data instead of the population values. Most applications today use **application programming interfaces (APIs)** or function calls to pass information from code to the server.

Figure 9.28 shows the first part of the HTML/Javascript file to create the Esri map. The first steps are largely html commands to identify the style sheet and location of the script files on the Esri server.

Figure 9.29 shows the main Javascript commands to handle the mapping tasks. The bottom (first) layer is simply the base map that is provided from Esri servers. The top (last) layer is also from an Esri server that provides the national boundaries and names. These two layers (or similar ones) will likely be a part of almost any map you generate. The middle layer retrieves and displays the population data using shaded regions. It comes from a separate Esri server used for demonstra-

```

<script type="text/javascript">
dojo.require("dijit.layout.BorderContainer");
dojo.require("dijit.layout.ContentPane");
dojo.require("esri.map");
var map;

function init() {
var initExtent = new esri.geometry.Extent({"xmin":-5.54,"ymin":40.81,"xmax":44.46,"ymax":
58.35,"spatialReference":{"wkid":4326}});
map = new esri.Map("map",{ extent:esri.geometry.geographicToWebMercator(initExtent)
});
//Add the terrain service to the map. View the ArcGIS Online site for services http://
arcgisonline/home/search.html?t=content&f=typekeywords:service
var basemap = new esri.layers.ArcGISTiledMapServiceLayer("http://server.arcgisonline.
com/ArcGIS/rest/services/World_Terrain_Base/MapServer");
map.addLayer(basemap);

//add custom services in the middle. This is typically data like demographic data, soils,
geology etc.
//This layer is typically partly opaque so the base layer is visible.
var operationalLayer = new esri.layers.ArcGISDynamicMapServiceLayer("http://
sampleserver1.arcgisonline.com/ArcGIS/rest/services/Demographics/ESRI_Population_
World/MapServer", {"opacity":0.5});
map.addLayer(operationalLayer);

//add the reference layer
var referenceLayer = new esri.layers.ArcGISTiledMapServiceLayer("http://server.
arcgisonline.com/ArcGIS/rest/services/Reference/World_Reference_Overlay/
MapServer");
map.addLayer(referenceLayer);

dojo.connect(map, 'onLoad', function(theMap) {
//resize the map when the browser resizes
dojo.connect(dijit.byId('map'), 'resize', map,map.resize);
});
}

dojo.addOnLoad(init);
</script>

```

### Figure 9.29

Esri Javascript to load three layers for the map. The bottom (first) layer is the Esri base map. The top (last) layer is the Esri reference layer that loads the country boundaries and names. The middle layer is the data layer that shows the population, which could be replaced with a custom service.

tions. This data layer would be replaced by calls to your own data source to plot custom data.

Figure 9.30 shows the final HTML needed to display the actual map. The work is handled by the two <div> tags by using the Esri-specific “dojotype” attribute. Almost all of the work is handled automatically by the Esri pre-written scripts. Defining your own data source takes a few more steps, but the Esri documentation shows how to load data from a variety of sources, including sample text data from online Web sites. Note that the HTML file itself can exist on almost any device. For example, the lines can be copied and pasted into a text file and the graph will be displayed simply by opening the file in a browser.



```
</head>

<body class="claro">
<div dojotype="dijit.layout.BorderContainer" design="headline" gutters="false"
style="width: 100%; height: 100%; margin: 0;">
<div id="map" dojotype="dijit.layout.ContentPane" region="center"
style="overflow:hidden;">
</div>
</div>
</body>

</html>
```

### Figure 9.30

Simple HTML for the Esri map file. The remaining portion of the HTML file simply defines the divs needed to display the actual map.

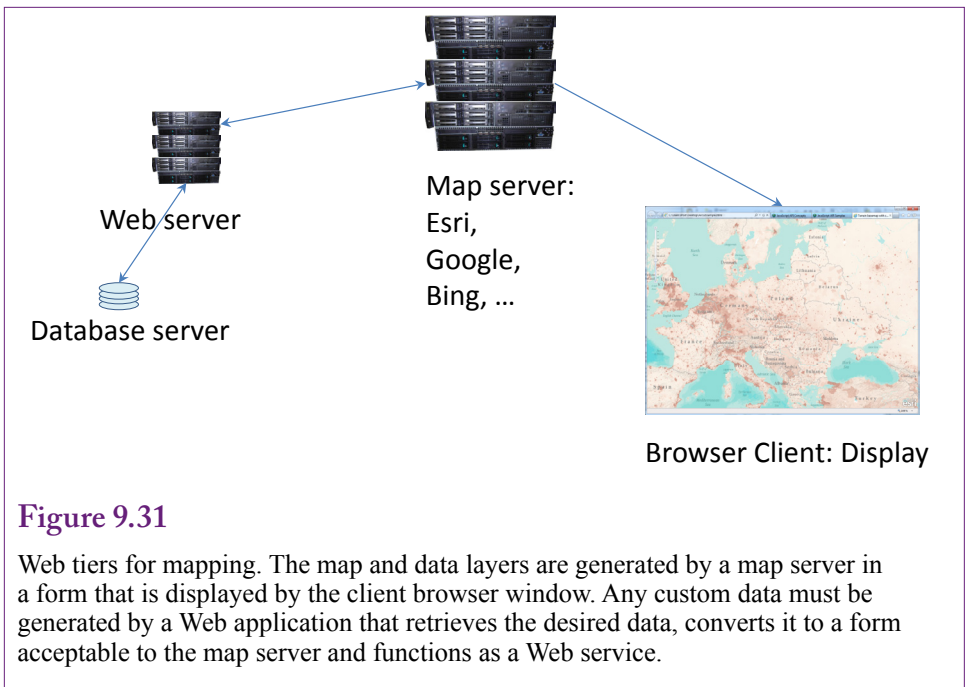
## Google

Google is probably the best-known online map provider, and Google continues to add features to its tools. It also provides the desktop Google Earth application which supports more interaction than the standard online maps. In terms of GIS capabilities, Google supports several online tools to embed maps in Web sites and add points of interest. To a point, Google tools are provided free of charge. But even for publicly-available data, the tools impose limitations—largely on the number of daily users. The developer Web site (<https://developers.google.com/maps/licensing>) lists the details.

Google also uses HTML and Javascript to pass data to the servers and display maps. Documentation is provided on the Google maps Web site. One catch is that you have to sign up for a personal account first to obtain a personal key value. A second issue is that it is relatively easy to display markers (icons) and paths or routes on Google maps. It is more challenging to create filled areas (using polygons) to display data. Consequently, most online Google maps are basic maps with markers.

Google has many other online tools and some of them are being integrated into the mapping environment. In particular, a Fusion Table is a simple row-column database table that can be stored online. A Fusion table that holds location data can be used to generate a layer of data for a Google map. For instance, a Fusion table could be created that lists the location retail stores for a company (based on its address). Google maps can plot the location of each of the stores simply by passing it the data from the Fusion table. Some data limits are imposed, such as no more than 100,000 rows of data for mapping and a limit of one million bytes of data.

The terminology, APIs, and integration are unique to Google. Documentation is available online and some samples exist to help learn the system. However, Google is also known for continually “upgrading” its capabilities, so expect changes over time. Additionally, you will have to register for a personal account to obtain a special key that must be included in all of your code. This key value is used to monitor traffic and usage so that if the limits are exceeded, you can be billed for the additional data usage.



**Figure 9.31**

Web tiers for mapping. The map and data layers are generated by a map server in a form that is displayed by the client browser window. Any custom data must be generated by a Web application that retrieves the desired data, converts it to a form acceptable to the map server and functions as a Web service.

## Bing

Microsoft's Bing maps has similar features to Google and Esri. In particular, you have to sign up for an account and use the AppID in all of the files. Accounts with high numbers of transactions will be billed. The overall development process is similar to Esri and Google—develop HTML and javascript code as a Web page that connects to the server. Documentation is available at <http://www.microsoft.com/maps/developers>. One useful tool is an AJAX Control 7.0 which makes it easier to find the special APIs and install them into your file. Selecting items from a list automatically install the corresponding API code into your file, which you then customize for your specific needs.

Microsoft also provides many other tools, including compilers and databases, including the online Azure SQL tools. Although SQL Server supports spatial data types, as of 2012, Azure does not automatically connect to Bing maps. However, several third-party companies provide connectors to facilitate the transfer and conversion of data into a form that Bing maps can handle. Search the Codeplex sharing site for examples such as the `ajaxmapdataconnector` that works with Microsoft's AJAX 7.0 API tool.

A key step to remember whether using Google, Esri, or Microsoft is that Web mapping involves three major elements: (1) The client browser, (2) The map server, and (3) A data server. Figure 9.31 applies to all of the online mapping services. To provide custom data on a map, you need to develop a Web application that runs on your own server to retrieve data, format it for the map server, and provide it on command as a Web service. In many cases, this step requires the help of a Web server programmer.



Census Bureau  
<http://www.census.gov>

USGS: United States Geological Survey  
<http://www.usgs.gov>

Geospatial  
<http://geo.data.gov>

FAA  
<http://aeronav.faa.gov>

NOAA: National Oceanic and Atmospheric Administration  
<http://www.nauticalcharts.noaa.gov>

NGA: National Geospatial Intelligence Agency  
<http://msi.nga.mil>

Others: Many agencies maps to display data

Local/County: Many use Esri to display property maps and local data. Search for the county assessor.

Canada  
<http://www.fedmaps.com>

United Nations  
<http://www.un.org/Depts/Cartographic/english/htmain.html>  
<http://www.grida.no/graphicslib>

### Figure 9.32

Government sources of maps and data. Some online data and maps are free but watch the dates. It costs time and money to update map data so some sources take time to get current data.

## Federal Government

Several government agencies have begun providing mapping data online. In many cases, this data is already integrated with standard mapping services—particularly Esri. In a few cases, the maps are created automatically, but with limited options. For example, the Census Bureau produces several maps to highlight interesting data. The government access portals have changed over time so custom searches might be needed to find specific data and maps. However, the geospatial site <http://geo.data.gov> is designed to handle some standard mapping tasks.

For many years (centuries), the U.S. Geological Survey (USGS) has been responsible for producing accurate maps of various features. In particular, the USGS is well-known for its detailed topographical maps. Most of these maps were printed and sold at various locations. Most of the printed maps are still available, and the large format makes them convenient for many purposes. However, the USGS is in the process of converting the maps to digital format. A few private companies resell this data for GPS units. The USGS is working to provide the topo maps for free download. The maps show detailed points of interest, but are often out of date. Still, they can be critical when examining and displaying data at a very local level of detail. Figure 9.32 lists some of the common sources of government maps. Some of the online maps and data are free, but it takes time and money to update and create maps so current, high-value data often carries a charge. One interesting source of map data is not a government agency but an organization that uses crowd sourcing to collect and share GPS data: [www.openstreetmap.org](http://www.openstreetmap.org).

By itself a basic map is not a GIS and it is not a geographical analysis. However, base maps provide the foundation for many of the displays. Placing your data on a map with the proper content and level of detail will make it easier to see and

understand the analysis. Hence, it helps to have multiple choices when selecting a base map.

## Geographic Summary

---

Many business decisions are related to location. In particular, almost any data related to customers or suppliers can be tied to location—even if only at a national or state level. In one sense, every organization constantly searches for new customers, so it makes sense to search for new customers in the same location as existing customers. More broadly, the phrase “same location” actually means locations with similar demographics. So a key aspect of geographical analysis is to identify common demographic traits among existing customers and then find other areas that have similar demographics. In the U.S., the Census Bureau demographic data is useful for estimating customer characteristics and for identifying similar locations across the United States.

Geographic analysis goes beyond basic mapping. The analytical component often consists of finding correlations across dimensions and demographics. Data variables can be directly correlated with geography—such as latitude, access to water, days of sunshine, and so on. Or, multiple data sets can be correlated with each other through their location. For example, sales might be tied to a certain level of per capita income and people with that level of income might tend to live in specific areas—such as suburbs. The main key to beginning a geographic analysis is to collect and tag data by location. The location can be highly detailed—such as GPS coordinates; or simpler items including address, ZIP Code, city, or state. GIS tools display the data on a base map, using shaded regions, markers, or charts. The displays are useful for visualizing relationships, but statistical analyses on the data by region can provide more detailed estimates and measures of significance.

## Key Words

---

application program-  
ming interfaces (APIs)  
bins  
classification  
clustering  
comma-separated-values (CSV)  
discretize  
edit distance  
gap

geographic correlation  
geographic information system (GIS)  
global positioning system (GPS)  
layer  
Levenshtein distance  
Markov chain  
sequence  
transition probability

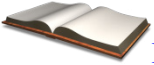
## Review Questions

---

1. What are the two primary types of sequence analysis?
2. What are the key data elements needed to perform sequence clustering analysis?
3. Why do sequence values need to be discrete instead of continuous?
4. How is missing data handled in sequence analysis?
5. What are the sequence and random elements available from Web server logs?
6. With Microsoft Sequence Clustering what are the roles of the Case and Nested tables?
7. With Microsoft Sequence Clustering is a sequence uniquely identified with a single cluster?
8. What are the differences between cluster views in Microsoft Sequence Clustering?
9. How do state transition probabilities reflect the dynamic sequence?
10. How is geographic analysis different from a mapping system?
11. How is indirect geographic correlation different from direct correlation?
12. What is the standard hierarchy for location data?
13. Why is it important for GIS tools like MapPoint and ArcGIS to include demographic data?
14. Which GIS tools support layer opacity to make it easier to display multiple data series?
15. What basic tools are needed to provide custom data to an online mapping system for use by your customers?

## Exercises

---



### Book

1. Set up and run the sequence clustering example using the Web site data. Write a brief report to the site manager about the results.
2. Research how the BLAST tool is used in DNA research and summarize the main characteristics of the types of problems it solves. Provide a business or social (non-biological) example where the tool might be used if it can be modified.
3. Explain how sequence mining might be used in a university to help students.
4. Select a business category and briefly describe how sequence analysis could be used to help a company in that business. Identify the data that would need to be collected.
5. Use an online tool (Esri, Google, or Bing) to create a map with markers for at least three locations that are of interest to you.
6. Find two government data series and identify the geographic levels available.
7. Research Esri's online ArcGIS and briefly explain the command needed to retrieve text data from an existing Web site and display it on a map.



### Rolling Thunder Database

8. Compute total sales by month for each year. Let each year become a sequence, discretize the sales data (try using standard deviations). Use sequence analysis to see if clusters exist.
9. Similar to the previous question, compute total sales by month by model type for each year. Each model type year is a separate sequence. Count the number of bikes and discretize the data. Use sequence clustering and include the ModelType as an additional input dimension to see if any patterns exist.
10. Use MapPoint or similar tool to explore correlations between sales value, population, and income.
11. Use MapPoint or similar GIS tool to see if there are differences in sales of model types based on elevation (altitude). That is, do people in mountainous regions buy more mountain bikes than road bikes?



## Diner

12. The diner data contains a DOW (day of week) column. Use that column as the sequence order and compute total sales by DOW for each week in the year. Each week represents one sequence. Discretize the sales data—such as by using standard deviations. Run sequence clustering to find any patterns in weekly sales data.
13. Find state or regional data on the amount of money consumers spend on restaurant meals to help find a location for a new diner.



## Corner Med

14. Count the number of procedures by each employee on each day and create a day-of-week (DOW) column for the date. Essentially, build a sequence for each employee for each week by DOW to count the number of procedures performed. Analyze those sequences by Week (case) to see if any clusters or interesting patterns exist.
15. Try to find health care statistics and use them along with income data to help identify potential locations for new Corner Med outlets. You can search nationwide or within a specific city.



## Basketball

16. Compute the total points scored by a team by game for a season and run sequence clustering on the teams to see if any groupings exist. Add dimensions for whether the team made the playoffs and the division or region to see if they affect the clustering.
17. Compute total points scored by person by game for a season. Be sure to include missing values for games not played by a specific person. Run sequence clustering to see if groupings exist and comment on the results.
18. Compute the average points scored per game for the main NBA Divisions and plot them using a GIS tool to see if there are regional differences. Hint: Assign states to the regions.
19. Find the teams that won the final championship game for the past 20 years or so. Build a table with the list of the cities and the number of times each has won. Plot the results with a GIS and comment on them.



## Bakery

20. The bakery SaleItem table includes the column Seq to indicate the order in which items were purchased. Assume this value is accurate and use sequence clustering to see if patterns exist in the order in which items are purchased. Use the product category instead of item name to reduce the number of item values.
21. Using online directories choose two similar-sized cities in two different states and count the number of bakeries in each city. Combine counts from other students and see if there is a significant regional difference. Based on the data where might you open a new bakery?



## Cars

22. Find data on car sales by location and plot the data using a GIS. Hint: NADA has sales by state.



## Teamwork

23. Have each person in the group find one additional tool that can be used to analyze sequence data. Briefly explain the purpose, the type of data, and types of analyses. Combine the reports and as a team identify the tool that offers the most usefulness to business. Include SQL Server in the final comparison.
24. Find at least three studies that use sequence analysis. Hint: Try research studies if you cannot find business results. Briefly summarize the data, the tool used, and the results found.
25. Find at least three studies or examples that use geographic correlations. Briefly describe the data, how the data was collected, and the results. Which study provides the best example of geographic analysis?
26. Select a local region and using a GPS unit (possibly on a phone), obtain the location of all diners and restaurants within that region. Plot the results on a mapping system. (Using Google Local or similar tools is cheating.)

## Additional Reading

---

Dong, Guozhu and Jian Pei, 2007, *Sequence Data Mining*, Springer: New York. [A relatively short but technical (mathematical) reference for sequence mining methods.]

Esri documentation center: <http://edn.esri.com/index.cfm?fa=doclibrary.gateway> [Links to the online documentation for Esri products.]

Google documentation for maps. <https://developers.google.com/maps/documentation/javascript/> API documentation: <https://developers.google.com/maps/documentation/> [Links to using V3 of Google maps.]

Microsoft documentation for Bing maps. <http://msdn.microsoft.com/en-us/library/dd877180.aspx> [Includes links to the latest AJAX control.]

National Institute of Health: <http://blast.ncbi.nlm.nih.gov/>. Downloads at: <ftp://ftp.ncbi.nih.gov/blast/executables/>. For a description see <http://www.ncbi.nlm.nih.gov/books/NBK21097/>. [BLAST tools for aligning sequences—notably genomic/DNA sequences.]

Trevor Hastie, Robert Tibshirani, and Jerome Friedman, 2001, *The Elements of Statistical Learning*, Springer: New York. [An outstanding book on data mining, with an emphasis on theory. A graduated-level book that requires a strong mathematics background.]



## MDX

### Chapter Outline

Introduction, 482	501
OLAP Cube Review, 482	<i>Conditions with IIF and CoalesceEmpty,</i> 502
<i>Dimensions and Hierarchies, 484</i>	<i>TopCount Function, 504</i>
<i>Rolling Thunder Bicycles Cube, 484</i>	<i>Year to Date, 505</i>
Definitions and Concepts, 487	<i>Moving Averages, 507</i>
Main Syntax, 489	Summary, 509
Basic Examples, 490	Key Words, 510
<i>A First Example, 491</i>	Review Questions, 511
<i>Adding a WHERE Condition, 492</i>	Exercises, 512
<i>Displaying Specific Dimension Values,</i> 494	Additional Reading, 515
<i>Cross Join, 494</i>	
Calculated Measures, 495	
Complex Computations, 496	
<i>Percentages, 497</i>	
<i>Compute Changes, 498</i>	
<i>ParallelPeriod Function, 499</i>	
Some MDX Functions, 501	
<i>EXCEPT: Taking Values Out of Totals,</i>	

### What You Will Learn in This Chapter

- How is data retrieved from a cube and used in calculations?
- What are the basic elements of an OLAP cube?
- What are the main objects in MDX queries?
- What is the primary structure of an MDX query?
- How are MDX queries written and what basic data do they provide?
- How are computations and new measures defined?
- How does MDX handle complex computations that cross levels or rows of data?
- What other MDX functions are commonly used in business problems?

### **Dallas Cowboys Merchandizing**

Merchandising is an important element for any sports team, including those in the National Football League (NFL). The Dallas Cowboys are no exception—in fact, Bill Priakos notes that “nobody gives out their exact numbers, but we feel comfortable we are in the upper echelon,” along with teams like Manchester United and the Chicago Bulls. As one of the top merchandizer, the Cowboys sold more than \$100 million worth of gear—shirts, jerseys, and other items in 2009. The Tony Romo jersey was estimated to be the most popular sports jersey in the nation in 2008—selling half a million items alone. But, as any clothing retailer knows, retail sales are challenging. In fact, all of the NFL teams except the Cowboys outsource all of the retail operations to Reebok. In 2002, the Cowboys chose to control all manufacturing, sales, and distribution themselves. The merchandizing organization installed software from Microsoft to handle basic sales and data tasks, but ended up buying software from Tableau Software, Inc. to answer the harder questions. Creation of a digital dashboard was a key element in tying together all of the underlying data. Priakos notes that the software lets “us find answers instantly.” Such as answering “How are Internet sales of our jerseys doing?” or “Where do our jerseys sell well outside of Texas?” Without the dashboards and visualization software from Tableau, it used to take 30 minutes to create queries to answer these types of questions. [Lai 2009]

Digital dashboards are constructed from key performance indicators and many of the steps can be automated using MDX tools.

Eric Lai, “BI Visualization Tool helps Dallas Cowboys Sell More Tony Romo Jerseys,” *Computerworld*, October 8, 2009. [http://www.computerworld.com/s/article/9139140/BI\\_visualization\\_tool\\_helps\\_Dallas\\_Cowboys\\_sell\\_more\\_Tony\\_Romo\\_jerseys](http://www.computerworld.com/s/article/9139140/BI_visualization_tool_helps_Dallas_Cowboys_sell_more_Tony_Romo_jerseys)

## Introduction

---

### How is data retrieved from a cube and used in calculations?

**Multi-dimension expression or MDX** is a query language for OLAP cubes. It appears to have been initially defined by Microsoft but has been adopted by almost all of the major data mining tool providers, including SAS and IBM. Documentation and examples can be found on the Web both from official (corporate) sources and individual writers. The big question is whether you need to know how to write (or even read) MDX queries. For most business users of data mining, you can simply use the browsing tools created by various vendors and explore cubes with the graphical designers. Microsoft's PivotTable—widely available in Excel—is a good example of a tool that is easy to use with drag-and-drop features.

In some ways, this chapter is optional. However, it turns out that the foundations of MDX are relatively straightforward; and some advanced problems might be easier to solve using MDX directly instead of trying to force a graphical browser to do what you want. Also, some situations call for integrating OLAP data into other tasks, such as spreadsheets, Excel, or other tools. In these cases, the power of MDX is useful because it can be written as a text query, passed to the server, and the results returned directly to an application for further analysis or processing.

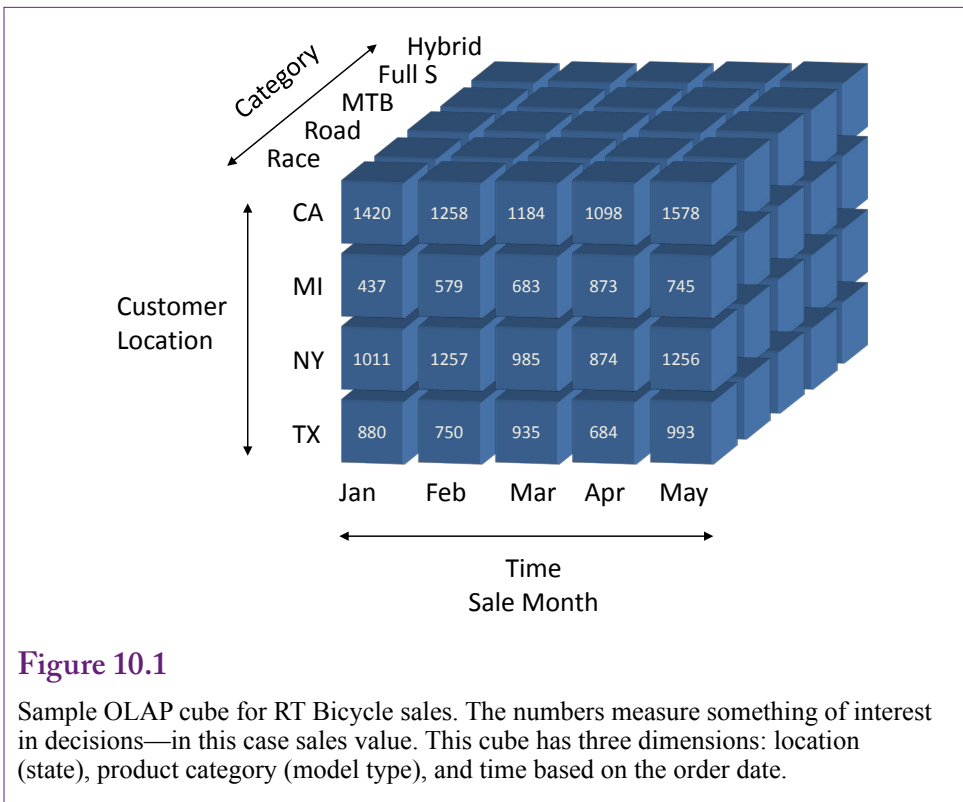
A few writers have suggested that MDX is similar to SQL, but these two tools are completely different. Confusingly, MDX uses similar keywords (SELECT, FROM, WHERE), but they have no relationship to anything in SQL. More importantly, the goals are completely different. SQL is designed to retrieve data from relational tables. MDX is designed to retrieve data from an OLAP cube. Consequently, one of the first steps in this chapter is to review the concepts of OLAP cubes and to create a sample cube. A key feature of OLAP cubes is that they are designed to work with aggregate data—or multiple subtotals. This difference is critical to MDX—all queries are designed to return subtotal data—and you almost never enter a SUM function; so the process is automatic.

This chapter is relatively short. It reviews basic concepts and terminology for OLAP cubes. Then a section highlights the main concepts in MDX—largely in terms of hierarchical dimensions. The main syntax of MDX is relatively short, but somewhat confusing. The best way to understand MDX is to work with examples. This chapter uses the Rolling Thunder Bicycle company database to generate a cube and show how to apply MDX. The examples use Microsoft SSAS (SQL Server Analysis Services) and Visual Studio to generate the OLAP cubes and process the MDX commands. Because of changes in SSAS and Visual Studio, PivotTables in Excel are also used to provide better displays of the cubes.

## OLAP Cube Review

---

**What are the basic elements of an OLAP cube?** Recall the main concepts of OLAP cubes from Chapter 3. A **cube** is a way to visualize and explore data relationships. The main data in a cube consists of **measures** which represent variables that are important to the decision. Common business measures include sales revenue, profit, cost, and counts of items sold. Measure variables have to be numeric. A cube presents a way to quickly explore subtotals of the measure data relative to various attributes. Attributes can be derived from a sale (location, date), the product itself (color, size), the customer (age, income), or internal data such as salesperson or division. Ultimately, the question is to examine how the measure subtotals vary based on the different dimensions. For example, are sales in some states consistently higher than those in other states?

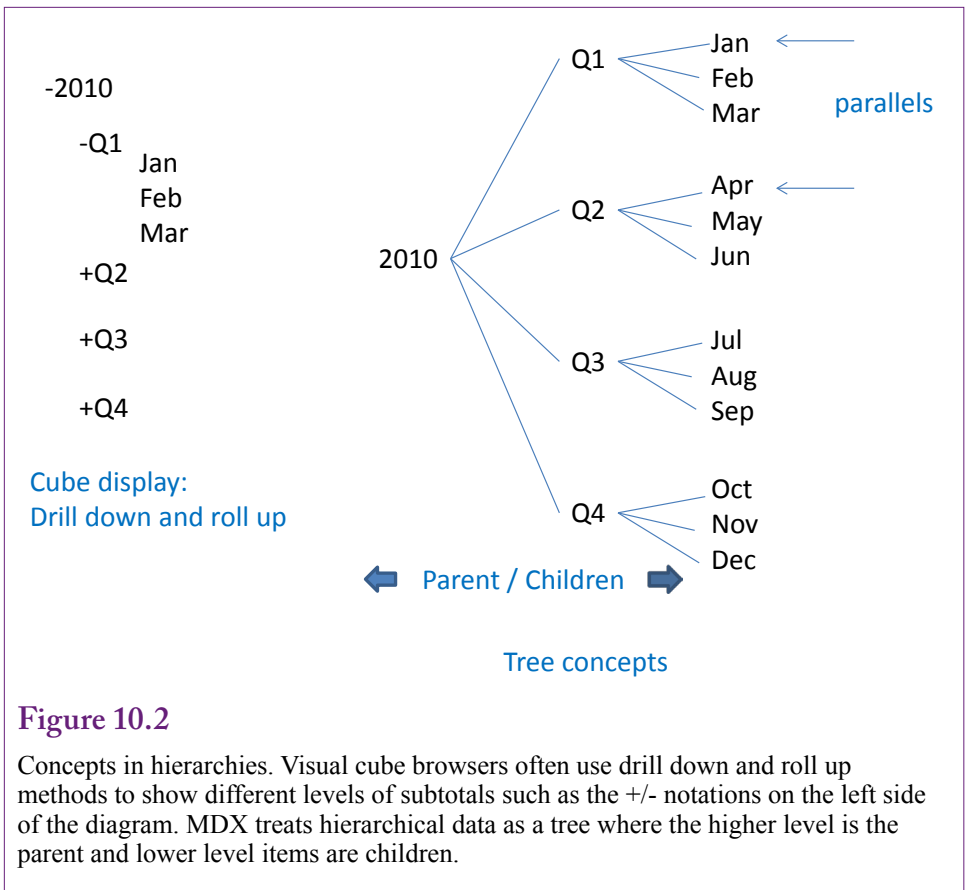


**Figure 10.1**

Sample OLAP cube for RT Bicycle sales. The numbers measure something of interest in decisions—in this case sales value. This cube has three dimensions: location (state), product category (model type), and time based on the order date.

Figure 10.1 shows a sample cube for Rolling Thunder Bicycles. The measure consists of sales value. The three dimensions pictured are: (1) product category or bicycle model type, (2) customer location or state, and (3) date or time of the sale—expressed in months. The values within a single cell of the cube show the sales value for a specific category to a given state in a specified month. Hyper cubes can have many dimensions—but it is difficult to draw anything above three dimensions. The point is that attributes are displayed as **dimensions** on the cube. Analysts can then use tools to choose dimensions, examine individual values on specified cells, and compute subtotals across dimensions (such as all states for a given month). It is important to remember that the cube displays subtotals.

A key aspect of dimensions is that many of them are **hierarchical**. Some of the hierarchies are “natural” in the sense that everyone commonly uses them. The two main examples are time and location. Cube browsers often have pre-built tools to handle these standard hierarchies. For example, time is often examined from the top down: Year – Quarter – Month – Date. Similarly, geographical location can be written in terms of Nation – State – County – City – ZIP Code. Both dimensions could have multiple hierarchies. For instance, some companies emphasize sales by week which creates a slightly different time hierarchy: Year – Week – Date. Other dimensions can also have hierarchies, which are customized based on the specific problem. For instance, product categories might fall into hierarchies, such as by department. And a company itself is probably organized into departments and sub-departments which could be important for some questions. These hierarchies have to be manually defined but they ultimately behave the same as the natural hierarchies.



## Dimensions and Hierarchies

Cubes and MDX have many options to handle and explore hierarchical data. Figure 10.2 shows two of the main concepts. Cube browsers tend to be visual and the data is often displayed in an expansion format where analysts can drill down (expand) or roll up (compact) the data levels to look at specific items or higher-level subtotals. However, MDX treats hierarchical dimensions as trees. Trees have **levels** where increasing detail is shown at lower levels of the tree. The topmost level is the **root** and items in the bottom level are sometimes referred to as **leaves**. Functions exist to refer to a **parent** level (an item value immediately above the current level), and to list all **children** of any given node. An interesting concept is the ability to examine items in parallel. For instance, it might be necessary to compare data for each month within two different quarters. Parallel tracking could be used to compare the first month in quarters one and two (January versus April) and then move to the second month in each quarter at the same time. These advanced features are built into MDX and are difficult to handle with visual browsers.

## Rolling Thunder Bicycles Cube

It is difficult to discuss and understand MDX in abstract terms. All of the concepts are easier to follow by illustrating them with actual data. Rolling Thunder Bicycles provides a good case example. It has enough data with a reasonable number

of dimensions and measures to illustrate most points without overburdening most systems. If you want the ability to write and test MDX queries, you should install the SQL Server version of the RT Bicycles data. Also be sure that Microsoft SSAS is installed on your computer along with at least the Visual Studio client components to develop new business intelligence projects. All of these components can be installed from the Developer version of SQL Server which is available through MSDN or with a time-limited free download version. If the Developer version is not available, use the Enterprise version. The SQL Server RT database is available from the book's Web site: <http://www.JerryPost.com>.

If you do not install the SQL Server elements, you should carefully check the following figures which show the steps in creating the Cube. The cube structure is important to understand the sample queries. The queries all refer to the structure and data in the database.

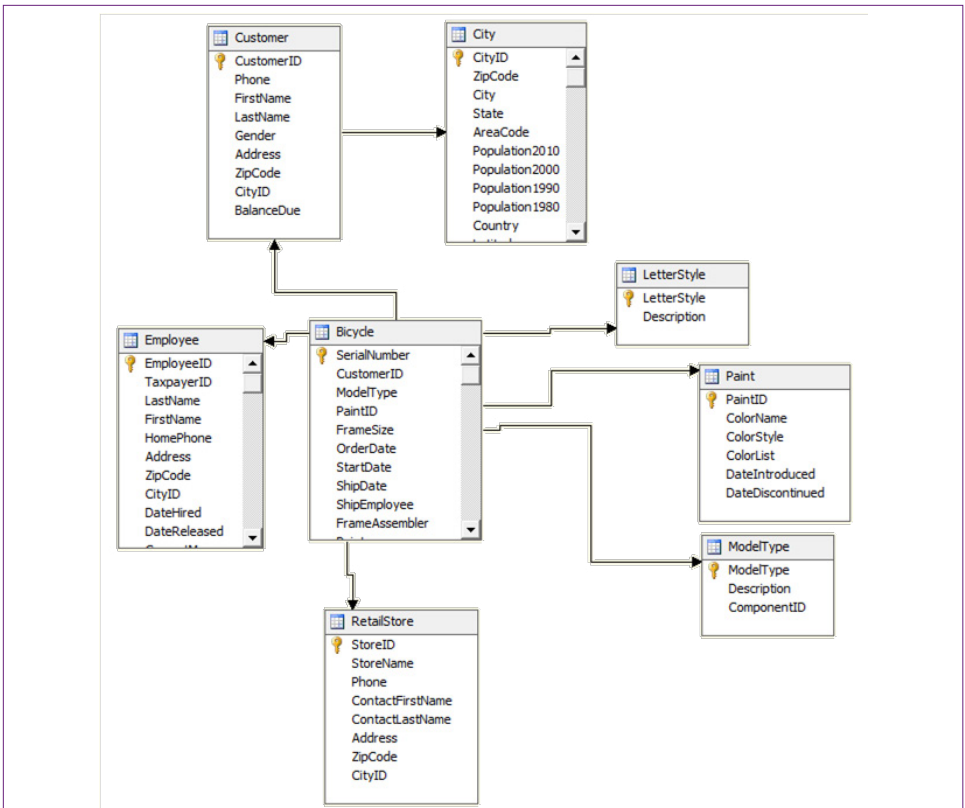
The main steps to creating a cube are: (1) In Visual Studio, start a New/Project: Analysis Services, Multidimensional..., (2) Add a new Data Source to connect to the SQL Server RT database, (3) Create a new Data Source View with the desired tables, and (4) Create a new Sales Cube with the desired measures and dimensions.

Creating the new project and the data source should be straightforward. Figure 10.3 shows the tables and relationships needed in the Data Source View for RT Sales. Clearly the Bicycle table is needed because it contains the data measure (Sale Price). The other tables provide data for the dimensions. One catch arises after adding the specified tables. Notice that three tables refer to the City table: Customer, Employee, and RetailStore. All three tables contain address information and link to the City table to get standardized information about cities. But, in terms of sales, only the link between the Customer and City table is useful. It is important to delete the links between Employee -> City and RetailStore -> City. If these links remain, the results will be severely constrained because the links would force all customers, employees, and retail stores to be in the same city. Simply select the two extraneous links and delete them.

The process of creating the Cube uses the wizard, and the main step is to select the tables correctly. First, the bicycle table is selected for its measures. The columns to use for measures need to include at least: Sale Price, List Price, Sales Tax, and Bicycle Count. A couple of other columns might be interesting, such as Frame Size if there is a reason to examine bicycle sizes across states. But by default the wizard selects all numerical columns and many of them will just clutter up the displays later so they should be unchecked. For the attribute dimensions, use all of the other selected tables—which are checked by default.

The Bicycle table includes an OrderDate column which is a useful measure for tracking when bicycles were sold. Some of the other dates might also be useful but just stick with OrderDate for now to keep the problem simpler. However, dates are a natural hierarchy, so it is important to create this date dimension. Right-click the Dimensions entry in the Solution Explorer and add a new dimension. Choose the option to “Generate a time table on the server.” This new table becomes a lookup table that contains all possible dates. You need to set the starting and ending dates to January 1, 1994 and December 31, 2015. Then pick the time periods as: Year + Quarter + Month + Date. Change the name to: Date Hierarchy to remind you that it includes multiple levels.

An interesting aspect to dimensions in SSAS is that they standalone—a dimension is really just a lookup table that contains a distinct list of all possible values



**Figure 10.3**

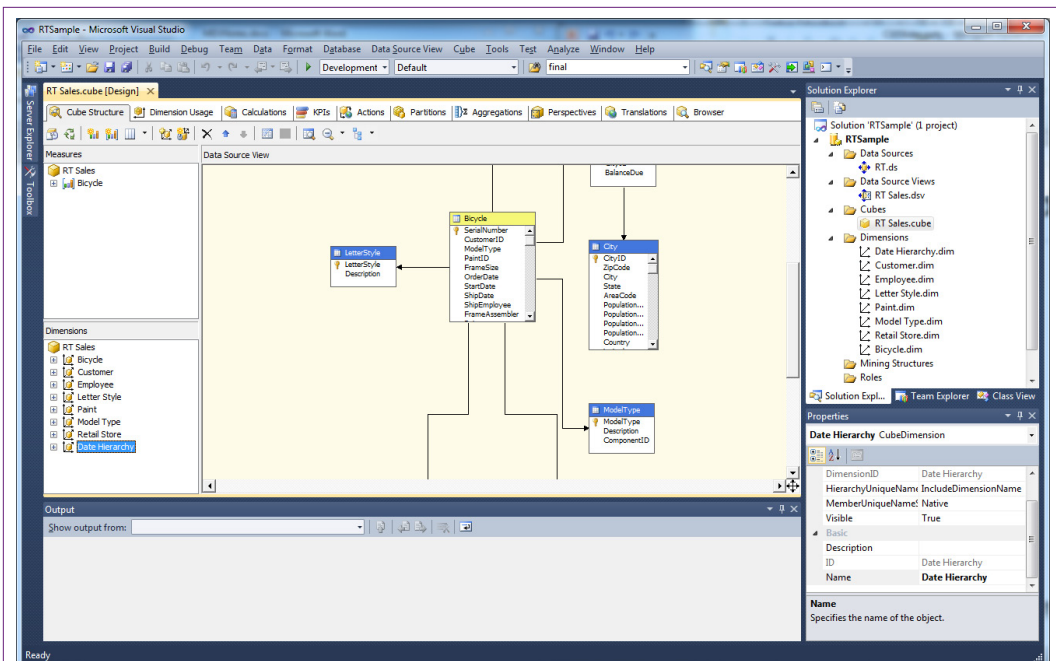
Data source view for RT Sales. The Bicycle table has the measures (Sale Price). Then add the tables with the desired attributes: Customer and City, Employee, RetailStore, LetterStyle, Paint, and ModelType. These tables are needed to provide lookup data.

for that dimension. The new Date Hierarchy dimension still needs to be added to the cube and connected to the data. Figure 10.4 shows the basic steps. Open the Cube window and examine its structure. Right-click the Dimensions window to add the new Date Hierarchy dimension.

It is also critical to click the “Dimension Usage” tab. Click the empty box next to the Date Hierarchy. As shown in Figure 10.5, the hierarchy needs to be assigned to the OrderDate column. First set the relationship type as Regular, and set the Granularity to Date if it is not set automatically. In the Measure Group Columns, pick the OrderDate column. This process assigns the Date Hierarchy specifically to the OrderDate column. Finish the wizard and return to the Cube Structure tab.

The last step is to add some elements to the Customer dimension. By default, the Customer dimension includes only ID values, but analysts will find those almost useless. As shown in Figure 10.6, edit the Customer dimension and add the more useful columns by dragging them to the attribute list. Include at least the columns: City, State; and Gender and ZIP Code from the City and Customer tables. By default, the Customer dimension includes only ID values which are Code.ludes only the ID values. Edit the dimension and dra Analysts might also find Income and population to be useful but they are not required for





**Figure 10.4**

Add Date Hierarchy dimension to the cube. Right-click the left Dimensions window and add a new dimension. Select the newly created Date Hierarchy. Then click the Dimension Usage tab.

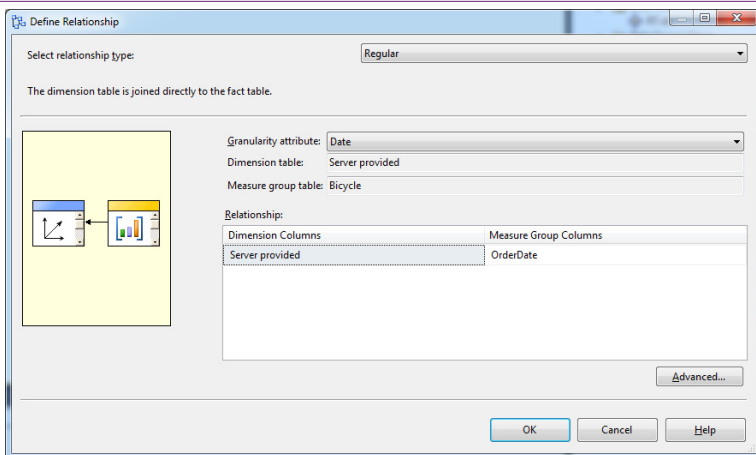
this demonstration cube. Similarly, you should eventually perform a similar process for the Employee and Retail Store dimensions to add the employee and store names. To verify your work and to populate the cube, it needs to be saved and processed.

The cube now contains measures—notably Sale Price; plus dimensions for Customer, Employee, Letter Style, Paint, Model Type, Retail Store, and the Order Date hierarchy. The name of the cube is RT Sales, but you might choose a different name. Keep this list handy, the exact names will be needed when building MDX queries.

## Definitions and Concepts

**What are the main objects in MDX queries?** MDX is a text language designed to compute and display the subtotals displayed on the cube. It uses most of the same terminology and concepts as the cube. The concepts of totals, dimensions, and hierarchy trees are critical to MDX. Probably the most important thing to remember about MDX is that it is designed to query data from an OLAP cube—so all of the results are subtotals. If no constraints or details are specified, a simple MDX query will retrieve a single total. Specifying dimensions or values generates subtotals for those dimensional values. Think of a cube as a huge collection of SQL GROUP BY subtotals. MDX simplifies the syntax so that only the GROUP BY and WHERE conditions need to be provided.

MDX uses a few key concepts or terms. Measures are numeric values that will be displayed as results—generally summed. Most cubes have a specific measures

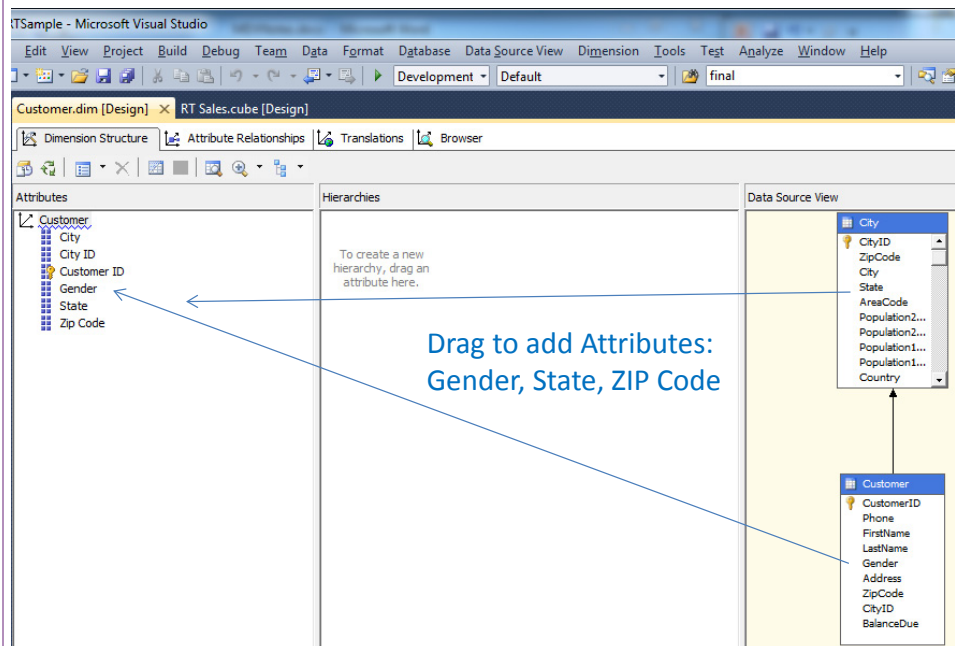


**Figure 10.5**

Assign Date Hierarchy to the OrderDate column. Set the relationship to Regular, Granularity to Date, and pick the OrderDate column in the Measure Group list.

**Figure 10.6**

Add elements to the Customer dimension. By default the Customer dimension includes only the ID values. Edit the dimension and drag useful columns into the attribute list: City, State, Gender, and ZIP Code.



set that contains a list of variables that can be used as totals. Within MDX, a specific measure variable is referenced by its full name, such as [Measures].[Sale Price]. The square brackets are used to support names that might contain spaces, key words, or special characters. Essentially, the brackets provide a way to delimit a name that could use characters that might be misinterpreted. In the example, the dot (.) indicates that the [Sale Price] attribute is a **member** of the [Measures] set.

A similar syntax is used to identify items within other dimensions. For example, several model types exist (race, road, mountain, and so on). To indicate a specific value of a model, use a term of the form: [Model Type].[Race]. For hierarchies, use the names from the top down, such as [Date Hierarchy].[Year – Quarter – Month – Date].[Date].members. Note that the last item here is members, which returns a list of all the values within the context (Date in this case). The members (or sometimes allmembers) keyword returns all item values at the specified **level** in the descriptor. It is similar to the children keyword, but technically the term children refers to the entries in a hierarchy tree; whereas members applies to any values—even if no tree is involved. For instance, to get a list of all bicycle model types, use the notation: [Model Type].members. Yes, the choice between “children” and “members” is confusing. When in doubt try “members” first and switch to “children” to see if the difference matters.

MDX notation also relies heavily on the concept of a **set**. Similar to a mathematical set, MDX uses collections of items. These collections can be a simple list of entries—such as a list of the model types; or a complex collection of dimensions—such as combining all model types with a list of states. Sets are defined inside of curly braces { }. They are most commonly used to specify the row and column dimensions for the cube. The terms **rows** and **columns** represent two common **axes** of a cube. The column axis is number 0 and the rows axis is number 1 so they can be referred to as axis(0) or axis(1); or even just 0 and 1. Sometimes the numbers are more convenient, such as when building a cube that contains more than two axes.

MDX borrows the concept of a **tuple** from SQL, which is somewhat of a strange term. In the context of MDX, think of a tuple as a specification of dimension values. For example, a tuple (Race, March 2007, CA) provides specific values for the Model Type, Month, and State dimensions. MDX uses this concept in formulas to define specific values. For instance, a formula with the term [Measures].[Sale Price] refers to the total sale price across all dimensions. To refer to the Sale Price value at a specific set of values or points, use the parentheses notation. For example, ( [Model Type].[Race], [Measures].[Sale Price] ) computes the total sale price just for the Race model type. Multiple dimensions are supported simply by separating them by commas.

## Main Syntax

---

**What is the primary structure of an MDX query?** An MDX query can become relatively complex, but as shown in Figure 10.7, the basic structure is fairly simple. A query has four basic elements: (1) calculated values defined at the top using the **WITH MEMBER** command, (2) the sets of dimensions used for each axis specified using the **SELECT** command, (3) the name of the OLAP cube holding the data using the **FROM** command, and (4) **WHERE** conditions that restrict the display to specified dimension values. These keywords might look familiar, but they have nothing to do with SQL; and the syntax and results are completely different.

WITH MEMBER	create calculated values
SELECT	define the axes by selecting dimensions
{ ... }	On Columns,
{ ... }	On Rows
FROM [cube name]	name of the cube
WHERE ( ... )	restrict results to specified dimension values

**Figure 10.7**

Basic MDX syntax. The four primary keywords are used to define calculated values, choose the sets of dimensions for each axis, specify the cube name (only one cube), and limit the display to specified attribute values.

Note that the WITH MEMBER and WHERE phrases are optional, so the initial examples will ignore those elements to focus on the underlying goals and syntax first. These two elements along with several useful functions are covered in later sections. At first glance, the main structure of MDX focuses on defining the axes for the OLAP cube and the dimensions to assign to each axis (in the SELECT section). The WHERE section simply provides a way to slice the cube and control the data. However, the optional WITH section has considerable power to perform complex calculations. The results are still computed as sums and displayed in the form of a cube, but the computations can be used to answer some relatively complex business questions. A key aspect of MDX is that all of the elements rely on sets of data and almost all of the computations involve subtotals.

## Basic Examples

**How are MDX queries written and what basic data do they provide?** The flexibility and power of MDX is easiest to see through examples. This section covers the basics—to highlight the syntax and the results. Two additional sections cover computed values and some more advanced tricks that can be useful in business analyses.

SQL Server has two main ways to run MDX queries: (1) Within a Visual Studio Analysis Project Cube Browser, and (2) Connecting SQL Server Management Studio to the Analysis database. Both of them use similar steps and produce similar results. The Management Studio supports cut-and-paste on some of the results, but the Visual Studio Cube Browser supports a Designer mode where basic cubes can be built by dragging and dropping dimensions. The examples in this chapter can be run with either method; but some examples found on the Web will run only in the Management Studio.

Creating and running MDX queries is a little tricky in the recent versions of Visual Studio. If necessary, open the project used to define the cube. Open the RT Sales cube. If necessary, process the cube so all dimensions, hierarchies, and subtotals are built. Click the cube's Browser tab. By default, the browser is in designer mode—where you can drag and drop dimensions and measures to create a cube. This process automatically generates MDX queries—which can provide useful examples. But for now, turn off the designer by deselecting the Design Mode icon, which switches the display to a window to type MDX queries and a window to display results.

```
SELECT
  { [Model Type].members } on rows,
  { [Measures].[Sale Price] } on columns
FROM [RT Sales]
```

Model Type	Sale Price
(null)	208438543
Hybrid	3399366.21
Mountain	25793699.63
Mountain full	61436230
Race	61700574.07
Road	46617603.73
Tour	9268120.59
Track	222948.77
Unknown	(null)

**Figure 10.8**

Initial MDX query. Total sales (Sale Price) by Model Type. The query needs to specify only the rows dimension (Model Type), and the column measure (Sale Price). MDX automatically totals across all other dimensions. The (null) row in the result shows the grand total. The Unknown row is automatically included in case some values are not specified. Changing [Model Type].members to [Model Type].children removes the (null) total.

To create MDX queries in SQL Server Management Studio, start the Management Studio from Windows and change the Server type to: Analysis Services. Choose the correct server name and login information. In the Object Explorer, expand the Databases entry to find the project you created and saved within Visual Studio. Right-click that entry (RT Sample) and choose the option for New Query/MDX.

### A First Example

Begin with a query that uses the simplest syntax possible. The business question is to display total sales by model type. Figure 10.8 shows the query and the results. The FROM command is the simplest because it just lists the name of the data cube. Only one cube can be used in any query. The SELECT element specifies two sets—one for each axis of rows and columns. The braces indicating a set are required—even if only a single dimension is needed. The first set { [Model Type].members } is assigned to the rows. The .members notation tells the system to look up all entries for model type. The second set { [Measures].[Sale Price] } specifies the values to be displayed. Every query must have at least one measure in the SELECT statement. Note that the order of the sets in the SELECT statement does not matter (columns can be defined before rows). Check the results to see that all of the model types have been retrieved. Notice the (null) and Unknown entries. The Unknown value is automatically included to handle cases where the model type might not be given. The (null) row shows the grand total. Out of curiosity, change the .members to .children and rerun the results. The values will be the same, but the .children approach does not include the grand total row.

```

SELECT
  { [Model Type].members } on rows,
  { [Measures].[Sale Price] } on columns
FROM [RT Sales]
WHERE ( [Year].[Calendar 2010] )

```

Model Type	Sale Price
(null)	14501690
Hybrid	(null)
Mountain	1018340
Mountain full	4277740
Race	5171940
Road	3417090
Tour	616580
Track	(null)
Unknown	(null)

**Figure 10.9**

Adding a WHERE condition. Conditions added to WHERE limit the data to the specified values. The new table results are structurally the same as before but the values are computed only for 2010. Note the need to specify [Calendar 2010] because that is the way the values are entered in the Date Hierarchy dimension.

## Adding a WHERE Condition

What if the analyst wants to limit the results? For example, the analyst wants to see the sales of model type but only for 2010. Note that the date is not currently included in the display—the existing values are computed for all possible years. Restricting the results to a single year can be thought of as looking at a **slice** of the cube. As shown in Figure 10.9, this basic condition is straightforward to add to the WHERE clause. The one catch is that the condition has to be written as [Calendar 2010] because that is the way the data was generated for the Date Hierarchy dimension. In many cases it helps to build a sample query using the designer first—just to see the exact specifications of the data for these generated dimensions.

Note that a WHERE condition specifies values that will limit the subtotal computations. To specify additional conditions, separate them by commas. The totals will then be computed where all of the specified conditions are true. For example, keep the year 2010 condition and add another restriction to the state of California (CA). Figure 10.10 shows the syntax and the results. Note that the values in the WHERE clause are separated by commas and data must match both conditions to be included in the result totals. More complex conditions require the use of functions; which are described in a later section.

Notice that Track and Hybrid bikes were only sold in some years—and not in 2010—so the results include missing (null) values. Because this effect occurs on only two rows it might not matter. But if a data result contains many empty rows, the analyst might want to remove them to focus on the values that do exist. MDX uses the **NON EMPTY** keyword to remove rows that are completely empty. Figure 10.11 shows the keyword and the result.

```

SELECT
  { [Model Type].members } on rows,
  { [Measures].[Sale Price] } on columns
FROM [RT Sales]
WHERE ( [Year].[Calendar 2010], [State].[CA] )

```

Model Type	Sale Price
(null)	895730
Hybrid	(null)
Mountain	73530
Mountain full	226640
Race	290470
Road	262160
Tour	42930
Track	(null)
Unknown	(null)

**Figure 10.10**

Multiple WHERE conditions. The WHERE clause specifies values that will be included. Separate them by commas and only data matching all conditions will be used for the totals. Here, the state of California for the year 2010.

**Figure 10.11**

Removing empty rows. The NON EMPTY keyword drops the rows from the results that are completely empty (null).

```

SELECT NON EMPTY
  { [Model Type].members } on rows,
  { [Measures].[Sale Price] } on columns
FROM [RT Sales]
WHERE ( [Year].[Calendar 2010], [State].[CA] )

```

Model Type	Sale Price
(null)	895730
Mountain	73530
Mountain full	226640
Race	290470
Road	262160
Tour	42930



```

SELECT NON EMPTY
  { [Model Type].[Mountain], [Model Type].[Mountain full] } on rows,
  { [Measures].[Sale Price] } on columns
FROM [RT Sales]
WHERE ( [Year].[Calendar 2010], [State].[CA] )

```

Model Type	Sale Price
Mountain	73530
Mountain full	226640

**Figure 10.12**

Selecting only some rows. Selection values are sets can contain lists and combinations of dimensions. Remove the .members value that retrieved all model types and specify just the two values for mountain and mountain full model types.

## Displaying Specific Dimension Values

What if the analyst wants to focus on just the mountain and mountain full bikes? Because the Model Type dimension is used in the SELECT clause, it should not (or cannot) be used as a WHERE condition. The answer is to not select all members but just specify the desired entries on the SELECT statement. Remember that selection of dimensions for the axes use a set so many different values can be entered. Figure 10.12 shows the change in the MDX query that removes the .members keyword and replaces it with the two model types (mountain and mountain full). The computed values are the same as the earlier query but now display just the values for the two selected model types.

It is possible to add other types of dimensions to the same set. However, it might be more useful to put different dimensions on different axes. On the other hand, the browser with SSAS 2012 really only supports rows and columns. It has no interface to show different combinations of dimensions. The Microsoft PivotTable in Excel does a better job of handling multiple dimensions at the same time. So, a cube could be constructed and saved in SSAS and then browsed with a PivotTable. Some of these issues arise in the next section on cross joins.

## Cross Join

The whole point of an OLAP cube is to provide the ability to explore data and browse through different dimensional subtotals and different hierarchical levels. MDX has some useful tools to create these types of cubes. Unfortunately, the cube browser shipped with SSAS 2012 is weak and does not support interactive roll-up and drill down. Hopefully, the browser will be improved in future editions. In the meantime, it is still important to understand how MDX handles multiple dimensions.

The cross join is an important tool in building a cube. A **cross join** takes all the values of one dimension and combines them with every value from a second dimension. For example, say the state dimension has 50 states and the model type dimension has 7 entries. Crossing every state with every model type leads to  $7 * 50 = 350$  entries. Think of the results in terms of a matrix—using 7 columns against 50 rows. In fact, this is the way most cube browsers would display this cross join.

```

SELECT NON EMPTY
  { [Measures].[Sale Price] } ON COLUMNS,
NON EMPTY
  { ([Date Hierarchy].[Year - Quarter - Month - Date].[Date].ALLMEMBERS
    * [Model Type].[Model Type].[Model Type].ALLMEMBERS ) }
ON ROWS
FROM [RT Sales]

```

*Note that the hyphens in the [Year – Quarter ...] term are picky. You might have to copy and paste the notation from the designer.*

Year	Quarter	Month	Date	Model Type	Sale Price
Calendar 1994	Quarter 1	January	01-JAN-1994	Race	2990
Calendar 1994	Quarter 1	January	01-JAN-1994	Road	3490
Calendar 1994	Quarter 1	January	02-JAN-1994	Race	2478.95
Calendar 1994	Quarter 1	January	03-JAN-1994	Mountain	5020
Calendar 1994	Quarter 1	January	03-JAN-1994	Race	10410
...					

**Figure 10.13**

Selecting only some rows. Selection values are sets can contain lists and combinations of dimensions. Remove the .members value that retrieved all model types and specify just the two values for mountain and mountain full model types.

To demonstrate the cross join in MDX, start with a new problem. The goal is to cross the Date Hierarchy with the Model Type dimension to examine Sale Price over time and model. Figure 10.13 shows the MDX command and a small portion of the results. The key to creating the cross join is the asterisk (\*). MDX syntax includes a *cross join* key word but the asterisk is actually easier to read. The basic syntax is `dimension1.members * dimension2.members`. The SSAS 2012 browser display is not interactive so most people would transfer the cube to an Excel PivotTable for browsing instead. The Visual Studio editor does have a button to transfer the data to Excel but the cube has to be reconfigured. Of course, filters can still be used to reduce the number of rows—such as selecting individual states or the gender of customers.

Notice that the SSAS 2012 browser does not display the data as a cube. Instead, it generates a new row for each entry and then lists the corresponding dimension value. This approach makes it harder to see patterns, but easier to display multiple measure values. Also, the data can be passed cleanly to other analysis tools such as statistical packages.

## Calculated Measures

**How are computations and new measures defined?** Many business problems require manipulating the measure values to compute new variables. Simple computations include profit margin (revenue – costs) and simple percentages such as Costs / Revenue. These computations are straightforward because they operate on data within the same cell.

```

WITH MEMBER [Measures].[Margin]
  AS '[Measures].[Sale Price] - [Measures].[Component List] - [Measures].[Frame Price]'
SELECT
  NON EMPTY { [Measures].[Margin], [Measures].[Sale Price] } ON columns,
  NON EMPTY { [Year].members * [Model Type].members } ON rows
FROM [RT Sales]

```

Year	Model Type	Margin	Sale Price
(null)	(null)	47211482.32	208438543
(null)	Hybrid	884842.47	3399366.21
...			
Calendar 1994	(null)	261651.05	2555013
Calendar 1994	Hybrid	14310.22	199006.21
...			

**Figure 10.14**

Calculated Measures. Computations are defined in the WITH MEMBER section which contains the name of a new measure (margin) and its definition delimited in single quotes. The new variable can then be used in the SELECT statement.

In MDX, calculated measures are defined in a separate section at the top of the query. A new variable name is assigned and the formula is delimited with single quotes. Figure 10.14 shows the formula for computing the simple profit margin for Rolling Thunder Bicycles (ignoring labor costs). The measure is defined in the WITH MEMBER section added to the top of the query. It must be given a unique name (Measures.Margin) and the calculation is delimited in single quotes. The new variable can then be used within a SELECT statement.

The syntax for defining a new measure is straightforward—just remember to place single quotes around the expression that defines the computation. Also, be sure that the name of the new variable is unique. Similar computations involving data from the same source at the same level are equally straightforward to create. However, MDX has several built-in functions that make it possible to answer more complex questions.

## Complex Computations

**How does MDX handle complex computations that cross levels or rows of data?** The computations in the previous section were deliberately kept simple to illustrate the syntax for calculating new values. MDX has several powerful functions that can be used to answer more complex questions. Most of these types of questions involve the use of data beyond that found in a single row. For example, calculating percentages requires dividing the current value by a subtotal which must be computed across all the members within the same level. Similarly, many time-based problems involve looking at consecutive values to compute changes over time. Special functions are often needed to handle these jumps across different levels, but MDX has several of these capabilities.

```

WITH
  MEMBER Measures.PctSales AS '([Model Type].CurrentMember, [Sale Price] ) / ([Model
  Type].CurrentMember.Parent, [Sale Price])' , FORMAT_STRING="0.00%"
SELECT
  { [Measures].[Sale Price], [Measures].[PctSales] } ON COLUMNS,
  { [Model Type].Members } ON ROWS
FROM [RT Sales]

```

Model Type	Sale Price	PctSales
(null)	208438543	Infinity
Hybrid	3399366.21	0.01631
Mountain	25793699.63	0.12375
Mountain full	61436230	0.29475
Race	61700574.07	0.29601
Road	46617603.73	0.22365
Tour	9268120.59	0.04446
Track	222948.77	0.00107

**Figure 10.15**

Computing percentages using Parent function. The computation relies on the fact that MDX always computes totals. The key term is: ( [Model Type].CurrentMember.Parent, [Sale Price] ) which computes the overall total to use as the divisor. Note that the Format\_String does not work in Visual Studio 2012.

## Percentages

A common business situation is the need to display subtotals along with their percentage of the overall total. Consider the sales for Rolling Thunder by model type. A listing of the model types followed by the total sales value is interesting. But, the numbers can be easier to understand if they are computed as percentages. For instance, rather than just listing: Race 2000, Mountain 4000, and so on; the display could list: Race 15%, Mountain 35%, and so on. These percentages would be even more useful to examine changes over time. The computation of the percentages in MDX is handled by defining a new measure and using the parent property.

Figure 10.15 shows the MDX command and the results with the percentages. The key trick is in the formula for the divisor. Look closely at the formula: ( [Model Type].CurrentMember.Parent, [Sale Price] ). This formula uses the Parent function to move up one level in the hierarchy; and one level up means to look at all model types. MDX then sums the [Sale Price] at that parent level—which provides the total sales for all model types.

What would happen if you combine the cross-join with percentages? That is, the business analyst wants to examine the total sales by model type by year; and wants to see the percentages computed within each year. The answer is that you simply need to add the cross join to the MDX row axis. Everything else stays the same. Figure 10.16 shows the MDX query and a portion of the results. Again, the results would be easier to browse if the model type were displayed across the top as columns as in an Excel PivotTable. But the main point is that MDX computed the percentages using the correct totals. The totals are correct because the Parent

```

WITH
  MEMBER Measures.PctSales AS '([Model Type].CurrentMember, [Sale Price] ) /
    ([Model Type].CurrentMember.Parent, [Sale Price])'
SELECT NON EMPTY
  { [Measures].[Sale Price], [Measures].[PctSales] } ON COLUMNS,
NON EMPTY
  { ([Year].AllMembers
    * [Model Type].[Model Type].ALLMEMBERS ) } ON ROWS
FROM [RT Sales]

```

Model Type	Sale Price	PctSales
Hybrid	199006.21	0.07789
Mountain	559019.63	0.21879
Race	928984.07	0.36359
Road	486773.73	0.19052
Tour	346200.59	0.13550
Track	35028.77	0.01371
Hybrid	124240	0.04200
Mountain	567940	0.19199
Race	294390	0.99516
Road	1074490	0.36322
Tour	709230	0.23975
Track	187920	0.06352

**Figure 10.16**

Cross-join with percentages. Compute sales by model type by year and the corresponding percentages within each year. The only change is to add the cross-join. The parent function automatically computes the total across model types within the same year.

function simply moves up one level from the current item (model type within a given year) and then computes the total at that level. So, regardless of how the cross join is defined, the totals and percentages will be computed across the model types for the level.

Yes, it is possible to navigate further up the data tree using multiple parent commands (.parent.parent). Such a command might be used to compute a grand-grand total. But, be careful to ensure that the data tree has enough levels or trying to move to a non-existent parent will trigger an error.

### Compute Changes

Another common analytical issue is the need to examine changes in data—particularly over time. As shown in the earlier examples, computing totals at any point in time is straightforward in MDX. Picture a table listing the Year and Sale Price for each year. The challenge now is to take the value for one row and subtract the value from the previous row. That means MDX needs a function to retrieve the value on the previous row. The function **PrevMember** does exactly that task. There is also a **NextMember** function that looks forward instead of back. For

```

WITH
  MEMBER [Measures].[Change] AS '[Measures].[Sale Price]
    - ([Measures].[Sale Price], [Year].PrevMember)'
SELECT
  { [Measures].[Sale Price], [Measures].[Change] } ON COLUMNS,
  { [Date Hierarchy].[Year].Children } ON ROWS
FROM [RT Sales]

```

Year	Sale Price	Change
Calendar 1994	2555013	2555013
Calendar 1995	2958210	403197
Calendar 1996	4050860	1092650
Calendar 1997	5358120	1307260
...		

**Figure 10.17**

Computing change values with PrevMember. Change values require looking at values in prior (or next) rows. The PrevMember function looks back one item. NextMember looks forward by one, and Lag and Lead functions can jump multiple rows at one time.

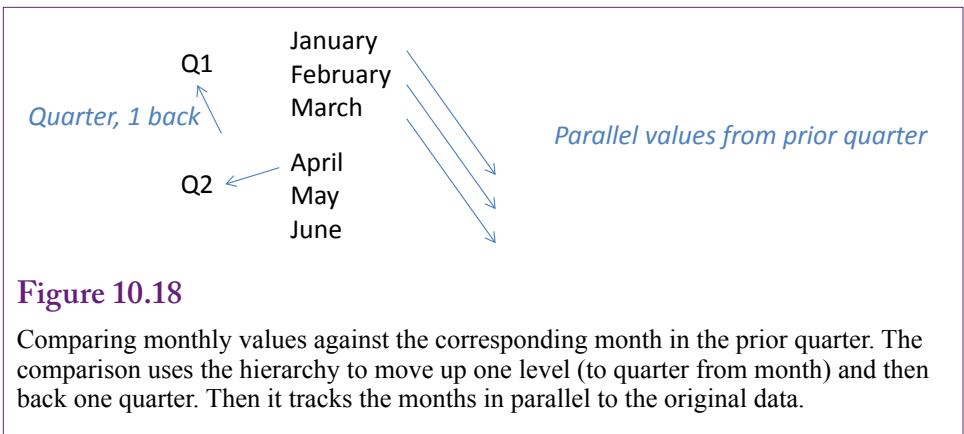
more extreme situations, the **Lag** and **Lead** functions accept a number to go back or forward more than one item at a time. These functions overlap somewhat. For example, data on the previous row could be referenced by any of the commands: PrevMember, Lag(1), or Lead(-1).

Figure 10.17 shows the Change variable defined with the PrevMember function. The basic syntax is to use parentheses to specify the variable to look up (Sale Price) and then the PrevMember function of the year variable: ([Measures].[Sale Price], [Year].PrevMember). Check the change values in the partial results table. Notice that the first row does have an entry—it is equal to the first value of sales (for 1994). Specifying this value is always a question when computing changes. MDX assumes that the system starts at zero, so any data in the first year must be the full change amount from zero. Other systems might leave the first row empty, so it is worth remembering this approach in MDX.

## ParallelPeriod Function

Many business questions require comparing numbers at different points in time. Comparing data to the prior period is common, but more complex comparisons are also common. For example, consider a listing of monthly sales using the time hierarchy. The hierarchy shows Year, Quarter, and Month. The challenge is that the business managers want to compare sales in a given month to the same month in the prior quarter. For example, sales in April (second quarter, first month) should be compared to sales in January (first quarter, first month). The **ParallelPeriod** function was designed to handle these types of question.

Figure 10.18 shows the basic objective. Looking at the month of April, the matching value is found by moving up the tree (parent) to the quarter; then moving back one quarter. From that point, the three months are matched in parallel to the months in the original quarter.



**Figure 10.18**

Comparing monthly values against the corresponding month in the prior quarter. The comparison uses the hierarchy to move up one level (to quarter from month) and then back one quarter. Then it tracks the months in parallel to the original data.

Figure 10.19 shows the MDX command to compare the corresponding months from a quarter to those in the prior quarter. The heart of the query is function: `ParallelPeriod([Date Hierarchy].[Year - Quarter - Month - Date].[Quarter], 1)`. The first parameter tells the function to use the quarters (parent of the month level). The second parameter specifies to go back a single quarter. A third parameter can be added, but it defaults to the current member (month) being displayed and is easier to leave blank. Actually, for this specific query, all three parameters could be blank because the default values match the desired elements. Looking at the

**Figure 10.19**

`ParallelPeriod` function. The totals are monthly so the function uses the parent level (Quarter) to move back one period. It then automatically parallels or matches the corresponding months.

```
WITH
MEMBER [Measures].[ParQtr] AS
  ( [Measures].[Sale Price],
    ParallelPeriod ( [Date Hierarchy].[Year - Quarter - Month - Date].[Quarter], 1 ) )
SELECT NON EMPTY
  { [Measures].[Sale Price], [Measures].[ParQtr] } ON COLUMNS,
NON EMPTY
  { ([Date Hierarchy].[Year - Quarter - Month - Date].[Month].ALLMEMBERS ) } ON
ROWS
FROM [RT Sales]
```

Year	Quarter	Month	Sale Price	ParQtr
Calendar 1994	Quarter 1	January	215836.97	(null)
Calendar 1994	Quarter 1	February	205952.48	(null)
Calendar 1994	Quarter 1	March	211128.77	(null)
Calendar 1994	Quarter 2	April	188159.83	215836.97
Calendar 1994	Quarter 2	May	225387.63	205952.48
Calendar 1994	Quarter 2	June	210278.66	211128.77
...				



partial results, it is clear that the function has picked up the matching months from the prior quarter—except for the first quarter of null values because nothing exists before that time.

In a similar manner, companies often want to compare monthly sales to the value in the prior year. The `ParallelPeriod` function in this example could be modified to handle a full year simply by changing the second parameter from 1 to 4, because 4 quarters make up a year. This function simplifies many common business comparisons over time. Note that the function itself returns the time value not the actual data. Hence the new member measure is defined using the tuple notation with parentheses: (data item, member/level) where the member value is obtained using the `ParallelPeriod` function.

## Some MDX Functions

---

**What other MDX functions are commonly used in business problems?** MDX has many functions and they can be used to solve tricky problems. For example, check the Web for examples of problems such as computing year-to-date totals for current and prior years. The purpose of this section is to briefly describe some of the commonly-used functions in MDX. Only a partial list is covered here—the online reference documents for MDX provide complete lists and more examples. Figure 10.20 lists some of the main MDX functions by category. Detailed lists and descriptions of the functions can be found online in the reference documents. Some of the functions are straightforward; others are complex and require detailed explanations and examples. Some of the functions have already been mentioned (`Lag`, `Lead`, `ParallelPeriod`, `Parent`, `PrevMember`). A couple of additional functions are described in this section, but a comprehensive discussion would require an entire textbook just to cover the functions.

### EXCEPT: Taking Values Out of Totals

Recall the `WHERE` conditions given in the initial examples. Expressions entered in the `WHERE` clause act as a cube slicer or filter and limit the totals to just that data. The example computed total sales by Model Type for the state of California (CA). The syntax was straight forward:

```
SELECT NON EMPTY
  { [Model Type].members } on rows,
  { [Measures].[Sale Price] } on columns
FROM [RT Sales]
WHERE ([State].[CA])
```

But, what if the business manager wanted the opposite information: Sales in all states except California? So far, all of the functions and tools of MDX have focused on selecting data to be included and displayed. The solution is to use the **EXCEPT** function, which returns all of the values from a set minus the ones specified. The `EXCEPT` function takes two parameters: (1) The full set of items, and (2) The item to be excluded.

Figure 10.21 shows the MDX query to display the sales by model type for all states except the state of California. The results show the query executed three times with different `WHERE` conditions. The data can be tested in Excel to verify that the `EXCEPT` column contains the total for all states except California. The `EXCEPT` function can also be used within the `SELECT` clause to choose which model types should be displayed. In most cases, it will be easier to read the query

Meta	Axis Count Hierarchy	Level Name Ordinal	
Navigation	Ancestor Ascendants Children Cousin Current CurrentMember	DataMember FirstChild FirstSibling Lag LastChild LastSibling	Lead LinkMember NextMember Parent PrevMember Siblings
Logical	IIF IsAncestor IsEmpty	IsGeneration IsLeaf IsSibling	
Sets	AllMembers BottomCount BottomPercent BottomSum Crossjoin Descendants Distinct Except	Exists Extract Filter Generate Head Hierarchize Intersect Members	NonEmpty Order Subset Tail TopCount TopPercent TopSum Union
Statistical	Avg Correlation Count Covariance DistinctCount	Max Median Min Rank RollupChildren	Stdev Sum Var VisualTotals LinReg...
Time	ClosingPeriod LastPeriods OpeningPeriod ParallelPeriod	Mtd Qtd Wtd Ytd	PeriodsToDate

**Figure 10.20**

Some MDX functions. Some are more useful than others and the full list and details can be found in the online MDX reference documents.

if the desired items are entered. However, for long lists, the EXCEPT function could make it easier to create a query. The syntax remains the same; simply enter the EXCEPT function inside a SELECT brace and enter the full list and the exception item.

### Conditions with IIF and CoalesceEmpty

Much like SQL, MDX is designed to operate on sets of data and it is not designed to function as a sequential programming language. Conditions are a key aspect of sequential languages (IF ... THEN ... ELSE), but are less important in queries. Instead, the SELECT and WHERE statements control which items are included or excluded. Nonetheless, sometimes queries require different treatment for certain situations. A common example is to handle problems arising from division by zero. A calculation might be performed differently, or skipped, if a divisor is zero. MDX, as with many other tools such as Excel, handle conditions with the **IIF** function. IIF stands for “immediate if,” and consists of three parameters:

```
IIF ( condition, true, false)
```

```

SELECT NON EMPTY
  { [Model Type].members } on rows,
  { [Measures].[Sale Price] } on columns
FROM [RT Sales]
WHERE ( EXCEPT ( [State].Children, [State].[CA]) )

```

Model Type	All	CA	EXCEPT CA
All	208438543	16274614.3	192163928.7
Hybrid	3399366.21	219987.38	3179378.83
Mountain	25793699.63	1742601.88	24051097.75
Mountain full	61436230	4362440	57073790
Race	61700574.07	5879280.04	55821294.03
Road	46617603.73	3380330	43237273.73
Tour	9268120.59	654335	8613785.59
Track	222948.77	35640	187308.77

**Figure 10.21**

Except function. The query was run with three different WHERE conditions to verify the results. The EXCEPT function requires two parameters: (1) The full list, (2) The item to be excluded from the list.

Most of the data in the Rolling Thunder Bicycles case is relatively clean, with few chances of dividing by zero. However, several examples do generate missing or null data which can also be used to illustrate the IIF function. Figure 10.22 shows an example of the IIF function. Full suspension mountain bikes were not introduced by the company until 1997, so model type computations before that year contain missing (null) values. The IIF function is used here to check for missing (IsEmpty) values before dividing by sale price. If the value is missing, the function returns a zero, otherwise it performs the division. Technically, the IIF function is not needed for null data because any computation with null value always results in a null value so the system does not bother to attempt the actual computation. But the format and use of the function is the same if testing for zero values—simply replace the IsEmpty function with a test to see if the item equals zero ( $a = 0$ ).

The IIF function can also be used to recode data—but it only codes two items at a time. For instance, the condition could test to see if total sales are above some limit and then return a positive indicator, otherwise, it returns a zero or some other value.

Dealing with null values is common enough that MDX has a separate function just to handle this situation: CoalesceEmpty. Essentially it is a simplified test to see if a value is empty. Remember that any computation with null values creates a null result. Consider a calculation that adds two numbers: [Measures].[Frame Price] + [Measures].[Component List]. What happens if a customer purchases a frame but no components? Then [Component List] will be empty and the total will also be defined as null, and essentially discarded from the analysis. To treat

```

WITH
  MEMBER [Measures].[ComponentPct] AS
    'IIF(IsEmpty([Measures].[Sale Price]),0, [Measures].[Component List]/[Measures].[Sale
  Price])'
SELECT
  { [Year].members * [Model Type].members } on rows,
  { [Measures].[Sale Price], [Measures].[ComponentPct] } on columns
FROM [RT Sales]

```

Year	Model Type	Sale Price	ComponentPct
Calendar 1994	All	2555013	0.68167
Calendar 1994	Hybrid	199006.21	0.68211
Calendar 1994	Mountain	559019.63	0.64696
Calendar 1994	Mountain full	(null)	0
Calendar 1994	Race	928984.07	0.69753
Calendar 1994	Road	486773.73	0.70422
Calendar 1994	Tour	346200.59	0.64459
Calendar 1994	Track	35028.77	0.86555
Calendar 1994	Unknown	(null)	0

**Figure 10.22**

IIF function. Commonly used to handle division by zero or missing data, the IIF function takes three parameters: (1) logical condition, (2) value if condition is true, and (3) value if condition is false.

the two items as optional, they can be coalesced to a zero value before trying to add them:

```

CoalesceEmpty([Measures].[Frame Price], 0)
+ CoalesceEmpty([Measures].[Component List], 0)

```

Now if either value is missing, it will first be converted to zero before attempting to add the values together. As a result, the calculation will not generate null values even if one of the two items is missing.

### TopCount Function

With SQL, Microsoft SQL Server introduced the TOP n option which is often used to cut off a display list to show just the top number of items. This same concept shows up in the MDX **TopCount** (and **BottomCount**) function. The purpose of the TopCount function is to compute the desired totals, sort the totals, and return just the top n rows. The function requires three parameters:

```

TopCount ( row list members, n, measure to total)

```

The Bottom count function works the same way but counts up from the bottom of the list.

Figure 10.23 shows a sample use of the TopCount function. It computes the total sales by state and returns to five highest values.

Initially, the **TopSum** and **TopPercent** functions appear to be similar to the TopCount function; however, the behavior is different. The TopCount function

```

SELECT
  { [Measures].[Sale Price] } ON Columns,
  TOPCOUNT ( [State].Children, 5, [Measures].[Sale Price]) ON ROWS
FROM [RT Sales]

```

State	Sale Price
CA	16274614.3
NY	12437726.13
TX	11914093.18
IL	11408168.16
PA	11294417.46

**Figure 10.23**

TopCount function. The function sorts the data and returns the top rows from the list. The function needs to know the dimension for the rows, the number of rows to return, and the measure to be summed.

computes the totals and then cuts off the display list by counting the number of rows. The TopSum function has a similar syntax:

```

TopSum( [State].Children, 50000000, [Measures].[Sale
Price] )

```

But the control number (50,000,000) represents a total sum. The query starts at the top of the list and returns rows until the sum of the values exceeds the specified control value. So the sample query will list the top states needed to accumulate a total sales level of at least 50 million. The TopPercent function behaves similarly to the TopSum function. It lists states from the top down until the sales total from those states exceeds the specified percentage of the aggregate sum. So the business query would be of the form: List the states that generated 20 percent of the total sales. The BottomX functions operate the same but work from the bottom up instead of top down.

The TopCount function also introduces some issues in how sets are organized. What happens when a new list is created from Year cross-joined to the States? Should the top 5 count apply within each year or across the states? Figure 10.24 shows two ways to write the function: (1) Year cross-joined to TopCount, and (2) TopCount of Year cross-joined to State. Of course, another possibility would be to cross join State to TopCount by Year. The point of the examples is to highlight the importance of understanding when to apply a function versus when to perform a cross-join. The choice depends on the specific business question to be answered; but it is critical to match the method to the question—and then test the query by carefully examining the results.

## Year to Date

Business managers are often interested in the progress of sales throughout the year. In particular, they want to see year-to-date totals, so they can get a feel for the total sales for the year. Year-to-date totals are often displayed next to monthly (or quarterly) values so the display essentially provides the monthly values and a

```

SELECT
  { [Measures].[Sale Price] } ON Columns,
  { [Year].Children *
    TopCount ( [State].Children, 5, [Measures].[Sale Price] ) } ON ROWS
FROM [RT Sales]

```

Year	State	Sale Price
1994	CA	257734.3
1994	NY	188326.13
1994	TX	151563.18
1994	IL	158728.16
1994	PA	163677.46
1995	CA	286920
1995	NY	186600

```

SELECT
  { [Measures].[Sale Price] } ON Columns,
  { TopCount( [Year].Children * [State].Children, 5, [Measures].[Sale Price] ) } ON ROWS
FROM [RT Sales]

```

Year	State	Sale Price
2014	CA	1160690
2012	CA	1157050
2007	CA	1150560
2014	TX	1142330

**Figure 10.24**

TopCount function in a cross-join. The first version cross joins year to the top count, which results in the top 5 states for each year. The second version performs the cross join first and finds the top 5 overall totals.

running total. Because of the popularity of these totals, MDX has a specific **YTD** function.

Technically, the YTD function returns a set of time periods within a hierarchy from the start of the year to the current time. So, time has to be in a hierarchy that specifies a calendar year and the MDX query has to use the Sum or Aggregate function to total the desired measure. Figure 10.25 shows the simplest version of the YTD function. The function automatically evaluates the time dimension hierarchy and finds the start of the year for the current time variable—month in this case. Hence, the function can be called with no parameters. The Aggregate (or Sum) function is used to add up the values for the Sale Price measure over the time periods returned from the YTD function.

Of course, business managers are never happy with the data provided to them. So, the next question asked is going to be: Can MDX compare year-to-date values for the current year with the values from the previous year? The solution is to

```

WITH
  MEMBER [Measures].[MonthYTD] AS
    'Aggregate ( YTD(), [Measures].[Sale Price] ) '
SELECT
  { [Measures].[Sale Price], [Measures].[MonthYTD] } on 0,
  { [Month].Children } on 1
FROM [RT Sales]

```

Month	Sale Price	MonthYTD
Jan-94	215836.97	215836.97
Feb-94	205952.48	421789.45
Mar-94	211128.77	632918.22
Apr-94	188159.83	821078.05
May-94	225387.63	1046465.68
Jun-94	210278.66	1256744.34
Jul-94	244788.67	1501533.01
Aug-94	197913.77	1699446.78
Sep-94	200731.22	1900178
Oct-94	219020.42	2119198.42
Nov-94	216574.56	2335772.98
Dec-94	219240.02	2555013
Jan-95	240050	240050

**Figure 10.25**

YTD function. The YTD function knows about time hierarchies and calendar years automatically so it can be called with no parameters to use the current time variable (month).

combine the YTD function with the ParallelPeriod function. Define a new measure (PriorYTD) as:

```

Aggregate (
  YTD ( ParallelPeriod ( [Date Hierarchy].[Year -
    Quarter - Month - Date].[Year], 1 )
    , [Measures].[Sale Price] )

```

## Moving Averages

One more common business problem involves moving averages—which is a statistical concept that applies to time series data. Moving averages are commonly used in finance—they are useful for smoothing out short-term variations in prices. But they can be used on any time series data, including sales. A moving average is exactly what it says: an average that moves over time. The average moves because it is defined for a set number of periods. For instance, an MA(3) refers to an average of three data points and MA(12) averages data across 12 consecutive periods. Consider sales by month for Rolling Thunder Bicycles. An MA(3) would begin by averaging the values for the first three months: 1994-01, 1994-02, and 1994-03. This value would be entered for 1994-03. For the next month (1994-04), a new average of three values would be computed from 1994-02, 1994-03, and 1994-04.



```

WITH
  MEMBER [Measures].[MA03] AS
    'Avg([Month].CurrentMember.Lag(2):[Month], [Measures].[Sale Price] )'
  MEMBER [Measures].[MA12] AS
    'Avg([Month].CurrentMember.Lag(11):[Month], [Measures].[Sale Price] )'
SELECT
  { [Measures].[Sale Price], [Measures].[MA03], [Measures].[MA12] } ON 0,
  { [Month].Children } ON 1
FROM [RT Sales]

```

Month	Sale Price	MA03	MA12
1994-01	215837	215837	215837
1994-02	205952.5	210894.7	210894.7
1994-03	211128.8	210972.7	210972.7
1994-04	188159.8	201747	205269.5
1994-05	225387.6	208225.4	209293.1
1994-06	210278.7	207942	209457.4
1994-07	244788.7	226818.3	214504.7

**Figure 10.26**

Moving average examples. The Avg function does the work but the Lag function and : operator set the data range.

Because the MA(3) average always applies to three consecutive months, the calculation continually moves forward one month at a time.

With moving averages, a question arises over how to handle the first data points. For MA(3) over months, what should be done with the first two months? At that point, not enough data exists to compute an average for 3 months. Should the first two be skipped (null), or should the average simply use the data available? One point for the first month, and two points for the second month. With MDX, this latter option is the easiest to create because the Avg function automatically uses the number of data points available. It is possible to create null values for the first months if desired—by using the IIF function.

The Visual Studio Designer has a template under the calculations tab that generates the syntax for creating a moving average. However, the syntax is easier to understand using an actual example. Figure 10.26 shows two moving average measures for monthly sales—one for 3 months and one for 12 months. The Avg function does the actual work of computing the average. The trick is to get the correct data range so that it moves with the month being displayed. The solution is to use the Lag function to go back two months and then use the range operator (: ) to specify all of the months from that lagged value to and including the current month. The 12-month moving average is included to show that the Lag function goes back 11 months because 11 plus the current month equals the 12 months needed.

The question of how to handle the first months—before reaching the number requested—is automatically solved by the Avg and Lag functions. The Lag function returns null values when trying to reach earlier than the starting month; and the Avg function ignores null values. So the MA(3) for the first month includes

only one value and the second month includes two but the averages are computed based only on the count available.

Several other useful functions exist in MDX, including year-to-date computations. They generally follow principles similar to the functions covered in this section. A few more exotic functions exist, including *Generate*, and *Sets*. Documentation and examples of these functions can be found on the Web. A key concept for many of them is that they create and modify sets of data. A powerful aspect of sets is that a cube itself is a set, and many functions operate on sets. Hence, it is possible to use functions to create a cube, essentially save (name) it as a set, and then apply MDX functions on those results. This iterative process is similar to saving a query and then writing another query using those results. Some problems are easier to solve by creating intermediate steps and applying calculations to those initial results.

## Summary

---

Multidimensional expression (MDX) queries are a powerful way to retrieve data from OLAP cubes. MDX commands rely on four basic clauses: *WITH* to define calculations, *SELECT* to specify cube axes and dimensions, *FROM* to indicate the cube data source, and *WHERE* to set filters for the data. A key element of MDX queries is that sums are computed automatically—virtually all data returned consists of subtotals across selected dimensions. Dimensions, particularly hierarchies of dimensions, play an important role in MDX and in the terminology.

The *SELECT* statement contains sets of data for the various axes of a cube. Typically a cube consists of at least two axes: *Columns* and *Rows*, which can also be numbered 0 and 1. The *Columns* axis generally must contain values from the *Measures* set—specifically numbers that can be added to compute totals and subtotals. Sets in the *SELECT* statement are generally enclosed within curly braces. The cross join (\*) is an important function in MDX because it combines all elements from one set of data with all elements from a second set. For example, a cross join would be used to create a matrix or table of sales for each month against model type in the Rolling Thunder Bicycles case.

New measures or columns of data can be calculated in a *WITH* statement by assigning a new name to the variable and writing a calculation. Simple computations use basic arithmetic on data at the same level. More complex calculations can reach across levels or dimensions. For example, percentages can be computed by dividing by the total value derived from the parent of the current member. Changes from one period to the next can be computed using the *PrevMember*, *NextMember*, *Lag*, or *Lead* functions. The *ParallelPeriod* function is a powerful way of comparing data to similar levels at different points in the hierarchy tree. For example, it is useful for comparing monthly sales to sales in the similar month in the previous quarter.

MDX includes several functions to handle tree navigation, logical conditions, sets of data, statistical computations, and operations with respect to time. Additional tools, such as *string* and *value* functions also exist. Some of the more common functions include *Except*, *IIF*, *CoalesceEmpty*, *TopCount*, and *Avg*. The main syntax of MDX is designed to return values that match specified dimensions, so the *Except* function is a negation method that returns data that matches items in a list except the ones specified. The immediate if (*IIF*) function provides inline conditional tests—returning different values if the condition is true instead of false. The *CoalesceEmpty* function provides a convenient way to handle missing data

by converting null values to almost any other value. Several “Top” and “Bottom” functions exist to return a smaller set of results. TopCount returns rows up to the count specified, while TopSum and TopPercent are used to limit results based on a running total. They are used to address business questions of the form: How many states does it take to reach 50 million in sales? Moving averages are quickly computed using the Avg and Lag function. The Lag function and range operator ( : ) are used to specify the data points relative to the current location, and the Avg function computes the average of those values.

Many other functions exist to solve complex business questions. Because MDX is designed to work with sets, and OLAP cubes are also sets, difficult problems can be solved by creating intermediate cubes and writing new queries against those named sets. Several Web sites provide detailed descriptions and examples of MDX functions and capabilities.

## Key Words

---

* (cross join operator)	Member
: (range operator)	Multi-dimension expression or MDX
Avg	NextMember
axes	NON EMPTY
BottomCount	ParallelPeriod
Children	Parent
columns	PrevMember
cross join	root
cube	rows
dimensions	SELECT
EXCEPT	set
FROM	slice
hierarchical	TopCount
IIF	TopPercent
Lag	TopSum
Lead	tuple
Leaves	WHERE
level	WITH MEMBER
measure	YTD

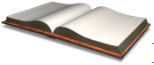
## Review Questions

---

1. What is a dimension hierarchy?
2. How are OLAP cube measures different from dimensions?
3. Compared to SQL, how is MDX better for dealing with OLAP cubes?
4. What is the main structure of an MDX query?
5. What would be the basic MDX query structure to list total sales by state for half of the states?
6. What is the difference between using several dimension axes and a cross join?
7. How do calculations involving percentages rely on the automatic sub-totals in MDX?
8. Briefly explain how the following functions work and give an example situation: Parent, PrevMember, Lag, Lead.
9. How is the ParallelPeriod function useful for common business accounting reports that compare data to earlier time periods?
10. How does MDX handle reverse condition where an element is to be excluded from a total?
11. Which MDX function would be best used for each of the following problems?
  - a. How many production days are required in each month to reach 50 percent of the total output?
  - b. Classify customers as “whales” or “minnows” based on whether their total purchases for a month exceed some fixed value.
  - c. Compute the percentage change in sales from year-to-year.
  - d. List all of the months in a year within a dimension hierarchy.
  - e. Treat missing values as 1 instead of null.
  - f. When listing sales by month keep a running total of year-to-date sums.
  - g. Compute a moving average.
  - h. List the ten states with the lowest sales for each year.

## Exercises

---



### Book

1. Create and save the OLAP cube for Rolling Thunder Bicycles as explained in the chapter.
2. Use the MDX references to find two functions not covered in the chapter. Explain the purpose of each function, and create a sample query that uses the function.
3. Run the following query in both SQL Server Management Studio and the Visual Studio Cube Browser. Explain any differences and rewrite the query so that it runs the same in both tools. `SELECT ([Measures].[Sale Price], [State].[CA]) ON 0 FROM [RT Sales]`.
4. Modify the query in Figure 10.17 to compute the percentage change from one year to the next.
5. Modify the query in Figure 10.19 to obtain the values from a year ago for each month.
6. Modify the query in Figure 10.21 to list the sales by model type by state but exclude models Hybrid and Track, and exclude Alaska and Hawaii (AK, HI).
7. Create a query that shows the ten states with the lowest sales in 2010.
8. Modify the query in Figure 10.24 do perform the cross join as: `State * TopCount( Year)` and explain what the results mean in the context of the other two versions from that figure.



### Rolling Thunder Database

9. Create a new table (Region) that assigns states to regions (Northeast, Midwest, Northwest, Southeast, South, Southwest, and West). Create a new dimension hierarchy based on customer location using region, state, and city. Write the MDX query to examine sales by location over time.
10. Write the MDX query to compute the percentage change in sales over time (year) for each employee/sales person.
11. Create three-month moving average columns for the model types but exclude Hybrid and Track because of limited sales of those models.
12. Using the data from the previous exercise, transfer the results to an Excel PivotTable (copy/paste), reformat the cube with months as rows and the model types as columns. Plot the moving averages.

- Using quarterly totals instead of monthly values, create the MDX query to compute quarterly, YTD, and prior YTD totals for Sale Price. Copy at least a couple of years' worth of data to Excel and test the computations.



## Diner

- By week and gender, compute the average bill per person (diner).
- Similar to the previous exercise, but edit the data source view and add a named calculation to the table: BillTotal/Number and add that new column to the cube measures (drag it in the designer). First, write a similar query in SQL and find the average values for week 1. (Hint: Use DatePart(wk, DineDate)). Now, write the MDX query that generates the same averages as the SQL query. Explain why these values are different from those in the previous exercise.



## Corner Med

- Create an OLAP cube for the financial aspects of Corner Med—using the Visit and Visit Procedures tables. Create the MDX statement to compute the monthly totals paid by the insurance company and the patient separately. Then include the year-to-date total that combines both values.
- Because two Measures tables exist, the Time hierarchy has to be connected to both separately. The link from Time to Visit is handled as a “regular” relationship as usual. The link from Time to Visit Procedures is handled as a many-to-many relationship through the Visit table. Create this second relationship and create the MDX query to show the Procedure Amount charged per month.
- Create the MDX query to list the number of procedures performed each month by each category of employee.
- Create a three-month moving average of the total amount charged for procedures.
- Compare the number of patient visits in a month for patients who use tobacco and those who do not. List the count of each type by month and show the corresponding percentage of the number of visits.



## Basketball

21. Create a new OLAP cube based on the view for Team Game Totals. You might have to build a named query using the saved view in order to set the primary key correctly. Create a time dimension hierarchy that at least includes year and week; use a fiscal year for the season instead of a calendar year. Choose a team and examine the total points per game scored by week.
22. Choose a team and examine items that might be different for wins and losses [Won Loss]. Consider at least points per game, three points per game, and total rebounds [TRB].
23. List the teams by conference and compute the total points scored by each team per year. Within each year, compute the percentage of points scored by each team for its conference. That is, find the highest-scoring teams per conference. Use the Order function to sort the rows.



## Bakery

24. Build an OLAP cube for sales that includes at least week and product category as dimensions. Note that because the SaleDate includes time, it will not join with a standard Time table. Hence, it is necessary to create a new named calculation SaleDateAlone: Convert(date, SaleDate). Also, remember to create SaleAmount=SalePrice\*Quantity in the Sale Item table. Use MDX to create a four-week moving average for total sales.
25. Use MDX to create a cube that shows total sales quantity (not value) by year and category. Show the percentage of sales by category within each year.



## Cars

26. Create a new OLAP cube with just the Cars table (not sales). Use MDX to create a cube that shows the average highway MPG by manufacturer (Make).
27. Use MDX to create a cube that lists the Drive type within each car Category and shows the average weight and average highway MPG. As a separate MDX query, compute the correlation coefficient between weight and highway MPG across all cars [Car ID].





## Teamwork

28. Each team member should choose an MDX function not covered in this chapter and trade it with a team member. Each person then uses the received function in an MDX query for one of the databases.
29. Each team member should choose one database and write a business question to be answered with an MDX query. Trade problems with another team member and solve the received problem.
30. Each person should research a different software tool that supports MDX. Briefly summarize any major differences or features in the implementation. Combine the comments and draw a table comparing the tools.

## Additional Reading

---

IBM, *MDX Language Reference*, [http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=%2Fcom.ibm.dwe.cubemdx.doc%2Fmdx\\_concepts.html](http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=%2Fcom.ibm.dwe.cubemdx.doc%2Fmdx_concepts.html). [IBM's reference documentation for MDX.]

Carl Nolan, 1999, "Manipulate and Query OLAP Data Using ADOMD and Multidimensional Expressions," *Microsoft Systems Journal*, August. <http://www.microsoft.com/msj/0899/mdx/mdx.aspx>. [Introduction to MDX with simple examples and some discussion.]

Microsoft, *Multidimensional Expressions (MDX) Reference*, <http://msdn.microsoft.com/en-us/library/ms145506.aspx>. [Microsoft's main reference site for MDX, best for function reference.]

Microsoft, *MDX (SQL Server 2000)*, [http://msdn.microsoft.com/en-us/library/aa216767\(v=sql.80\)](http://msdn.microsoft.com/en-us/library/aa216767(v=sql.80)). [Microsoft overview and examples for MDX.]

SAS, *SAS 9.2 OLAP Server MDX Guide*, <http://support.sas.com/documentation/cdl/en/mdxag/59575/HTML/default/viewer.htm#titlepage.htm>. [MDX introduction and examples from SAS.]

Russ Whitney, 2001, "MDX by Example," *SQL Magazine*, <http://www.sqlmag.com/article/quering/mdx-by-example>. [Interesting business problems with MDX samples. Some sample code might not work with current versions.]

**\* (cross join operator)** In MDX, the operator used to signify that all rows in the first set are matched to all rows in the second set.

**: (range operator)** In MDX, the operator used to specify a range of values. Typically the items are part of a time series so a function might use [Month].CurrentMember(2):[Month] to specify values from two months back through the current month.

**a priori algorithm** A data mining tool used in association analysis. It reduces the number of computations by dropping all combinations of attributes with less than a specified level of support.

**addition rule** Probability rule that states the probabilities of mutually exclusive events can be added together. More generally, the probability of either of two events occurring is  $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ , or the sum of the probabilities minus the probability of both events occurring together.

**agglomerative** A clustering approach that starts with the smallest level of items and combines them together to create groups. the opposite of divisive clustering.

**aggregation** Summarizing over rows of data. SQL aggregate functions include Sum, Count, and Average. Often used with a GROUP BY clause to compute subtotals. OLAP cube browsing displays aggregate totals interactively.

**Akaike information criterion (AIC)** A goodness of fit measure of forecast error based on the squared difference between observed and predicted Y values. In time series analysis, it measures the error from the autoregressive component and becomes smaller as the AR variance decreases. It is used to compare various models. In general,  $AIC = 2k + n \cdot \ln(\text{residual sum of squares}/n)$ , where k is the number of parameters and is the likelihood value. Also see Schwarz criterion.

**antecedent** The left side of a rule in the statement of an association rule. Typically read as indicating that purchases of the items on the left side lead to purchases of items on the right (consequent) side.

**application programming interface (API)** A function that defines a specified task in an application. It is designed to be used by

external programs so they can use features of the base application. Parameters are passed into the application through the function and results returned via the function.

**arrangements** The number of ways of arranging a set of items. Used for determining probabilities in terms of relative frequency. The number of ways of arranging n items if there are no duplicates is simply n! (n factorial).

**association rule** A relationship determined from the data that indicates which events or purchase items occur together. Typically written: antecedent -> consequent, the left side implies or indicates that the right side event happens with some estimated level of probability.

**attribute relationship** Hierarchical dimensions can be created with a hyper cube browser. In Microsoft's system, the attributes within that hierarchy should be connected via defined relationships to specify the various levels. For example, a common date hierarchy runs: Date -> Month -> Quarter -> Year.

**auto regression** In time series analysis an auto regressive relationship is a relationship between the variable in the current time versus lagged time periods:  $Y_t = a_0 + a_1 Y_{t-1} + a_2 Y_{t-2} + \dots + a_p Y_{t-p} + \epsilon_t$

**auto regressive integrated moving average (ARIMA)** A time series estimation technique from Box and Jenkins that estimates coefficients for the auto regression (AR) and moving average (MA) components. The integrated (I) element constitutes the removal of the trend through differencing.

**autocorrelation function (ACF)** A chart used in time series analysis to help determine the number of lags to use for the moving average component of an ARIMA model. It plots the auto correlation values for each lag value. If the ACF cuts off at a specific lag while the PACF dies down, the cutoff point is a good estimate for the maximum MA lag.

**auto-regressive tree with cross prediction (ARTxp)** Microsoft's primary time series analysis tool that uses a decision tree approach and cross correlations to predict time series values. The decision tree component identifies break points within the data that have different effects on the time series. An ARIMA model

is mixed with the decision tree to predict the series farther out in time.

**Avg** A common abbreviation for the average function--particularly in the MDX language.

**axes** From the mathematical concept that measures the coordinates of a chart, axes represent the dimensions of an OLAP cube.

**Bayes' theorem** In simple form:  $P(A | B) = P(B | A)P(A) / P(B)$ . When B is a compound event consisting of many similar events, the denominator is written  $\sum P(B | A=i) P(A=i)$ . The theorem is commonly used to find values when events are sequenced or some data is unavailable. Knowing only the probabilities  $P(B | A=i)$ , it is possible to compute the reverse  $P(A | B)$ . The rule is easiest to see with a decision tree or contingency table.

**Bayesian information criterion** See Schwarz criterion.

**Bayesian probability** A way of looking at probability and statistics where probability is subjective and Bayes' theorem is used to update the estimate of probabilities. Rewrite Baye's Theorem as:  $P(A | B) = P(A) P(B | A) / P(B)$ .  $P(A)$  is the initial or a priori estimate and  $P(A|B)$  is the posterior probability after the information from event B has been incorporated.

**BETWEEN** A portion of a SQL statement used to specify a lower and upper bound in a WHERE clause. Commonly used for dates, such as OrderDate BETWEEN 01-Jan-2010 AND 31-Dec-2010.

**bin** A category used to discretize continuous data or to compute a frequency distribution. It is generally defined by upper and lower limits on the underlying data.

**Boolean algebra** Mathematical analysis often applied to query conditions that focuses on the role of AND, OR, and NOT connectors. Complex conditions require thought and testing to ensure all conditions are defined correctly.

**Bootstrap** A process of expanding limited data to more cases enabling small data sets to be used for more complex analysis. The initial data is treated as a distribution and new samples are generated by randomly drawing data from that sample distribution. It is a common practice for small samples, but it is always better to obtain more data if possible. The small sample might not accurately

represent the true data.

**BottomCount** A function in MDX similar to TopCount that returns the specified number of items from the bottom of a sorted list. The items represent totals by defined attributes.

**Business Intelligence Development Studio** Microsoft's client tool used to define analyses for data mining and to create data cubes for browsing. It runs in conjunction with Visual Studio and can be installed from the SQL Server installation process.

**calculation** Computations on data—commonly performed within a query to operate on data contained within one row of a table.

**categorical attribute** A discrete measure, often written as text categories, such as gender. When tools require numeric data, CASE or IF statements can be used to assign numbers to each category value. Clustering is difficult with categorical or nominal values because distance measures are arbitrary.

**causality** A relationship specified in a model where one event causes a second event to happen. The second event always occurs as a result of the first one. In comparison to correlation, correlation is an observed relationship that two variables appear to be related, but it does not mean that causality exists. Commonly, the two variables might both depend on a third variable.

**central limit theorem** The fundamental theorem in statistics which states that the distribution of the average for any random variable approaches the normal distribution with a sufficiently large number of observations.

**chaotic** A series or events that exhibit strong variations. In particular, from chaos theory, small changes in independent variables can result in large, possibly discontinuous jumps in the dependent variable.

**children** In a hierarchical or tree structure, a node has children or nodes further down (more detail) the tree--closer to the leaves.

**classification** The act of placing data into classes or categories. Typically based on attribute values and usually created via a logistic-type regression, neural network, or decision tree. For example, classify customers on the basis of payment history.

**classification matrix** A Microsoft tool to view the accuracy of classification tools. It compares the actual number (column) to the predicted number (row) of items placed into each category based on the current model. Cells on the main diagonally contain counts of the correctly predicted classes.

**CLUSTER\_COUNT** An option parameter in the Microsoft clustering tool that enables the analyst to specify the number of clusters to be estimated. A value of zero (0) tells the routine to heuristically find the appropriate number of clusters.

**clustering** A set of data mining techniques that attempt to define groups or clusters of data based on attribute values. The goal is to identify groups that have small distances from other members of the group relative to larger distances to other groups. It can also be applied to sequences.

**CLUSTERING\_METHOD** An option parameter in the Microsoft clustering tool that enables the analyst to specify the particular clustering method to be used. The choices are 1=Scalable EM (default), 2=Non-scalable EM, 3=Scalable K-means, and 4=Non-scalable K-means. The non-scalable options can be used only with small numbers of observations, so the main choices are 1 and 3 between the EM and K-means algorithms.

**column** In a database table, a column represents an attribute or dimension of the data object. For instance, Name and Phone would be typical columns in a Customer or Employee table.

**combination** The number of arrangements of a set of items when  $k$  items are pulled from a set of  $n$ . The specific ordering of the items does not matter.  $C(n, k) = n! / (n-k)! k!$  This term matches that of the binominal distribution. The Excel function is `Combin(n, k)`.

**combinatorial search** A clustering search method that tries all combinations of points to determine the best clusters. It is an expensive and time-consuming approach but can obtain the most accurate values. Most K-means cluster methods use combinatorial searches for at least part of the solution.

**comma-separated values (CSV)** A relatively standard method of storing data in flat files for transfer to other systems. Data for one

observation is stored in a single row in the file. Within the row, data for the columns are separated by commas—although most tools have options to specify the delimiters and separators.

**conditional probability** The probability of some event happening given that another event has already occurred. Written  $P(A | B)$  and read as the probability of  $A$  given  $B$ . It is easiest to see using a contingency table or decision tree. Mathematically,  $P(A | B) = P(A \cap B) / P(B)$ .

**confidence** Used in association analysis, confidence is a measure of the probability that a rule is true. In probability terms for a rule  $A \rightarrow B$ , confidence is the conditional probability  $P(B | A)$ .

**consequent** The right-side of an association rule, or implication event that is being predicted.  $A \rightarrow B$  makes  $B$  the consequent event based on the antecedent  $A$ . Association analysis estimates the probability of the rule.

**contingency table** A two-dimensional table used to display the count of the observations for two types of events (rows and columns). The values within a given cell are used to compute the joint probabilities  $P(A \text{ and } B)$ . The margin totals show the probabilities of each specified event.

**continuous data** Continuity means data can take any value and there are no gaps or jumps between values. Measures such as weight, height, or distance are common forms of continuous data. Even though a measuring device lacks infinite resolution, the underlying data could take on any value. Measures such as time could be continuous, but are sometimes handled as discrete data—depending on the desired model and the measurement method. If time is measured in seconds or fractions of a second, it is likely to be continuous. If it is measured in months or years, it might be handled as a discrete variable. The point is that the choice of continuous or discrete data is dependent on the problem and the model.

**correlation** An observed relationship between variables. If variable  $Y$  increases when  $X$  increases, it represents a positive correlation. Correlation between two variables is measured with the simple correlation coefficient. Regression techniques are used to measure correlation across several variables.

Correlation is a statistical observation, it does not imply that causation exists. However, it can sometimes be used to test the validity of a theoretical model that explains causation.

**correlation coefficient** A measure of the correspondence or association between two variables. It essentially computes the variation of the two variables compared to the standalone variance of each variable.

**critical value** In hypothesis testing, the critical value is the point at which the null hypothesis is rejected. The value is found from the distribution function or tables. With standard normal data, common critical values are 1.96 for a two-tailed test at 5 percent error, and 2.58 for a two-tailed test at 1 percent error. In Excel, NormSInv(probability/2) and TInv(probability, degrees of freedom) provide the critical values for the standard normal and T distribution.

**cross correlation** A relationship between two time series. One series might cause changes in second (positive or negative), or the two series might be related to a third series. If one series is cross correlated with a second and the second one is easier to predict, the original series can be forecast.

**cross join** In a query, a cross join matches every row in the first table with every row in the second table. If the first table contains  $m$  rows and the second contains  $n$  rows, the resulting query will contain  $m * n$  rows. Across join is created in SQL when no JOIN condition is listed. Cross joins should be avoided except for instances with a small number of rows. Cross joins are commonly used in MDX queries.

**cross validation** Testing a model against multiple subsets of the data. Commonly achieved by splitting a model into  $k$  non-overlapping sets and estimating the same model on  $k-1$  sets—rotating the dropped set until all  $k$  combinations have been estimated.

**cross-support** A problem that can arise in association analysis, when a market basket contains one item from a high-frequency set and one from a low-frequency set. If item A appears in almost all baskets, any other item that appears could be a random event, yet the computed confidence values will make it appear to be important.

**cube** See hyper cube.

**cube browser** A software tool to enable managers and analysts to explore data summaries commonly computed within a hyper cube. Most cube browsers highlight summaries in tables using two dimensions (rows and columns). Users can drill down to see details within the subtotals, or roll up the totals to see summaries at a higher level within a hierarchy. Options exist to slice the cube to display rows that meet specified conditions.

**cumulative distribution function (cdf)** The function  $F(x)$  that returns  $P(X \leq x)$ . For discrete data, it is the sum of the probabilities up to  $x$ . For continuous distributions, it is the integral of the probability density function.

**curse of dimensionality** Many data mining tools are hard to solve when the number of dimensions or attributes is large. Clustering with K-means and association analysis are two main examples, but similar problems arise with most tools. Often, the only solution is to reduce the number of dimensions; such as examining purchases by categories (such as soda and crackers) instead of detailed items (such as Coke, Diet Coke, Pepsi, and so on).

**Cyclical variation** In time series, variations or patterns that depend on the economic cycle. For example, high points of the cycle represent higher personal income which can lead to greater sales. Requires knowledge of the business cycle to estimate—typically with a series data for gross domestic product or income.

**data** The fundamental element to be analyzed. Typically a measure of either an attribute or a fact. Data can be numeric or categorical. In most systems, a row of data represents a single observation with columns representing measures of various attributes and facts.

**data associations** Events or situations that tend to happen together. Market basket analysis is a classic situation, where the goal is to identify items purchased together. But the association concept is general and can be useful for any events.

**data definition** A set of commands that are used to define data, such as CREATE TABLE. Graphical interfaces are often easier to use, but the data definition commands are useful for creating new tables with a program.



**data manipulation** A set of commands used to alter the data. The most common commands are UPDATE, INSERT, and DELETE.

**data mining** The process of using analytical tools to scan large data sets for patterns and provide insight to analysts and managers. The process emphasizes exploration of the data. Some people differentiate between data mining, business intelligence, and business analytics; but the three terms represent aspects of the same concepts. Machine or statistical learning can also be components, where data is used to train a system to identify categories and patterns so these can be used to make future decisions. Data mining typically includes tools to analyze data as well as to search and report data.

**Data Source** The connection string that defines a link to a source of data in Microsoft Business Intelligence Studio. Each analysis begins by defining at least one data source. Multiple data sources can be created to connect to different databases.

**Data Source View** In Microsoft Business Intelligence Studio, it defines the tables and named queries that can be used in data analysis. At least one data source must be created first to define the connection to a database. Views are created using SQL syntax, but tables can be pulled from multiple data sources.

**data type** A type of data that can be held by a column. Each DBMS has predefined system domains (integer, float, string, etc.). Some systems support user-defined domains that are named combinations of other data types.

**data warehouse** A copy of transaction data stored for high-speed searching, summarizing, and analysis. Special tools, typically including many complex indexes, are often used to store the data. Bulk uploads are generally used to update the data.

**database** A collection of data stored in a standardized format, designed to be shared by multiple users. A collection of tables for a particular business situation.

**database management system (DBMS)** A tool to efficiently store and retrieve large amounts of data. Many DBMSs are based on the relational data model which stores data for entities in tables. Rows of data represent a single instance of the entity (such as customer),

and the columns identify attributes of the object, such as name and phone number.

**decision tree** A method of examining data and a tool to classify data into a tree. Each node of the tree contains a conditional statement and represents a split point for the data. For example, one node might test the gender of participants, resulting in three branches from that point: Female, Male, and Other. Decision tree tools attempt to build trees that have significant split points where the contribution of each branch to the goal is different.

**dendrogram** A graphical display of clusters created with hierarchical clustering. Often used in chemistry, the chart shows various levels of clusters. The bottom level contains the most number of clusters, the top contains the fewest clusters.

**dependent variable** A variable or attribute that responds to changes in the values of the independent variables.

**DESC** The modifier in the SQL SELECT ... ORDER BY statement that specifies a descending sort (e.g., Z ... A). ASC can be used for ascending, but it is the default, so it is not necessary.

**digital dashboard** Also called digital cockpit or executive information system. A graphical way to display selected data items using gauges, icons, and color coding. The key performance indicators are selected by managers to highlight changing data that affects goals and progress critical to making decisions. The tools include links to details to explore the underlying data.

**dimension** One attribute or characteristic of an object in a hyper cube. Determining relevant dimensions is an important step in designing a hyper cube and configuring data analysis.

**discrete data** Data that takes on specific values, but possibly an infinite number. For example, the set of integers is discrete. In many cases, the choice of discrete or continuous data depends on the problem and the model being used.

**discretizing** The process of converting continuous data into discrete categories. Some tools require discrete data, so categories or bins can be defined by specifying ranges of data. For instance, all people less than 18 versus people

18 or older. The ranges can be defined based on external factors or clusters can be used to find groups of ranges.

**distance** Most important in cluster analysis, a distance measure is used to determine how far one point is from another, or from the center of a proposed cluster. For numerical attributes, distance can be a common Euclidean measure,  $(x-y)^2$ . For multiple attributes, the individual squared differences are summed—giving equal weight to each measure. If attributes have highly different scales, the data might be scaled first. Distance between categorical data is often arbitrary, but is sometimes reduced to zero or one.

**divisive clustering** A top-down approach to clustering where the top node contains all of the data elements. At each level, the algorithm divides the existing cluster into two new clusters. Typically, the algorithm takes the point that is farthest away from the existing center and then determines which points are closer to the new point than to the existing center.

**drill down** Expanding a point of data to explore the details. A method associated with data hierarchies to examine data at a lower, more-detailed level within the hierarchy. The opposite of roll up.

**drill through** An option within many SQL Server analytical tools that enables users to select a result and obtain more detailed data for the item.

**dummy variable** A variable that is assigned discrete values (often zero and one) to represent various events or characteristics. For example, a variable Fall could be defined as 1 for the fall months and 0 for others; then used to estimate seasonal variations. Be careful when adding multiple dummy variables to a problem because they can lead to exact multicollinearity. For example, dummy variables for Fall, Winter, Spring, and Summer would cover all of the cases for a year and all four cannot be used if an intercept term is also used.

**edit distance** A method of comparing two items such as strings that consists of counting the minimal number of steps needed to convert (edit) the first item into the second item.

**eigenvalue** In the mathematics of linear algebra, it is a scalar value  $\lambda$  such that  $\mathbf{A}\mathbf{X} = \lambda\mathbf{X}$ , where  $\mathbf{A}$  is a square matrix and  $\mathbf{X}$  is a

vector of real numbers. Sometimes written as  $\mathbf{A}\mathbf{X} - \lambda\mathbf{I}\mathbf{X} = 0$ , where  $\mathbf{I}$  is the identity matrix. Eigenvalues and the corresponding eigenvectors are used to decompose a matrix into simpler elements. In decision statistics, the result is known as principal components.

**elasticity** Percent change in dependent variable ( $Y$ ) divided by percent change in independent variable ( $X$ ). If the slope is known,  $E = dY/dX (X/Y)$ . A convenient way to display change data without the dimensions so values are comparable regardless of the underlying data.

**enterprise resource planning (ERP)** An integrated computer system running on top of a DBMS. It is designed to collect and organize data from all operations in an organization. Existing systems are strong in accounting, purchasing, and HRM.

**Euclidean distance** The most common distance measure:  $d = \text{sqrt}(\sum (p_i - q_i)^2)$ . Using the squared terms, the distance measure is always non-negative. It is the standard measure taught in basic geometry.

**EXCEPT** The keyword used to define a condition that removes items from analysis and display in an MDX query.

**expectation maximization (EM)** One of the two main algorithms to identify clusters in data. K-means is the other. Responsibility functions are defined for each potential cluster based on the relative pdfs. The algorithm computes the responsibility values for each point to assign numbers for both potential clusters. The algorithm creates a soft assignment by allowing a point to belong to more than one cluster—as defined by the responsibility values. The clusters are defined in terms of the means and standard deviations.

**expected value** For discrete data,  $\sum p(x)$ . For continuous data,  $\int x p(x)$ . The mean of the distribution, or the average that would be expected after a sufficiently large number of trials. Typically written  $E(X)$ .

**experiment** A set of events or trials defined on a sample space. Clinical experiments often involve controlled environments where effects of external factors are minimized. Social or business experiments typically measure external variables and estimate the impact of those variables as well as the control variables.



**extraction, transformation, and loading (ETL)** The process of transferring data from an existing database into a data warehouse. The three steps generally need to be automated so that data can be cleaned and inserted into the data warehouse automatically on a defined schedule. Data warehouse products typically include tools to automate the ETL tasks.

**fact** An attribute of the data that can be measured and used within a hyper cube to compute summaries. Facts are specified by managers to define concepts of interest.

**factorial** Given a positive integer  $n$ , its factorial or  $n!$  is  $n*(n-1)*(n-2)*\dots(1)$ . The Excel function is Fact( $n$ ). The gamma function is sometimes interpreted as a factorial function for real and complex numbers.  $\Gamma(N) = (n - 1)!$

**forecasting** The process of analyzing data to predict values for future or hypothetical situations. Forecasting is often based on models where parameters are estimated from existing data, or on time series analysis which is used to predict future values based on trends and seasonal variations.

**FROM** The SQL SELECT clause that signifies the tables from which the query will retrieve data. Used in conjunction with the JOIN or INNER JOIN statement. Also used in MDX queries to specify the cube that is the source of the data.

**gap** An issue in evaluating sequences. The challenge is to decide whether sequences match if one of them contains essentially the same elements but has a gap of missing or different values. For example, should ABC match the sequence ABdC?

**gap statistic** A method of heuristically selecting the number of means (clusters) to use when clustering data. The number of clusters begins at  $K=1$  and the total within-cluster variance is computed. As  $K$  is increased, this value drops. Plotting the total value against  $K$  often reveals a break point, presumably indicating the natural number of clusters to be used. Various gap statistics have been defined to formalize this change and highlight the number of clusters to choose for the K-means process.

**general multiplication rule** The method of multiplying probabilities when events might be interdependent:  $P(A \cap B) = P(A | B) * P(B)$ .

**geographic correlation** A relationship between two series that are connected through location. For example, in health care, quality of care might depend on the local region.

**geographic information system (GIS)** A tool that helps evaluate data based on location and identify or highlight patterns based on geographical location.

**global positioning system (GPS)** A system that uses a set of satellites broadcasting highly-accurate time signals so that receivers can use triangulation to compute their geographical location in terms of latitude, longitude, and altitude in relation to a base geographical model. The original system was built and maintained by the U.S. military but systems are now operated by Europe, Russia, and China.

**goodness of fit** A method of testing how closely two distributions match each other. The distribution values are split into  $J$  categories. The number of observed observations within each category are counted and compared to the number of observations expected to fall within each category. The statistic  $X = \sum (O-E)^2/E$  has a chi-square distribution with  $J-1$  degrees of freedom. If the sum is too high, the null hypothesis of equal distributions is rejected.

**GROUP BY** A SQL SELECT clause that computes an aggregate value for each item in a group. For example, SELECT Department, SUM(Salary) FROM Employee GROUP BY Department; computes and lists the total employee salaries for each department.

**HAVING** A SQL clause used with the GROUP BY statement. It restricts the output to only those groups that meet the specified condition.

**hierarchical clustering** A cluster approach that begins with a single cluster and repeatedly divides clusters to compare the results at various numbers of clusters. A dendrogram is often used to show the results for multiple cluster levels.

**hierarchy** A set of levels that are used to explore data summaries. Natural hierarchies include dates (year, quarter, month, day) and location (continent, nation, state, city). Hierarchies can also be defined for specific circumstances, such as product groupings or employee/managerial levels.

**hybrid (HOLAP)** A method of storing data for hyper cubes. The base data is stored in relational tables and aggregated totals are stored in a data warehouse. In general, performance is similar to that of the ROLAP model. Compare it to the MOLAP approach.

**hyper cube** A method of summarizing data, used both to store data for high-speed retrieval and to browse summarized data. A hyper cube represents subtotals across multiple dimensions, where each dimension is one side of the cube. For example, a cross-tabulation is a two-dimensional cube that contains subtotals for dimensions by rows and columns. A three-dimensional cube would add a depth dimension. Hyper cubes can have any number of dimensions, but most tools focus on displaying values for two dimensions at a time in a table.

**hypothesis testing** The statistical process of evaluating data results. A null hypothesis is defined for a neutral state (such as a coefficient equal to zero). An error rate is specified for a Type I error (the probability of rejecting the null hypothesis if it is true)—often set at 5 or 1 percent. The test statistic is defined. When the experiment is conducted, the null hypothesis is rejected if the observations are improbable given the null hypothesis.

**identity** An option in SQL Server where the DBMS automatically generates sequential numbers to insert into a primary key column in a database table.

**immediate if function (IIF)** An MDX function used in SQL Server cubes to return values based on a condition. Useful for setting colors or creating categorical data. For example, color properties can be set using `IIF([Measures].[Discount] > 2000, 255, 0)`. The function has three parameters: 1) the condition, 2) the value to return if the condition is true, and 3) the value to return if the condition is false.

**Independent events** Events are independent if they are not directly affected by each other and their probabilities do not influence each other.  $P(A \cap B) = P(A) * P(B)$ .

**independent variable** A variable or attribute that is usually controllable and changes in values affect the dependent variable.

**index** A small file that is sorted and provide

fast access to data through key values. The sorted dimension can be searched quickly with B-Tree tools and the index points to the matching data. Multiple indexes dramatically reduce data retrieval times, but they slow down data inserts and updates. Hence, data mining tools often use a data warehouse which contains a heavily indexed copy of transaction data.

**information** Data that has been organized and put into a meaningful context. Information is used to make decisions. For example, information can be the answer to a question, or the results of an analysis that leads to a decision.

**information measure** From Shannon, sometimes called Shannon's entropy:  $H(X) = E[I(X)]$  where  $I(X) = \log(1/p) = -\log(p)$ . It is a measure of the surprise value of data. It is highest for uniformly random data—because there is no way to predict which value might arise.

**interestingness** A big question in association analysis and data mining in general. Correlations and associations can be found statistically, but the results might not be interesting or useful. Dozens of measures of interestingness exist—usually related to the surprise value of the information—but ultimately, decision makers need to evaluate results for potential value.

**itemsets** Combinations of attributes or products. A simple itemset consists of a single item, but association analysis and other tools often consider multiple combinations of items.

**JOIN** When data is retrieved from more than one table, the tables must be joined by some column or columns of data. See INNER JOIN and LEFT JOIN.

**joint events** Two or more events occurring together.

**joint probability** The probability of two events occurring together:  $P(A \text{ and } B)$  or  $P(A \cap B)$ . Using the general multiplication rule,  $P(A \cap B) = P(A | B) * P(B)$ .

**just-in-time** A manufacturing technique that relies on minimal inventories, instead relying on vendors and subcontractors to provide components just in time to be assembled. The method requires detailed communication with vendors.

**key performance indicator (KPI)** Variables that represent critical data to decision makers. The definition of a KPI typically includes the attribute measure, a trend over time, and a function to convert the measure into a graphical view—either by scaling or by categorizing the data.

**K-means** One of the two main algorithms to identify clusters in data. Expectation maximization is the other. The algorithm begins with a target of identify K-clusters. The goal is to find the best way to split the data to assign each point to a single cluster. In raw form, the process compares each point to the K clusters computing the distance to minimize the variance within each cluster. It can be a slow algorithm for large data sets.

**Knowledge** A higher level of understanding, including rules, patterns, and decisions. In an ideal system, data leads to information which leads to knowledge.

**layer** A method of displaying multiple levels or types of data. Most commonly used in a GIS to display multiple sets of data on one map. High-end systems use overlays to display multiple items on the same display.

**Lead** An MDX function used to retrieve values in the future periods of a time series. The opposite direction of the Lag function.

**leaves** The outermost nodes in a tree structure, furthest from the root. A leaf has no children.

**level** In a hierarchical tree structure it is a distance from the root. But most levels are measured in terms relative to the current level.

**Levenshtein distance** A version of an edit distance measure for sequences. Edit operations are defined such as insert, delete, and change. The distance between two sequences is defined as the minimal number of operations needed to convert one sequence into another.

**lift** A measure of impact of a rule or result. In association analysis, lift is often defined as  $P(B|A)/P(B)$ . This ratio measures the probability of item B being purchased with the rule (A already chosen) versus without the rule—B by itself.

**linear regression** A data mining tool that is a classic statistical research method.

Regression has a dependent variable and several independent variables and determines coefficients that fit the best line to the data points. The process estimates coefficients of the equation:  $Y = b_0 + b_1 X_1 + b_2 X_2 + \dots + b_k X_k$ . In effect, the coefficients identify the strength of the variables in how they affect the dependent variable. The dependent variable must be continuous.

**Logical Primary Key** Used within Microsoft's Data Source View, a named query should be assigned a logical primary key which acts similarly to a primary key in database design. The selected columns uniquely identify each row within the query.

**logistic regression** A data mining tool similar to linear regression but the dependent variable is categorical or discrete. It is typically used for classification problems. The method estimates a function which determines the probability of each Y-outcome.

**margin totals** In a contingency table, the totals of the observations or probabilities computed for a given row or column observation. Often written in the margin, the total represents the probability of the specified event occurring regardless of the value of the secondary event.

**market basket** A collection of items purchased at the same time. Association analysis can identify specific items that are commonly purchased together.

**Markov chain** A sequence of events where any next state (or item) can be defined in terms of a transition probability from the current state.

**maximum likelihood estimator (MLE)** A method of estimating coefficients for a variety of tools for evaluating dimensions. It is often a choice parameter in how models are estimated. It is one of the more robust estimation methods, but sometimes can be slow.

**mean absolute deviation (MAD)** A method of measuring distance. The traditional Euclidean measure uses a squared difference, MAD uses the absolute value of the difference. Euclidean places a stronger emphasis on outliers than does MAD.

**measure** An attribute that evaluates some fact or dimension of the problem. In most systems,

it must be a numeric attribute that can be subtotaled or averaged.

**member** In MDX it is a keyword used to signify the items of a set of data. Most commonly referenced as members or allmembers, such as [Date Hierarchy].[Year\_Month].[Date].members to reference all items in the specified date hierarchy.

**metadata** Literally, data about data. The explanation or documentation of data items. Simple metadata includes the data type and name. More complex metadata includes descriptions, source information, ownership, and security conditions.

**minimum confidence** In association analysis, the cutoff level for the confidence measure used to determine whether a rule should be displayed. It is usually a secondary measure and the levels can be changed interactively to increase or decrease the number of rules displayed.

**minimum support** In association analysis, the cutoff level for evaluating potential rules. Itemsets that fall below the specified level are dropped from further consideration. Setting the level too high can result in no rules that meet the condition. Changing the level typically requires reanalyzing the data.

**mixture model** The mixture model is the underlying method used in the EM clustering approach. Each cluster is assumed to have some unknown distribution and a given point can be assigned to multiple clusters by a linear combination of the probability functions. The linear coefficients essentially determine the percentage of weight assigned to each cluster.

**model** A simplification of reality, and an attempt to describe the interrelationship and causality between variables. Models typically are built from theory. Estimates based on models have more power and validity than basic statistical observations.

**moving average** In time series analysis, the estimation of the coefficients for the lag effects of the error terms. An average is computed across each specified interval, such as MA(3) which can be computed as  $(Y_0 + Y_1 + Y_2)/3$ , and then shifted forward one time period to compute  $(Y_1 + Y_2 + Y_3)/3$ , and so on. In the ARIMA model,  $Y_t = \mu + \psi_0 \varepsilon_t + \psi_1 \varepsilon_{t-1} + \psi_2 \varepsilon_{t-2} + \psi_3 \varepsilon_{t-3} \dots$

**multicollinearity** A problem that arises when many attributes or dimensions attempt to measure the same thing. Perfect multicollinearity arises when a collection of attributes can be written as linear combinations of each other. Most commonly encountered in regression analysis with too many similar attributes. In most cases, the easiest solution is to drop some of the attributes from the analysis.

**multidimensional expression (MDX)** A language initially created by Microsoft to define data cubes and configure data analysis. Several vendors support MDX. Most tools generate MDX automatically, so tools can be used without knowing the detailed syntax. It is useful for performing calculations.

**multidimensional OLAP (MOLAP)** A method of storing data in a data warehouse for hyper cubes. The data is cleaned and aggregations are pre-computed where possible. Joins and indexes are prebuilt, leading to some duplication. It is often the fastest method to retrieve data. Compare to ROLAP.

**multiplication rule** If two events A and B are independent, the joint probability can be computed with a simple multiplication of the two separate probabilities:  $P(A \cap B) = P(A) * P(B)$ . For example, with a fair die and random throws, the probability of obtaining any specific number is 1/6, so the probability of throwing “snake eyes” or two ones is  $(1/6)(1/6)$  or  $(1/36)$ .

**mutually exclusive** Two events that cannot happen together and have no common outcomes.  $P(A \cap B) = 0$ . Commonly used when creating discrete, non-overlapping categories.

**naïve Bayes** A data mining tool based on Bayes’ theorem. The goal is to determine which attributes have the strongest effect on a dependent variable. The method works with minimal supervision and is robust so it works well as an initial perspective on the data. The dependent and independent variables need to be discrete data.

**named calculation** Created within a table in a Microsoft data source view, a named calculation computes row-by-row values. It is similar to adding a computed column within an SQL query. By operating within the data source view, the named calculation can include data from multiple underlying sources.

**named query** Created within a Microsoft data

source view, a named query can combine data from multiple sources and perform calculations equivalent to those within an SQL query. By operating within the data source view, the named calculation can include data from multiple underlying sources.

**neural network** A collection of artificial neurons loosely designed to mimic the way the human brain operates. Especially useful for tasks that involve pattern recognition. The technique is one of the main machine learning methods and can run with minimal supervision. Effectively, the technique estimates a nonlinear relationship between input and output variables.

**NextMember** Used in MDX to step through the members in a set. Usually the members are sorted, often by a time variable. See *PrevMember* for the opposite direction.

**nominal dimension** A categorical attribute. In clustering, any distance measure of a categorical attribute is nominal because it is an arbitrary assignment.

**NON EMPTY** An MDX keyword used to stop the display of rows where the value is null or missing.

**normalization** The process of creating a well-behaved set of tables to efficiently store data, minimize redundancy, and ensure data integrity. See first, second, and third normal form.

**NOT** The SQL negation operator. Used in the WHERE clause to reverse the truth value of a statement.

**one-to-many** A common relationship among database tables. For example, a customer can place many orders, but orders come from one customer. As part of the design normalization process, many-to-many relationships are split into two one-to-many relationships.

**online analytical processing (OLAP)** A computer system designed to help managers retrieve and analyze data. The systems are optimized to rapidly integrate and retrieve data. The storage system is generally incompatible with transaction processing, so it is stored in a data warehouse. A hyper cube browser is a common way of examining data summaries.

**online transaction processing (OLTP)** A computer system designed to handle daily transactions. It is optimized to record and protect multiple transactions. Because it is generally not compatible with managerial

retrieval of data, data is extracted from these systems into a data warehouse.

**ORDER BY** The clause in the SQL SELECT statement that lists the columns to sort the output. The modifiers ASC and DESC are used to specify ascending and descending sort orders.

**order of operations** From mathematics, calculations are performed in a standard sequence. For example, multiplication is performed before addition. The order can be altered through the use of parentheses. The order becomes critical when computing values using a hyper cube. For example, dividing or multiplying data columns using a calculation on a cube rarely does what you might expect. Subtotals are computed first followed by cube calculations. Computations must often be made within queries or named calculations to be executed at the row level instead of the cube level.

**ordinal measure** A ranking such as 1, 2, 3. In clustering, distance is commonly defined by converting to a centered percentage:  $v = (i - 1/2) / M$ , where M is the highest value.

**Orthogonal** In geometric terms, it means perpendicular lines. In statistics, two orthogonal components are linearly independent. In principal components, the goal is to find orthogonal factors to describe the data with a smaller number of dimensions.

**over fitting** A classic problem with data mining and statistical testing in general. Given a set of observations, you could repeatedly build models for that data so that the sample data can be exactly explained by the model. But, the model could fail miserably at predicting the underlying population events because the model is tailored too closely to the specific sample. A simple example might consist of three data points. A quadratic curve could exactly fit those three points, but predicted values from the model might be terrible. The standard solution is to obtain large data sets and withhold part of the data to use to test the model.

**ParallelPeriod** A useful function within DMX, it is used to retrieve data from prior time periods for the same data attribute. For example, `ParallelPeriod ([Calendar].[Year].[Year], 1, [Calendar].[Year].CurrentMember)` retrieves data from the prior year. The function



works for the currently aggregated data, such as Week, Day, Month, or Quarter.

**parameter** A variable in a model or distribution that has specific meaning. The parameters are estimated from the sample data to define the exact shape of the distribution. For example, the normal or Gaussian distribution has two parameters:  $\mu$  and  $\sigma$  that represent the mean and standard deviation of the distribution and tailor it to the specific data.

**parent** In a hierarchical or tree structure, the node above the current node, which is closer to the root level. The opposite of a child. In most trees, a node can have only a single parent.

**partial autocorrelation function (PACF)** A chart used in time series analysis to help determine the number of lags to use for the moving average component of an ARIMA model. It plots the partial auto correlation values for each lag value. If the PACF cuts off at a specific lag while the ACF dies down, the cutoff point is a good estimate for the maximum AR lag term. It is the partial autocorrelation because it estimates the autocorrelation between  $Y_t$  and  $Y_{t+k}$ , with the intervening terms removed.

**permutation** The number of ways of arranging a set of items when some of them are not included. For example, the number of ways of selecting 3 items from 20 is 6840.  $P(n, k) = n!/(n-k)!$  In Excel `Permut(n, k)`. With permutations, each ordering is considered to be different (A, B, C is different from B, A, C). If ordering does not matter, use the formula for combinations.

**perspective** A defined view of the data in a Microsoft hyper cube. Multiple perspectives can be defined on any cube to limit the data available to one group of users.

**Poisson distribution** A distribution for discrete data often used to estimate the number of events occurring during a fixed period of time.  $P(X = k) = (e^{-\alpha} \alpha^k)/k!$  where the parameter  $\alpha$  would be the average number of arrivals expected during the time period and the arrival times are independent of the last event.

**posterior distribution** In a Bayesian approach with subjective probabilities, it is a resulting distribution that was improved through information obtained in an experiment.

**prediction** A forecast based on a model

estimated from observations, which requires forecast estimates of the independent variables. Predictions can also be made from time series analyses that are formed based on trends and seasonal variations.

**PredictTimeSeries Microsoft function** A function in DMX that is used to compute predicted values from a time series. The model must already be built and run. The function takes two parameters: the name of the column and the number of periods to be forecast. The text has examples for a simple forecast and a forecast with the standard deviation. The basic format is: `SELECT FLATTENED PredictTimeSeries([Column Name], 12) As Forecast FROM [Model Name]`.

**PrevMember** Used in MDX to step back through the members in a set. `Prev` stands for previous. Usually the members are sorted, often by a time variable. See `NextMember` for the opposite direction.

**primary key** A column or set of columns that identify a particular row in a table.

**principal components analysis (PCA)** A method to identify the primary orthogonal factors that identify a set of data. The factors are listed in descending order of the percentage of variation explained by each factor. The goal is to describe the data with a smaller number of dimensions.

**prior distribution** In a Bayesian approach with subjective probabilities, it is the initial probability distribution. The probability and distribution are improved through the addition of information.

**probability** (1) The relative frequency of some event occurring. (2) A subjective belief about the chance of some event occurring. The first definition is the most common; the second is the foundation of the Bayesian approach. Some basic rules must hold for probabilities: (a)  $0 \leq p \leq 1$ , (b)  $P(A \text{ or } B) = P(A) + P(B)$  if A and B mutually exclusive, and (c) the sum of the probability of all events must be one.

**probability density function (pdf)** For continuous data, the probability of any specific point  $x$  is zero, so the density function is defined in terms of the cumulative probability  $P(X \leq x)$ . The cumulative probability function is the integral of the pdf:  $F(x) = \int f(x)dx$ .

**probability distribution** For discrete data,

the listing of the event  $x$  and its associated probability function  $p(x)$ .

**probability function**  $P(X = x_i)$  for discrete data—the assignment of a probability number to each event. Equivalent to the probability mass function or probability density function for continuous data.

**probability mass function** See probability density function

**query system** A DBMS tool that enables users to create queries to retrieve data from a database. SQL is a standard query system found on many DBMSs.

**random events** The inability to specify events with complete certainty.

**random sample** A sample of observations selected from a population using some method to randomly choose the observations. All of statistical theory is based on the assumption that random chance is involved in a selection process. If sample data is selected without randomness, the results will be biased by the selection method and could be completely inaccurate.

**random variable** A function that assigns a number to every possible outcome in the sample space.

**recommendation engine** An automated process that provides recommendations of similar products to customers. Amazon in books and Netflix in movies emphasize recommendations to increase sales and rentals.

**relational OLAP (ROLAP)** Data for OLAP and hyper cubes is stored in relational database tables. The approach is often simpler to configure than MOLAP, but it requires computing subtotals on the fly for all queries. Most systems that use ROLAP include tools to pre-build some subtotals, such as Oracle's materialized views.

**relative frequency** The most common expression of probability. The number of times an event can arise divided by the total number observations. Straightforward for common games of chance such as dice. The number 3 appears once on a die of 6 sides, so the relative frequency for observing the number 3 should be  $1/6$ .

**relative risk** A measure of interestingness.

It is used in Microsoft's association analysis as a method to compare potential rules. In probability terms, the risk =  $P(B|A) / P(B|\sim A)$ . The ratio of the probability that B is selected given A is in the basket, versus B selected when A is not in the basket. The effect is similar to the formula for lift—attempting to measure the gain in the probability of purchasing B when A is present versus not present.

**responsibilities** The relative probability density functions, such as  $g_0/(g_0+g_1)$ , used in the expectation maximization clustering algorithm. The responsibility functions identify the weighting assigned to each point by each cluster.

**roll up** The process of aggregating data to a higher level in a hierarchy. The opposite of drill down in the process of browsing a hyper cube.

**root** The top level node of a hierarchical or tree structure.

**root mean square error (RMSE)** A measure of error in an estimated model. It is computed as it is written:  $RMSE = \sqrt{\sum e^2 / n}$

**row-by-row calculations** Using queries, simple calculations can be made using data on a single row at a time. Standard arithmetical operations (+, -, \*, /) are supported. These calculations are performed before any aggregation operations. A few newer systems include support for Lag and Lead operators that can use data from rows above or below the current row.

**sample mean** The average of the observed values in a sample.  $Mean = \sum(x)/n$ . The unbiased measure of the central tendency of the sample data.

**sample space** The set of all possible outcomes of an experiment.

**sample variance** The sum-of-squared deviation of the observed values in a sample.  $Variance = \sum(x - mean)^2 / (n-1)$ .

**Schwarz criterion or Bayesian information criterion (BIC)** A goodness of fit measure of forecast error based on the squared difference between observed and predicted Y values. It is used for model selection.  $BIC = -2 \ln(L) + k \ln(n)$  where  $k$  is the number of estimated parameters,  $n$  is the number of observations and  $L$  is the likelihood function. Usually simplified to  $BIC = residual\ sum\ of\ squares / error\ variance$



+  $k \ln(n)$ . Also see Akaike information criteria.

**seasonal ARIMA (SARIMA)** A variation of the time series ARIMA method where the lags for auto-regression (AR) and moving average (MA) are defined in terms of multiples of the seasonality. For example, monthly data has a seasonality of 12, so the AR term (P) would be 1 or 2 to indicate 12 or 24 months. Similarly, differencing to resolve the trend occurs at the seasonal level, so a difference of 1 is accomplished by subtracting values that are 12 months apart.

**seasonal auto-regressive (SAR)** The auto-regressive lag structure of a seasonal model but the lag terms are specified in multiples of the seasonality. With monthly data, the seasonal factor is 12, so SAR(1) refers to a 12-month lag:  $Y_t = a_1 Y_{t-12}$ .

**seasonal moving average (SMA)** The moving average lag structure of a seasonal ARIMA model where the lag terms are specified in terms of the seasonality. Moving average is based on the error (observed – predicted) values. With monthly data, the seasonal factor is 12, so SMA(1) refers to a 12-month lag:  $e_t = b_1 e_{t-12}$ .

**seasonality** Time series data often exhibits a seasonal pattern or correlations across an interval of time that corresponds to an annual period. For instance, sales typically increase at the end of the year holiday shopping season or unemployment increases in the summer months when students graduate from school.

**seasonally adjusted** Time series data is sometimes adjusted by removing seasonal patterns to make it easier to identify trends—particularly with monthly data. For example, sales for November and December might always be higher than September and October, but does that increase represent a trend or the normal seasonal pattern? Government economic data is often seasonally adjusted and it is important to be careful when using government data for time series analysis. Most analyses work better with raw or unadjusted data so the tools can determine the seasonal effect.

**SELECT** The primary data retrieval command for SQL. The main components are SELECT ... FROM ... INNER JOIN ...

WHERE. Also used in MDX queries to define the start of a query.

**sequence** A set of finite items (or states or events) arranged in a particular order. Sequences can often be written as strings where each character represents one state and the order is read from left to right.

**set** A collection of items with similar attributes. SQL tables often represent sets of data but MDX emphasizes the use of sets.

**Shannon entropy** See information measure.

**Simpson's paradox** Also attributed to Yule, the paradox states that aggregate relationships across groups can be reversed when groups are combined. For instance, it is possible that in every department (subgroup), the percentage of men is less than the percentage of women; yet in the overall combined group, the percentage of men can be greater than the percentage of women.

**skewed support** Occurs in market basket analysis when the bulk of the items have few sales and a handful of items are sold in almost every basket. It leads to issues of cross-support errors. Because some items are in almost every basket, anything else might appear statistically useful—even though the purchase of low-support items could be completely random.

**slice** In an OLAP it represents a specified section of data, usually defined by a WHERE clause. Visually, a slice represents a cross section of the cube.

**snowflake** A design approach for OLAP data and hyper cubes. It extends the star design by enabling connections to tables through multiple links.

**spurious correlation** A combination of data or events that appears to be related but can easily occur by random chance. Because data mining tests so many extreme cases, it is helpful to estimate the random chance of critical events happening.

**SQL** A structured query language supported by most major database management systems. The most common command is of the form: SELECT column list FROM table list JOIN how tables are related WHERE condition ORDER BY columns.

**SQL Server Analysis Services (SSAS)** A

collection of data mining tools provided by Microsoft that are integrated with the SQL Server database management system. The services are typically installed on a server and analyses are created using Visual Studio Business Intelligence tools on a client computer. Tools include, decision tree, naïve Bayes, regression, neural network, and time series analysis.

**SQL Server Business Intelligence (BI)** See SQL Server Analysis Services (SSAS).

**standard deviation** The square root of the variance. It is defined in the same units as the original data. From common distributions, most sample data will lie within +/- 2 standard deviations of the mean.

**star design** A design approach for OLAP data and hyper cubes. A fact table at the center contains measure attributes. Tables containing dimension attributes are connected directly to the fact table. Only one level of connection is supported with a star design, compared to extended connections supported in the snowflake design.

**statistic** A function of a random sample. Common statistics include the sample mean and variance.

**subjective probability** The Bayesian method of looking at probability. Probability values are updated based on new information using the Bayesian rule. The relative frequency approach is probably easier to understand initially, but the subjective approach is useful for many business problems.

**support** In association analysis a measure of the number of times an itemset occurs. The number of times a specified set occurs divided by the total number of observations. In probability terms, the relative frequency or an estimate of  $P(A)$  or  $P(A \cap B)$ .

**table** A collection of data for one class or entity. It consists of columns for each attribute and a row of data for each specific entity or object.

**time series** Data that is measured over time. The time period must be specified, and generally must be at fixed intervals (such as year, quarter, month, week, or day). A single time series uses data from one attribute that is consistently measured over time.

**TopCount** A function in MDX similar to BottomCount that returns the specified number of items from the top of a sorted list. The items represent totals by defined attributes.

**TopPercent** An MDX function similar to TopSum, but less similar to TopCount. The function computes totals by specified groupings and sorts them. It then adds the values from the top down and lists the items up to the point where the total exceeds a specified percentage of the grand total.

**TopSum** An MDX function similar to TopCount but with a major difference. The query computes subtotals based on defined groupings and sorts them. It then computes a running total from the top down and lists all of the items until the specified total value is reached.

**transition probability** In a Markov chain it represents the probability of moving to a given state. It is useful in analyzing sequences.

**trend** A pattern in time series data over time that exists outside of seasonal and cyclical factors.

**tuple** A term from the early definitions of a relational database it represents a collection of attribute values. Consider it as a row of data.

**uniform distribution** A probability distribution (or pdf for continuous data) that uniformly allocates the data across a fixed range. All observations are equally likely to arise. For discrete data,  $p(x) = 1/n$ . For continuous data,  $f(x) = 1/(b-a)$  where  $a$  and  $b$  are the lower and upper bounds. It is a straight horizontal line.

**unsupervised learning** A general data mining classification where the tool performs with minimal input by an analyst. For example, neural networks operate relatively unsupervised. After the dependent variable and potential independent variables are selected, the tool determines a model that fits the data. In comparison, standard regression analysis requires experience and knowledge to guide the selection of the final model.

**variance** The second moment about the mean. Or  $E[(X - \text{mean})^2]$ . The squared-deviation exhibited within the distribution. A measure of the dispersion of the probability distribution.

**view** A saved query. You can build new

queries that retrieve data from the view. A view is saved as an SQL statement—not as the actual data.

**Weka** An open source set of data mining software written in Java and available free from The University of Waikato in New Zealand (<http://www.cs.waikato.ac.nz/ml/weka>). The set contains many standard analytic tools and reads standard comma-separated-values files.

**WHERE** The SQL clause that restricts the rows that will be used in the query. It can also refer to data in subqueries. Also used in MDX to restrict the values of data to be displayed.

**wisdom** A level above knowledge. Wisdom represents intelligence, or the ability to analyze, learn, adapt to changing conditions, and create knowledge.

**WITH MEMBER** The MDX keyword used to define calculated values. All calculated (new) values must be defined within the WITH MEMBER section.

**YTD** The MDX year-to-date function that tracks through a set of time periods from the first sibling in the specified level up to and including the current point. For example, YTD([Date].[Calendar].[Date].[January 17, 2012]) returns values from January 1 through January 17. Often useful when combined with the ParallelPeriod function.

**Symbols**

- \* asterisk MDX cross join, 495
- { } braces for sets in MDX, 491
- : range operator MDX, 508
- ε, 381

**A**

- absolute value, 222
- actions, 146
- addition rule, 166
- Adventure Works, 142
- agglomerative clustering, 231
- aggregation, 58, 142, 143
- Agrawal, R., 276
- Akaike information criterion (AIC), 395
- Alberta, 217
- Algorithm Parameters, 409
- Amazon, 270
- Analysis database, 490
- analytics, 3
- AND, 54
- AnnTaylor Stores Corp., 13
- anomaly, 205
- antecedent, 273
- application programming interface (API), 469
- a priori algorithm, 16, 283
- a priori tool, 293
- a priori value, 175
- ArcGIS, 468
- ARIMA, 378, 425
  - differencing, 390
  - lag coefficients, 390
- arithmetic operators, 57
- ARMA(p), 390
- ARMAX, 421
- AR(p), 381
- arrangements, 164
- ARTxp, 406, 419, 425
  - forecast, 415
  - model, 413
- association, 271
- association analysis
  - continuous data, 288

- data, 290
  - quantity differences, 289
  - traditional tools, 292
- association rules, 273, 294
- attribute, 220
  - categorical, 223
  - ordinal, 223
- Attribute Discrimination, 348
- attribute evaluation, 338
  - neural network, 362
- attribute relationship, 126, 130
- Aunt Bessie's, 376
- autocorrelation function (ACF), 391
- auto regression, 377, 381
- auto regressive integrated moving average (ARIMA), 389
- auto regressive moving average (ARIMA)
  - traditional tools, 397
- auto-regressive tree with cross prediction (ARTxp), 406
- average, 385
- axes
  - cube, 489

**B**

- Barabba, Vincent P, 37
- bar-code scanners, 9
- Bayesian, 162
- Bayes Theorem, 172, 173, 341
  - a priori value, 175
- Bayes, Thomas, 172
- best and worst, 65
- Best Buy, 11
- BETWEEN, 51, 56
- big data, 3, 16
- Bing, 468, 472
- BinomDist, 180
- binomial distribution, 177
- bins, 437
- BLAST, 437, 458
- Boolean algebra, 53
- bootstrap, 196
- bootstrapping, 25, 358
- Borgelt, Christian, 293

- BottomCount, 504
- Box and Jenkins, 419
- Box-Jenkins, 378, 389
- Brat, Ilan, 37
- brute force, 16
- Bryon, Ellen, 37
- BULK INSERT, 447
- Bureau of Economic Activity (BEA), 393
- business intelligence, 3
- Business Intelligence Development Studio, 109

**C**

- C#, 61
- calculated measures
  - MDX, 495
- calculation, 134
- CASE, 153, 317
- cases, 30
  - Bakery, 32
  - basketball, 32
  - Cars, 32
  - Corner Med, 31
  - diner, 31
  - NBA, 32
  - Rolling Thunder Bicycle Company, 30
- Cast, 312
- categorical attribute, 223
- categories, 437
- causality, 308
- causation, 27, 187, 322
- Census Bureau, 459
  - demographic data, 434
- central limit theorem, 195
- CGAT, 435
- chaotic, 381
- ChiDist, 194
- children, 484
- Chi-square distribution, 194, 204, 395
- Chi-square hypothesis tests, 203
- classification, 309
  - sequences, 436, 439
- Classification Matrix, 340, 356
- cluster

- hierarchical, 229
  - sequences model, 438
  - cluster analysis, 217
  - Cluster Characteristics, 245, 453
  - CLUSTER\_COUNT, 249, 451
  - Cluster Diagram, 244
  - Cluster Discrimination, 453, 454
  - clustering, 217, 218, 345
    - categorical data, 258
    - data, 236, 239
    - discrete data, 237
    - Microsoft, 241
    - missing data, 238
    - sequences, 436
    - Weka, 260
  - CLUSTERING\_METHOD, 249
  - ClusterProbability, 247
  - Cluster Profile, 244, 453
  - CoalesceEmpty, 503
  - columns, 43, 489
  - combination, 165
  - combinatorial search, 224
  - comma-separated values (CSV)
    - SQL Server Management Studio, 399
  - comma-separated-values (CSV), 22, 30, 237, 446
  - comparison
    - time series models, 425
  - concatenation, 321
  - conditional color, 135
  - conditional probability, 171, 341
    - continuous, 182
  - confidence, 277
  - confidence interval, 198
  - consequent, 282
  - contingency table, 170, 341
  - continuous data, 163, 181
  - CONVERT, 51
  - correlation, 308, 463
    - pair-wise, 254
  - correlation coefficient, 186, 223
  - correlogram, 400
    - residual, 402
  - counting
    - order does not matter, 165
    - order matters, 164
  - counting and combinations, 163
  - covariance, 344
  - CREATE TABLE, 446
  - CREATE VIEW, 69
  - critical value, 202
  - cross correlation, 380, 392
    - linear regression, 417
  - cross join, 49, 494
  - cross-support, 287
  - cross validation, 27, 325
  - cube, 482
    - actions, 146
    - aggregations, 142, 143
    - calculation, 134
    - partitions, 142
    - perspective, 134
    - structure, 120
  - cube browser, 81, 96, 102
    - SSAS, 119
  - cumulative distribution, 179
  - cumulative distribution function (cdf), 180
  - currencies, 142
  - curse of dimensionality, 282
  - customer behavior, 440
  - customers, 218
  - cyclical, 379
- D**
- Dallas Cowboys, 481
  - dangers, 24
    - bad data, 25
    - estimation instability, 28
    - human error, 24
    - insufficient data, 25
    - model instability, 29
    - over fitting, 26
  - Dartmouth Atlas of Health Care, 463
  - data, 3, 16
    - associations, 271
    - smoothing, 384
  - database, 17, 42
    - Corner Med, 258
    - design, 82
    - relational, 43
    - structure, 46
  - Database
    - Bakery, 272
    - cars, 238
  - database design
    - first normal form, 86
    - second normal form, 86
    - third normal form, 87
  - database management system (DBMS), 3, 42
  - data definition, 72
  - data manipulation, 72
  - Data Mapping Wizard, 465
  - data mining, 3
    - goal, 310
  - data reduction, 233
  - data source, 240
  - data source view, 241
  - data type, 45
  - data warehouse, 19, 43, 99
    - models, 101
  - decision tree, 308, 309, 313, 320, 323, 349, 406, 419
    - data, 352
  - degrees of freedom, 204
  - DELETE, 74
  - demographic data, 466
  - dendrogram, 232
  - dependent, 311
  - dependent variable, 27, 423
  - DESC, 52
  - detail section, 79
  - diagnostic tool, 387
  - differencing, 390
  - digital dashboard, 96, 481
  - dimension, 102, 120, 483
    - evaluation, 309
    - hierarchy, 103, 107, 483
    - nominal, 223
    - problem of size, 282
    - reduction, 233
    - structure, 132
    - time, 124
  - dimension evaluation
    - data, 311
  - discrete, 163, 444
  - discrete data, 176
  - discretize, 437
  - discretized data, 313
  - discretizing, 163

- distance, 222
  - DISTINCT, 58, 444
  - divisive clustering, 230
  - DNA, 435, 442
  - drill down, 81, 95, 108
  - drill through, 147, 354
  - drop-down list, 81
  - Duhigg, Charles, 37
  - dummy variable, 381, 423
  - Dvorak, Phred, 37
- E**
- e-commerce, 9
  - econometricians, 316
  - edit distance, 436
  - eigenvalues, 235
  - elasticity, 327
  - employees, 12
  - endogenous, 311
  - enterprise resource planning (ERP), 19, 42, 376
  - entity-relationship diagram (ERD), 68
  - entropy, 205
  - error term, 381
  - Esri, 468
  - Euclidean measure, 222
  - evaluate dimensions, 309
  - Excel, 119, 144, 180
    - linear regression, 318
  - EXCEPT, 501
  - EXECUTE, 446
  - exogenous, 311
  - expectation maximization (EM), 228
  - expected value, 183
  - experiment, 176
  - exponential, 368
  - exponential growth, 388
  - extensible markup language (XML), 46
  - extraction transformation and loading (ETL), 19
  - extraction, transformation, and loading (ETL), 100
- F**
- fact, 102
  - Federal Aviation Administration (FAA), 41
  - federal government
    - data sources, 473
  - finance, 4, 27, 220
  - first normal form, 86
  - forecast, 377, 378, 404, 415
  - forecasting, 3
  - FormatString, 121
  - four nines, 168
  - Friedman, Jerome, 38
  - FROM
    - MDX, 489
  - function, 60
    - length, 61
    - Max, 65
- G**
- gap
    - sequences, 436
  - gap statistic, 227
  - Gaussian, 189
  - General Motors, 6
  - general multiplication rule, 171
  - Generic Content Tree Viewer, 362
  - geographic analysis, 435
  - geographic correlation, 460
  - geographic information system (GIS), 434, 459
  - global positioning system (GPS), 459
  - Goodman, Peter S., 38
  - goodness of fit, 203, 204
  - Google, 12, 459, 468, 471
  - Google Analytics, 448
  - Google map, 147
  - Göransson, H., 38
  - gretl, 22, 400, 419
  - GROUP BY, 61, 63, 319, 335, 396, 408, 487
  - Gustafsson, M.G., 38
- H**
- Hastie, Trevor, 38
  - HAVING, 63
  - HIDDEN\_NODE\_RATIO, 361
  - hierarchical
    - dimension, 483
    - hierarchical clustering, 230
    - hierarchical clusters, 229
    - hierarchy, 103, 107
      - location, 128
      - product, 274
      - time, 123
    - high-frequency trading, 5
    - Hopper, Max, 3
    - Houston Pawn Shops, 434
    - HTML, 46
    - Hudson, Simon, 217
    - human biases, 160
    - human capital, 13
    - hybrid (HOLAP), 102
    - hyper cube, 3, 116
    - hypergeometric distribution, 178
    - hypotheses, 3
    - hypothesis testing, 200
- I**
- identity, 44
  - Ilan, 37
  - immediate if function
    - MDX, 502
  - immediate if function (IIF), 135
  - independent, 167, 311
  - independent variable, 27
  - index, 18, 98
  - information, 3, 174, 343, 351
    - Bayes Theorem, 174
  - information measure, 205
  - INNER JOIN, 67
  - INSERT, 73
  - instability
    - estimation, 28
    - model, 29
  - interestingness, 277
  - internationalization, 140
  - IP address, 438
  - Isaksson, A, 38
  - itemsets, 277
- J**
- Javascript, 469, 471
  - Johnson, Avery, 38
  - join



- LEFT JOIN, 71
  - many tables, 68
- JOIN, 66
  - joining tables, 66
  - joint events, 167
  - joint probability, 167, 169, 170
    - continuous, 182
  - just-in-time, 12
- K**
  - key column, 312, 321
  - key performance indicators (KPI)
    - status expression, 152
  - key performance indicators (KPI), 96, 149
    - trend expression, 152
  - key value, 44
  - K-means, 224, 250
  - knowledge, 4
  - Koudsi, Suzarne, 38
- L**
  - lag, 381, 386
  - lag limits, 390
  - layer
    - map data, 460
  - learning
    - machine, 14, 16
  - leaves, 484
  - LEFT JOIN, 71
  - levels, 484, 489
    - dimension, 273
  - Levenshtein distance, 436
  - lift, 277
  - LIKE, 56
  - Lilienfeld, Scott, 160
  - linear regression, 308, 309, 313, 381, 422, 426
    - cross correlation, 417
    - goals, 314
  - Linear Regression, 313
  - local geography, 461
  - location-based data, 435
  - location hierarchy, 128
  - Logical Primary Key, 259
  - logistic regression, 308, 313, 330
  - Log Parser 2.2, 445
- Lohr, Steve, 38
- M**
  - machine learning, 3, 16
  - MapPoint, 464
    - Data Mapping Wizard, 465
  - margin totals, 170
  - market basket, 16, 270, 271
  - market basket data structure, 291
  - marketing, 6, 220
  - Markov chain, 439
  - materialized view, 100
  - maximum likelihood estimator (MLE), 351
  - MAXIMUM\_SEQUENCE\_STATES, 451
  - McClelland, James L., 38
  - MDX, 482
    - calculated measures, 495
  - MDX function, 501
    - Avg, 508
    - CoalesceEmpty, 503
    - EXCEPT, 501
    - IIF, 502
    - NextMember, 498
    - ParallelPeriod, 499
    - PrevMember, 498
    - TopCount, 504
    - TopPercent, 504
    - TopSum, 504
    - YTD, 506
  - MDX structure
    - FROM, 489
    - SELECT, 489
    - WHERE, 489
    - WITH MEMBER, 489
  - mean, 191, 196
    - sample, 196
  - mean absolute deviation (MAD), 326
  - measures, 102, 195, 239, 482
    - continuous, 163
    - discrete, 163
  - member
    - MDX, 489
  - metadata, 102
  - Microsoft Generic Content
    - Tree Viewer, 456
  - Microsoft MapPoint, 460, 464
  - Miner3D, 240
  - minimum confidence, 284
  - minimum support, 284
  - Mining Accuracy Chart, 324, 337
  - Mining Legend, 244, 411
  - Mining Models, 249
  - Mining Model Viewer, 362
  - Mining Structure, 321, 353
  - missing data, 238, 313, 317
  - mixture model, 227
  - model, 4, 12, 29, 309
    - association analysis, 274
    - data warehouse, 101
    - time series, 379
  - model comparison, 365
  - Morgenson, Gretchen, 38
  - Morrison, Scott, 38
  - moving average, 384
    - MDX, 507
  - multicollinearity, 29, 233
  - multidimensional expression (MDX), 150
  - multidimensional OLAP (MOLAP), 101
  - multi-dimension expression (MDX), 482
  - multinomial, 333
  - multiple tables, 66
  - multiplication rule, 167
    - general, 171
  - multivariate normal distribution, 192
  - mutually exclusive, 166
  - MySQL, 22
- N**
  - Nabisco, 9
  - naïve Bayes, 308, 313, 340
  - named calculation, 112, 113
  - named query, 114, 320
  - National Center for Biotechnology Information, 458
  - National Football League (NFL), 481
  - National Institute for Health, 458
  - natural language, 43
  - Netflix, 9, 32, 269



- competition, 10
- neural network, 16, 308, 313, 358, 359
  - data, 361
  - goals, 359
- Neural Network Model
  - Viewer, 338
- neuron, 359, 360
- NextMember, 498
- node, 321
- nominal, 223
- NON EMPTY
  - MDX, 492
- nonlinear
  - complications, 368
- nonlinear dangers, 389
- nonlinear relationships, 16
- nonlinear trend, 387
- normal distribution, 189
- normalization, 82
- NormDist, 190
- NOT, 54
- NULL, 57, 313
- null hypothesis, 201

**O**

- O'Connell, Vanessa, 38
- ODBC, 22
- OLAP
  - snowflake design, 107
  - star design, 105
- OLAP cube, 396, 482
- one-to-many relationship, 83
- online analytical processing (OLAP), 19, 96, 482
- online transaction processing (OLTP), 18, 98
- opacity, 469
- openstreetmap.org, 473
- OR, 55
- ORDER BY, 52, 115
- order of operations, 136
- ordinal measure, 223
- Oreo, 9
- orthogonal, 233
- over fitting, 26, 196

**P**

- ParallelPeriod, 151, 499
- parallel processing, 24
- parameter, 183
  - auto regression, 383
- parent, 484
  - values in MDX, 497
- partial autocorrelation function (PACF), 391
- partitions, 142
- pattern matching, 436
- peak load, 179
- percentage, 137
  - MDX, 497
- performance
  - OLAP cube, 142
- periodicity, 412
- permutation, 164
- perspective, 134, 138
- Pfizer, 7
- PivotChart, 144
- PivotTable, 119, 144, 494
- Poisson distribution, 179
- polynomial, 368
- posterior distribution, 343
- PostgreSQL, 22
- prediction, 14, 246, 309, 366, 440, 457
  - decision tree, 355
  - logistic regression, 339
  - naïve Bayes, 348
  - neural network, 363
  - regression, 328
- Prediction Function, 247
- PREDICTION JOIN, 416
- predictive software, 307
- PredictNodeID, 416
- PredictTimeSeries, 415
- PredictVariance, 416
- PrevMember, 498
- primary key, 44, 66, 79, 87
- principal components analysis (PCA), 233
- prior distribution, 343
- probability, 15, 161
  - addition rule, 166

- conditional, 171
- frequency, 166
- joint, 170
- multiplication rule, 167
- relative frequency, 161
- rules, 166
- subjective, 162
- probability density function (pdf), 181
- probability distribution, 177
  - binomial, 177
  - chi-square, 194
  - hypergeometric, 178
  - normal or Gaussian, 189, 227
  - Poisson, 179
  - T (Student's T), 192
  - uniform, 188
- probability distributions, 176
- probability function, 177
- probability mass function, 177, 181
- problems
  - nonlinear, 389
- Providian, 5

**Q**

- Q-statistic, 395
- quality assurance, 41
- query, 47, 78, 101
  - basics, 49
  - editor, 50
  - four questions, 47
  - saved, 69
- query-level calculation, 137
- query system, 42

**R**

- random, 161
- random chance, 27
- random error, 27, 386
- random events, 161
- random paths, 437
- random sample, 195
- random variable, 177
- recommendation engine, 271

- Reebok, 481
  - relational database, 43, 109
  - relational OLAP (ROLAP), 102
  - relationship, 111, 309
    - attribute, 130
    - data source view, 115
    - one-to-many, 83
  - relationships, 15
  - relative frequency, 161
  - relative risk, 278
  - repeating section, 86
  - REPLACE\_MODEL\_CASES, 416
  - report
    - break item, 79
    - detail section, 79
    - query, 78
    - wizard, 80
  - Reporting Services, 75
  - reports, 75
  - report wizard, 76
  - residual, 402
  - responsibilities, 228
  - retail stores, 8
  - Ritchie, Brent, 217
  - ROLLBACK, 74
  - Rolling Thunder Bicycle Company, 44, 319, 345, 391, 463
  - roll up, 108
  - root, 484
  - root mean square error (RMSE), 326, 358, 395
  - row, 43, 489
  - row-by-row calculation, 58
  - R-squared, 323
  - R System, 22
  - rules, 270
  - Rumelhart, David E., 38
- S**
- sample, 195
    - mean, 196
    - variance, 196
  - sample space, 176
  - sample variance, 196
  - Schwarz criterion, 396
  - seasonal ARIMA (SARIMA), 407
  - seasonal auto-regressive (SAR), 407
  - seasonal effect, 377
  - seasonality, 379
    - evaluation, 405
  - seasonally adjusted, 394
  - seasonal moving average SMA, 407
  - second normal form, 86
  - SELECT
    - DISTINCT, 444
    - MDX, 489
  - SELECT, FROM, JOIN, WHERE, 52
  - sequence analysis
    - data, 442
    - missing data, 444
  - Sequence Cluster, 456
  - sequence clustering, 447
  - sequence mining, 436
  - sequences, 435
    - classification, 439
    - clustering, 452
  - set, 489
  - Shannon, Claude, 205, 351
  - Shannon's entropy, 351
  - Simpson, E.H., 285
  - Simpson's paradox, 285
  - skewed support, 286
  - slice, 492
  - slope coefficient, 315
  - Smith, Adam, 11
  - smooth data, 384
  - snowflake design, 107
  - software tools, 20
  - Solution Explorer, 321
  - spurious correlation, 287
  - SQL, 17, 42, 101, 487
    - aggregation, 58
    - AND, 54
    - BETWEEN, 56
    - CASE, 316, 332
    - CASE statement, 113
    - DELETE, 74
    - DESC, 52
    - DISTINCT, 58
    - function, 60
    - GROUP BY, 61, 63
    - HAVING, 63
    - INNER JOIN, 67
    - INSERT, 73
    - introduction, 52
    - JOIN, 66
    - LIKE, 56
    - NOT, 54
    - NULL, 57
    - OR, 55
    - ORDER BY, 52
    - ROLLBACK, 74
    - SELECT, 52
    - subquery, 70
    - UNION, 71
    - UPDATE, 72
    - view, 69
    - WHERE, 58
  - SQL function
    - CAST, 312
    - Month, 397
    - Year, 319, 397
  - SQL query, 299
  - SQL Server, 30, 43, 61
    - Business Intelligence (BI), 75
    - reports, 75
  - SQL Server Analysis Services (SSAS), 20, 21, 99
    - OLAP cube, 108
  - SQL Server Business Intelligence (BI), 75
  - SQL Server Management Studio, 399, 490
  - squared-difference, 222
  - SSAS
    - data sources, 109
    - data source view, 111
    - deployment and processing, 118
  - stability
    - model, 327, 368
  - standard deviation, 185, 191
    - mean, 199
  - star design, 105
  - state transitions, 456
  - stationary, 392
  - statistical mixture model, 227
  - statistical research, 3
  - statistical theory, 3
  - statistical tools, 22
    - gretl, 22
    - SAS, 22

SPSS, 22  
Stata, 22  
statistician, 25  
statistics, 15, 161, 195  
Status expression, 152  
strings, 437  
subjective, 162  
subquery, 70  
subtotal, 61, 119  
sum of squared errors, 316  
supply chain management, 11  
support, 277, 279  
    minimum level, 283  
surprise, 206  
swindle, 5

**T**

table, 43, 48  
    join, 66  
Tableau Software, 481  
table join, 48  
Taiwan Semiconductor Manufacturing (TSMC), 11  
Taylor III, alex, 38  
T distribution, 192  
third normal form, 87  
Tibshirani, Robert, 38  
TIGER mapping system, 461  
time dimension, 124  
time key, 408  
time series, 377  
    data, 396  
    missing data, 397  
    model, 379  
TOP 5, 65  
TopCount, 504  
TopPercent, 504  
TopSum, 504  
transactions, 4  
transition probability, 439  
translation, 140  
tree diagram, 171  
trend, 379, 387, 399  
    nonlinear, 387  
Trend Expression, 152  
T statistic, 323  
tuple, 489  
Type I error, 201  
Type II error, 201

**U**

Unicode, 45  
uniform distribution, 188, 206  
UNION, 71  
union of events, 166  
Union Pacific, 307  
University of Waikato, 252  
unsupervised learning, 20,  
    219, 270  
UPDATE, 72  
U.S. Geological Survey  
    (USGS), 473

**V**

variance, 184, 185, 328  
    sample, 196  
Venn diagram, 279  
view, 69, 99  
Vincent P., 37  
visualization, 254, 481  
Visual Studio, 21, 75, 109, 490  
Visual Studio Designer, 508

**W**

Wallman, M., 38  
Wal-Mart, 2, 32  
Washington Mutual, 5  
Web logs, 445  
Web site traffic, 447  
Web site usage, 436  
weighted average, 385  
Weka, 22, 238  
WHEN, 153  
WHERE, 58, 64  
    MDX, 489, 492  
    versus HAVING, 64  
Winmetrics Corp., 95  
wisdom, 4  
within-cluster distance, 225  
WITH MEMBER, 489  
wizard  
    report, 80

**X**

XML, 45  
XQuery, 46

**Y**

YTD, 506

**Z**

Zaltman, Gerald, 37  
Zimmerman, Ann, 37  
Zoran, 11  
Z statistic, 203  
Z value, 198  
Zweig, Jason, 38