Microsoft

EXAM 70-463

Implementing a Data Warehouse with Microsoft® SQL Server® 2012

Dejan Sarka
Matija Lah
Grega Jerkič

Training Kit

How to access your CD files

The print edition of this book includes a CD. To access the CD files, go to http://aka.ms/666092/files, and look for the Downloads tab.

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

Questions? Please contact: mspinput@microsoft.com

Microsoft Press

Exam 70-463: Implementing a Data Warehouse with Microsoft SQL Server 2012

OBJECTIVE	CHAPTER	LESSON
1. DESIGN AND IMPLEMENT A DATA WAREHOUSE		
1.1 Design and implement dimensions.	Chapter 1	Lessons 1 and, 2
	Chapter 2	Lessons 1, 2, and 3
1.2 Design and implement fact tables.	Chapter 1	Lesson 3
	Chapter 2	Lessons 1, 2, and 3
2. EXTRACT AND TRANSFORM DATA		
2.1 Define connection managers.	Chapter 3	Lessons 1 and 3
	Chapter 4	Lesson 1
	Chapter 9	Lesson 2
2.2 Design data flow.	Chapter 3	Lesson 1
	Chapter 5	Lessons 1, 2, and 3
	Chapter 7	Lesson 1
	Chapter 10	Lesson 2
	Chapter 13	Lesson 2
	Chapter 18	Lessons 1, 2, and 3
	Chapter 19	Lesson 2
	Chapter 20	Lesson 1
2.3 Implement data flow.	Chapter 3	Lesson 1
	Chapter 5	Lessons 1, 2, and 3
	Chapter 7	Lessons 1 and 3
	Chapter 13	Lesson 1 and 2
	Chapter 18	Lesson 1
	Chapter 20	Lessons 2 and 3
2.4 Manage SSIS package execution.	Chapter 8	Lessons 1 and 2
	Chapter 12	Lesson 1
2.5 Implement script tasks in SSIS.	Chapter 19	Lesson 1
3. LOAD DATA		
3.1 Design control flow.	Chapter 3	Lessons 2 and 3
	Chapter 4	Lessons 2 and 3
	Chapter 6	Lessons 1 and 3
	Chapter 8	Lessons 1, 2, and 3
	Chapter 10	Lesson 1
	Chapter 12	Lesson 1 and 2
	Chapter 19	Lesson 1
3.2 Implement package logic by using SSIS variables and parameters.	Chapter 6	Lessons 1 and 2
	Chapter 9	Lessons 1 and 2
3.3 Implement control flow.	Chapter 4	Lessons 2 and 3
	Chapter 6	Lesson 3
	Chapter 8	Lessons 1 and 2
	Chapter 10	Lesson 3
	Chapter 13	Lessons 1, 2, and 3
3.4 Implement data load options.	Chapter 7	Lesson 2
3.5 Implement script components in SSIS.	Chapter 19	Lesson 2

OBJECTIVE	CHAPTER	LESSON
4. CONFIGURE AND DEPLOY SSIS SOLUTIONS		
4.1 Troubleshoot data integration issues.	Chapter 10	Lesson 1
	Chapter 13	Lessons 1, 2, and 3
4.2 Install and maintain SSIS components.	Chapter 11	Lesson 1
4.3 Implement auditing, logging, and event handling.	Chapter 8	Lesson 3
	Chapter 10	Lessons 1 and 2
4.4 Deploy SSIS solutions.	Chapter 11	Lessons 1 and 2
	Chapter 19	Lesson 3
4.5 Configure SSIS security settings.	Chapter 12	Lessons 1 and 2
5. BUILD DATA QUALITY SOLUTIONS		
5.1 Install and maintain Data Quality Services.	Chapter 14	Lessons 1, 2, and 3
5.2 Implement master data management solutions.	Chapter 15	Lessons 1, 2, and 3
	Chapter 16	Lessons 1, 2, and 3
5.3 Create a data quality project to clean data.	Chapter 14	Lesson 1
	Chapter 17	Lessons 1, 2, and 3
	Chapter 20	Lessons 1 and 2

Exam Objectives The exam objectives listed here are current as of this book's publication date. Exam objectives are subject to change at any time without prior notice and at Microsoft's sole discretion. Please visit the Microsoft Learning website for the most current listing of exam objectives: *http://www.microsoft.com/learning/en/us/exam.aspx?ID=70-463&locale=en-us*.

Microsoft

Exam 70-463: Implementing a Data Warehouse with Microsoft® SQL Server® 2012

Training Kit

Dejan Sarka
Matija Lah
Grega Jerkič

Copyright © 2012 by SolidQuality Europe GmbH

All rights reserved. No part of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

ISBN: 978-0-7356-6609-2

Fifth Printing: October 2014

Printed and bound in the United States of America.

Microsoft Press books are available through booksellers and distributors worldwide. If you need support related to this book, email Microsoft Press Book Support at *mspinput@microsoft.com*. Please tell us what you think of this book at *http://www.microsoft.com/learning/booksurvey*.

Microsoft and the trademarks listed at *http://www.microsoft.com/about/legal/en/us/IntellectualProperty/Trademarks/EN-US.aspx* are trademarks of the Microsoft group of companies. All other marks are property of their respective owners.

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

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

Acquisitions and Developmental Editor: Russell Jones

Production Editor: Holly Bauer

Editorial Production: Online Training Solutions, Inc.

Technical Reviewer: Miloš Radivojević

Copyeditor: Kathy Krause, Online Training Solutions, Inc.

Indexer: Ginny Munroe, Judith McConville

Cover Design: Twist Creative • Seattle

Cover Composition: Zyg Group, LLC

Illustrator: Jeanne Craver, Online Training Solutions, Inc.

Contents at a Glance

	Introduction	*xxvii*
PART I	**DESIGNING AND IMPLEMENTING A DATA WAREHOUSE**	
CHAPTER 1	Data Warehouse Logical Design	3
CHAPTER 2	Implementing a Data Warehouse	41
PART II	**DEVELOPING SSIS PACKAGES**	
CHAPTER 3	Creating SSIS Packages	87
CHAPTER 4	Designing and Implementing Control Flow	131
CHAPTER 5	Designing and Implementing Data Flow	177
PART III	**ENHANCING SSIS PACKAGES**	
CHAPTER 6	Enhancing Control Flow	239
CHAPTER 7	Enhancing Data Flow	283
CHAPTER 8	Creating a Robust and Restartable Package	327
CHAPTER 9	Implementing Dynamic Packages	353
CHAPTER 10	Auditing and Logging	381
PART IV	**MANAGING AND MAINTAINING SSIS PACKAGES**	
CHAPTER 11	Installing SSIS and Deploying Packages	421
CHAPTER 12	Executing and Securing Packages	455
CHAPTER 13	Troubleshooting and Performance Tuning	497
PART V	**BUILDING DATA QUALITY SOLUTIONS**	
CHAPTER 14	Installing and Maintaining Data Quality Services	529
CHAPTER 15	Implementing Master Data Services	565
CHAPTER 16	Managing Master Data	605
CHAPTER 17	Creating a Data Quality Project to Clean Data	637

PART VI ADVANCED SSIS AND DATA QUALITY TOPICS

CHAPTER 18 SSIS and Data Mining 667

CHAPTER 19 Implementing Custom Code in SSIS Packages 699

CHAPTER 20 Identity Mapping and De-Duplicating 735

Index *769*

Contents

Introduction **xxvii**

System Requirements *xxviii*

Using the Companion CD *xxix*

Acknowledgments *xxxi*

Support & Feedback *xxxi*

Preparing for the Exam *xxxiii*

PART I DESIGNING AND IMPLEMENTING A DATA WAREHOUSE

Chapter 1 Data Warehouse Logical Design 3

Before You Begin. 4

Lesson 1: Introducing Star and Snowflake Schemas. 4

- Reporting Problems with a Normalized Schema 5
- Star Schema 7
- Snowflake Schema 9
- Granularity Level 12
- Auditing and Lineage 13
- Lesson Summary 16
- Lesson Review 16

Lesson 2: Designing Dimensions 17

- Dimension Column Types 17
- Hierarchies 19
- Slowly Changing Dimensions 21
- Lesson Summary 26
- Lesson Review 26

What do you think of this book? We want to hear from you!
Microsoft is interested in hearing your feedback so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit:

www.microsoft.com/learning/booksurvey/

Lesson 3: Designing Fact Tables 27
Fact Table Column Types 28
Additivity of Measures 29
Additivity of Measures in SSAS 30
Many-to-Many Relationships 30
Lesson Summary 33
Lesson Review 34

Case Scenarios 34
Case Scenario 1: A Quick POC Project 34
Case Scenario 2: Extending the POC Project 35

Suggested Practices 35
Analyze the AdventureWorksDW2012 Database Thoroughly 35
Check the SCD and Lineage in the AdventureWorks-DW2012 Database 36

Answers 37
Lesson 1 37
Lesson 2 37
Lesson 3 38
Case Scenario 1 39
Case Scenario 2 39

Chapter 2 Implementing a Data Warehouse 41

Before You Begin 42

Lesson 1: Implementing Dimensions and Fact Tables 42
Creating a Data Warehouse Database 42
Implementing Dimensions 45
Implementing Fact Tables 47
Lesson Summary 54
Lesson Review 54

Lesson 2: Managing the Performance of a Data Warehouse 55
Indexing Dimensions and Fact Tables 56
Indexed Views 58
Data Compression 61
Columnstore Indexes and Batch Processing 62

Lesson Summary 69
Lesson Review 70

Lesson 3: Loading and Auditing Loads 70
Using Partitions 71
Data Lineage 73
Lesson Summary 78
Lesson Review 78

Case Scenarios 78
Case Scenario 1: Slow DW Reports 79
Case Scenario 2: DW Administration Problems 79

Suggested Practices 79
Test Different Indexing Methods 79
Test Table Partitioning 80

Answers 81
Lesson 1 81
Lesson 2 81
Lesson 3 82
Case Scenario 1 83
Case Scenario 2 83

PART II DEVELOPING SSIS PACKAGES

Chapter 3 Creating SSIS Packages 87

Before You Begin 89

Lesson 1: Using the SQL Server Import and Export Wizard 89
Planning a Simple Data Movement 89
Lesson Summary 99
Lesson Review 99

Lesson 2: Developing SSIS Packages in SSDT 101
Introducing SSDT 102
Lesson Summary 107
Lesson Review 108

Lesson 3: Introducing Control Flow, Data Flow, and Connection Managers 109

Introducing SSIS Development 110
Introducing SSIS Project Deployment 110
Lesson Summary 124
Lesson Review 124

Case Scenarios 125
Case Scenario 1: Copying Production Data to Development 125
Case Scenario 2: Connection Manager Parameterization 125

Suggested Practices 125
Use the Right Tool 125
Account for the Differences Between Development and Production Environments 126

Answers. 127
Lesson 1 127
Lesson 2 128
Lesson 3 128
Case Scenario 1 129
Case Scenario 2 129

Chapter 4 Designing and Implementing Control Flow 131

Before You Begin. 132

Lesson 1: Connection Managers 133
Lesson Summary 144
Lesson Review 144

Lesson 2: Control Flow Tasks and Containers 145
Planning a Complex Data Movement 145
Tasks 147
Containers 155
Lesson Summary 163
Lesson Review 163

Lesson 3: Precedence Constraints 164
Lesson Summary 169
Lesson Review 169

Case Scenarios 170
Case Scenario 1: Creating a Cleanup Process 170
Case Scenario 2: Integrating External Processes 171

Suggested Practices 171
A Complete Data Movement Solution 171

Answers. 173
Lesson 1 173
Lesson 2 174
Lesson 3 175
Case Scenario 1 176
Case Scenario 2 176

Chapter 5 Designing and Implementing Data Flow 177

Before You Begin. 177

Lesson 1: Defining Data Sources and Destinations. 178
Creating a Data Flow Task 178
Defining Data Flow Source Adapters 180
Defining Data Flow Destination Adapters 184
SSIS Data Types 187
Lesson Summary 197
Lesson Review 197

Lesson 2: Working with Data Flow Transformations. 198
Selecting Transformations 198
Using Transformations 205
Lesson Summary 215
Lesson Review 215

Lesson 3: Determining Appropriate ETL Strategy and Tools 216
ETL Strategy 217
Lookup Transformations 218
Sorting the Data 224
Set-Based Updates 225
Lesson Summary 231
Lesson Review 231

Case Scenario. 232

Case Scenario: New Source System 232

Suggested Practices 233

Create and Load Additional Tables 233

Answers. 234

Lesson 1 234

Lesson 2 234

Lesson 3 235

Case Scenario 236

PART III ENHANCING SSIS PACKAGES

Chapter 6 Enhancing Control Flow 239

Before You Begin. 241

Lesson 1: SSIS Variables 241

System and User Variables 243

Variable Data Types 245

Variable Scope 248

Property Parameterization 251

Lesson Summary 253

Lesson Review 253

Lesson 2: Connection Managers, Tasks, and Precedence Constraint Expressions 254

Expressions 255

Property Expressions 259

Precedence Constraint Expressions 259

Lesson Summary 263

Lesson Review 264

Lesson 3: Using a Master Package for Advanced Control Flow 265

Separating Workloads, Purposes, and Objectives 267

Harmonizing Workflow and Configuration 268

The Execute Package Task 269

The Execute SQL Server Agent Job Task 269

The Execute Process Task 270

Lesson Summary 275
Lesson Review 275

Case Scenarios 276
Case Scenario 1: Complete Solutions 276
Case Scenario 2: Data-Driven Execution 277

Suggested Practices 277
Consider Using a Master Package 277

Answers. 278
Lesson 1 278
Lesson 2 279
Lesson 3 279
Case Scenario 1 280
Case Scenario 2 281

Chapter 7 Enhancing Data Flow 283

Before You Begin. 283

Lesson 1: Slowly Changing Dimensions 284
Defining Attribute Types 284
Inferred Dimension Members 285
Using the Slowly Changing Dimension Task 285
Effectively Updating Dimensions 290
Lesson Summary 298
Lesson Review 298

Lesson 2: Preparing a Package for Incremental Load. 299
Using Dynamic SQL to Read Data 299
Implementing CDC by Using SSIS 304
ETL Strategy for Incrementally Loading Fact Tables 307
Lesson Summary 316
Lesson Review 316

Lesson 3: Error Flow 317
Using Error Flows 317
Lesson Summary 321
Lesson Review 321

Case Scenario. 322
Case Scenario: Loading Large Dimension and Fact Tables 322
Suggested Practices 322
Load Additional Dimensions 322
Answers. 323
Lesson 1 323
Lesson 2 324
Lesson 3 324
Case Scenario 325

Chapter 8 Creating a Robust and Restartable Package 327

Before You Begin. 328
Lesson 1: Package Transactions 328
Defining Package and Task Transaction Settings 328
Transaction Isolation Levels 331
Manually Handling Transactions 332
Lesson Summary 335
Lesson Review 335
Lesson 2: Checkpoints 336
Implementing Restartability Checkpoints 336
Lesson Summary 341
Lesson Review 341
Lesson 3: Event Handlers 342
Using Event Handlers 342
Lesson Summary 346
Lesson Review 346
Case Scenario. 347
Case Scenario: Auditing and Notifications in SSIS Packages 347
Suggested Practices 348
Use Transactions and Event Handlers 348
Answers. 349
Lesson 1 349
Lesson 2 349

Lesson 3 350

Case Scenario 351

Chapter 9 Implementing Dynamic Packages 353

Before You Begin. 354

Lesson 1: Package-Level and Project-Level Connection Managers and Parameters. 354

Using Project-Level Connection Managers 355

Parameters 356

Build Configurations in SQL Server 2012 Integration Services 358

Property Expressions 361

Lesson Summary 366

Lesson Review 366

Lesson 2: Package Configurations 367

Implementing Package Configurations 368

Lesson Summary 377

Lesson Review 377

Case Scenario. 378

Case Scenario: Making SSIS Packages Dynamic 378

Suggested Practices 378

Use a Parameter to Incrementally Load a Fact Table 378

Answers. 379

Lesson 1 379

Lesson 2 379

Case Scenario 380

Chapter 10 Auditing and Logging 381

Before You Begin. 383

Lesson 1: Logging Packages 383

Log Providers 383

Configuring Logging 386

Lesson Summary 393

Lesson Review 394

Lesson 2: Implementing Auditing and Lineage 394

Auditing Techniques 395

Correlating Audit Data with SSIS Logs 401

Retention 401

Lesson Summary 405

Lesson Review 405

Lesson 3: Preparing Package Templates 406

SSIS Package Templates 407

Lesson Summary 410

Lesson Review 410

Case Scenarios 411

Case Scenario 1: Implementing SSIS Logging at Multiple Levels of the SSIS Object Hierarchy 411

Case Scenario 2: Implementing SSIS Auditing at Different Levels of the SSIS Object Hierarchy 412

Suggested Practices 412

Add Auditing to an Update Operation in an Existing Execute SQL Task 412

Create an SSIS Package Template in Your Own Environment 413

Answers. 414

Lesson 1 414

Lesson 2 415

Lesson 3 416

Case Scenario 1 417

Case Scenario 2 417

PART IV MANAGING AND MAINTAINING SSIS PACKAGES

Chapter 11 Installing SSIS and Deploying Packages 421

Before You Begin. 422

Lesson 1: Installing SSIS Components 423

Preparing an SSIS Installation 424

Installing SSIS 428

Lesson Summary 436

Lesson Review 436

Lesson 2: Deploying SSIS Packages 437

SSISDB Catalog 438

SSISDB Objects 440

Project Deployment 442

Lesson Summary 449

Lesson Review 450

Case Scenarios 450

Case Scenario 1: Using Strictly Structured Deployments 451

Case Scenario 2: Installing an SSIS Server 451

Suggested Practices 451

Upgrade Existing SSIS Solutions 451

Answers. 452

Lesson 1 452

Lesson 2 453

Case Scenario 1 454

Case Scenario 2 454

Chapter 12 Executing and Securing Packages 455

Before You Begin. 456

Lesson 1: Executing SSIS Packages. 456

On-Demand SSIS Execution 457

Automated SSIS Execution 462

Monitoring SSIS Execution 465

Lesson Summary 479

Lesson Review 479

Lesson 2: Securing SSIS Packages. 480

SSISDB Security 481

Lesson Summary 490

Lesson Review 490

Case Scenarios 491

Case Scenario 1: Deploying SSIS Packages to Multiple Environments 491

Case Scenario 2: Remote Executions 491

Suggested Practices 491

Improve the Reusability of an SSIS Solution 492

Answers. 493

Lesson 1 493

Lesson 2 494

Case Scenario 1 495

Case Scenario 2 495

Chapter 13 Troubleshooting and Performance Tuning 497

Before You Begin. 498

Lesson 1: Troubleshooting Package Execution 498

Design-Time Troubleshooting 498

Production-Time Troubleshooting 506

Lesson Summary 510

Lesson Review 510

Lesson 2: Performance Tuning 511

SSIS Data Flow Engine 512

Data Flow Tuning Options 514

Parallel Execution in SSIS 517

Troubleshooting and Benchmarking Performance 518

Lesson Summary 522

Lesson Review 522

Case Scenario. 523

Case Scenario: Tuning an SSIS Package 523

Suggested Practice 524

Get Familiar with SSISDB Catalog Views 524

Answers. 525

Lesson 1 525

Lesson 2 525

Case Scenario 526

PART V BUILDING DATA QUALITY SOLUTIONS

Chapter 14 Installing and Maintaining Data Quality Services 529

Before You Begin. 530

Lesson 1: Data Quality Problems and Roles. 530

Data Quality Dimensions 531

Data Quality Activities and Roles 535

Lesson Summary 539

Lesson Review 539

Lesson 2: Installing Data Quality Services. 540

DQS Architecture 540

DQS Installation 542

Lesson Summary 548

Lesson Review 548

Lesson 3: Maintaining and Securing Data Quality Services. 549

Performing Administrative Activities with Data Quality Client 549

Performing Administrative Activities with Other Tools 553

Lesson Summary 558

Lesson Review 558

Case Scenario. 559

Case Scenario: Data Warehouse Not Used 559

Suggested Practices 560

Analyze the AdventureWorksDW2012 Database 560

Review Data Profiling Tools 560

Answers. 561

Lesson 1 561

Lesson 2 561

Lesson 3 562

Case Scenario 563

Chapter 15 Implementing Master Data Services 565

Before You Begin. 565

Lesson 1: Defining Master Data 566

What Is Master Data? 567

Master Data Management 569

MDM Challenges 572

Lesson Summary 574

Lesson Review 574

Lesson 2: Installing Master Data Services. 575

Master Data Services Architecture 576

MDS Installation 577

Lesson Summary 587

Lesson Review 587

Lesson 3: Creating a Master Data Services Model .588

MDS Models and Objects in Models 588

MDS Objects 589

Lesson Summary 599

Lesson Review 600

Case Scenarios. 600

Case Scenario 1: Introducing an MDM Solution 600

Case Scenario 2: Extending the POC Project 601

Suggested Practices 601

Analyze the AdventureWorks2012 Database 601

Expand the MDS Model 601

Answers. 602

Lesson 1 602

Lesson 2 603

Lesson 3 603

Case Scenario 1 604

Case Scenario 2 604

Chapter 16 Managing Master Data 605

Before You Begin. 605

Lesson 1: Importing and Exporting Master Data .606

Creating and Deploying MDS Packages 606

Importing Batches of Data 607

Exporting Data 609

Lesson Summary 615

Lesson Review 616

Lesson 2: Defining Master Data Security 616

Users and Permissions 617

Overlapping Permissions 619

Lesson Summary 624

Lesson Review 624

Lesson 3: Using Master Data Services Add-in for Excel 624

Editing MDS Data in Excel 625

Creating MDS Objects in Excel 627

Lesson Summary 632

Lesson Review 632

Case Scenario. 633

Case Scenario: Editing Batches of MDS Data 633

Suggested Practices 633

Analyze the Staging Tables 633

Test Security 633

Answers. 634

Lesson 1 634

Lesson 2 635

Lesson 3 635

Case Scenario 636

Chapter 17 Creating a Data Quality Project to Clean Data 637

Before You Begin. 637

Lesson 1: Creating and Maintaining a Knowledge Base 638

Building a DQS Knowledge Base 638

Domain Management 639

Lesson Summary 645

Lesson Review 645

Lesson 2: Creating a Data Quality Project 646

DQS Projects 646

Data Cleansing 647

Lesson Summary 653

Lesson Review 653

Lesson 3: Profiling Data and Improving Data Quality 654

Using Queries to Profile Data 654

SSIS Data Profiling Task 656

Lesson Summary 659

Lesson Review 660

Case Scenario. 660

Case Scenario: Improving Data Quality 660

Suggested Practices 661

Create an Additional Knowledge Base and Project 661

Answers. 662

Lesson 1 662

Lesson 2 662

Lesson 3 663

Case Scenario 664

PART VI ADVANCED SSIS AND DATA QUALITY TOPICS

Chapter 18 SSIS and Data Mining 667

Before You Begin. 667

Lesson 1: Data Mining Task and Transformation. .668

What Is Data Mining? 668

SSAS Data Mining Algorithms 670

Using Data Mining Predictions in SSIS 671
Lesson Summary 679
Lesson Review 679

Lesson 2: Text Mining 679
Term Extraction 680
Term Lookup 681
Lesson Summary 686
Lesson Review 686

Lesson 3: Preparing Data for Data Mining 687
Preparing the Data 688
SSIS Sampling 689
Lesson Summary 693
Lesson Review 693

Case Scenario 694
Case Scenario: Preparing Data for Data Mining 694

Suggested Practices 694
Test the Row Sampling and Conditional Split Transformations 694

Answers 695
Lesson 1 695
Lesson 2 695
Lesson 3 696
Case Scenario 697

Chapter 19 Implementing Custom Code in SSIS Packages 699

Before You Begin 700

Lesson 1: Script Task 700
Configuring the Script Task 701
Coding the Script Task 702
Lesson Summary 707
Lesson Review 707

Lesson 2: Script Component 707
Configuring the Script Component 708
Coding the Script Component 709

Lesson Summary 715
Lesson Review 715

Lesson 3: Implementing Custom Components 716
Planning a Custom Component 717
Developing a Custom Component 718
Design Time and Run Time 719
Design-Time Methods 719
Run-Time Methods 721
Lesson Summary 730
Lesson Review 730

Case Scenario. 731
Case Scenario: Data Cleansing 731

Suggested Practices 731
Create a Web Service Source 731

Answers. 732
Lesson 1 732
Lesson 2 732
Lesson 3 733
Case Scenario 734

Chapter 20 Identity Mapping and De-Duplicating 735

Before You Begin. 736

Lesson 1: Understanding the Problem 736
Identity Mapping and De-Duplicating Problems 736
Solving the Problems 738
Lesson Summary 744
Lesson Review 744

Lesson 2: Using DQS and the DQS Cleansing Transformation 745
DQS Cleansing Transformation 746
DQS Matching 746
Lesson Summary 755
Lesson Review 755

Lesson 3: Implementing SSIS Fuzzy Transformations 756

Fuzzy Transformations Algorithm 756

Versions of Fuzzy Transformations 758

Lesson Summary 764

Lesson Review 764

Case Scenario. 765

Case Scenario: Improving Data Quality 765

Suggested Practices 765

Research More on Matching 765

Answers. 766

Lesson 1 766

Lesson 2 766

Lesson 3 767

Case Scenario 768

Index *769*

Introduction

This Training Kit is designed for information technology (IT) professionals who support or plan to support data warehouses, extract-transform-load (ETL) processes, data quality improvements, and master data management. It is designed for IT professionals who also plan to take the Microsoft Certified Technology Specialist (MCTS) exam 70-463. The authors assume that you have a solid, foundation-level understanding of Microsoft SQL Server 2012 and the Transact-SQL language, and that you understand basic relational modeling concepts.

The material covered in this Training Kit and on Exam 70-463 relates to the technologies provided by SQL Server 2012 for implementing and maintaining a data warehouse. The topics in this Training Kit cover what you need to know for the exam as described on the Skills Measured tab for the exam, available at:

http://www.microsoft.com/learning/en/us/exam.aspx?id=70-463

By studying this Training Kit, you will see how to perform the following tasks:

- Design an appropriate data model for a data warehouse
- Optimize the physical design of a data warehouse
- Extract data from different data sources, transform and cleanse the data, and load it in your data warehouse by using SQL Server Integration Services (SSIS)
- Use advanced SSIS components
- Use SQL Server 2012 Master Data Services (MDS) to take control of your master data
- Use SQL Server Data Quality Services (DQS) for data cleansing

Refer to the objective mapping page in the front of this book to see where in the book each exam objective is covered.

System Requirements

The following are the minimum system requirements for the computer you will be using to complete the practice exercises in this book and to run the companion CD.

SQL Server and Other Software Requirements

This section contains the minimum SQL Server and other software requirements you will need:

- **SQL Server 2012** You need access to a SQL Server 2012 instance with a logon that has permissions to create new databases—preferably one that is a member of the sysadmin role. For the purposes of this Training Kit, you can use almost any edition of on-premises SQL Server (Standard, Enterprise, Business Intelligence, and Developer), both 32-bit and 64-bit editions. If you don't have access to an existing SQL Server instance, you can install a trial copy of SQL Server 2012 that you can use for 180 days. You can download a trial copy here:

http://www.microsoft.com/sqlserver/en/us/get-sql-server/try-it.aspx

- **SQL Server 2012 Setup Feature Selection** When you are in the Feature Selection dialog box of the SQL Server 2012 setup program, choose at minimum the following components:
  - Database Engine Services
  - Documentation Components
  - Management Tools - Basic
  - Management Tools – Complete
  - SQL Server Data Tools
- **Windows Software Development Kit (SDK) or Microsoft Visual Studio 2010** The Windows SDK provides tools, compilers, headers, libraries, code samples, and a new help system that you can use to create applications that run on Windows. You need the Windows SDK for Chapter 19, "Implementing Custom Code in SSIS Packages" only. If you already have Visual Studio 2010, you do not need the Windows SDK. If you need the Windows SDK, you need to download the appropriate version for your operating system. For Windows 7, Windows Server 2003 R2 Standard Edition (32-bit x86), Windows Server 2003 R2 Standard x64 Edition, Windows Server 2008, Windows Server 2008 R2, Windows Vista, or Windows XP Service Pack 3, use the Microsoft Windows SDK for Windows 7 and the Microsoft .NET Framework 4 from:

  http://www.microsoft.com/en-us/download/details.aspx?id=8279

Hardware and Operating System Requirements

You can find the minimum hardware and operating system requirements for SQL Server 2012 here:

http://msdn.microsoft.com/en-us/library/ms143506(v=sql.110).aspx

Data Requirements

The minimum data requirements for the exercises in this Training Kit are the following:

- **The AdventureWorks OLTP and DW databases for SQL Server 2012** Exercises in this book use the AdventureWorks online transactional processing (OLTP) database, which supports standard online transaction processing scenarios for a fictitious bicycle manufacturer (Adventure Works Cycles), and the AdventureWorks data warehouse (DW) database, which demonstrates how to build a data warehouse. You need to download both databases for SQL Server 2012. You can download both databases from:

http://msftdbprodsamples.codeplex.com/releases/view/55330

You can also download the compressed file containing the data (.mdf) files for both databases from MS Press website here:

http://www.microsoftpressstore.com/title/9780735666092.

Using the Companion CD

A companion CD is included with this Training Kit. The companion CD contains the following:

- **Practice tests** You can reinforce your understanding of the topics covered in this Training Kit by using electronic practice tests that you customize to meet your needs. You can practice for the 70-463 certification exam by using tests created from a pool of over 200 realistic exam questions, which give you many practice exams to ensure that you are prepared.
- **An eBook** An electronic version (eBook) of this book is included for when you do not want to carry the printed book with you.
- **Source code** A compressed file called TK70463_CodeLabSolutions.zip includes the Training Kit's demo source code and exercise solutions. You can also download the compressed file from website here:

  http://www.microsoftpressstore.com/title/9780735666092.

  For convenient access to the source code, create a local folder called **C:\TK463\** and extract the compressed archive by using this folder as the destination for the extracted files.
- **Sample data** A compressed file called AdventureWorksDataFiles.zip includes the Training Kit's demo source code and exercise solutions. You can also download the compressed file from website here:

  http://www.microsoftpressstore.com/title/9780735666092.

  For convenient access to the source code, create a local folder called **C:\TK463\** and extract the compressed archive by using this folder as the destination for the extracted files. Then use SQL Server Management Studio (SSMS) to attach both databases and create the log files for them.

How to Install the Practice Tests

To install the practice test software from the companion CD to your hard disk, perform the following steps:

1. Insert the companion CD into your CD drive and accept the license agreement. A CD menu appears.

> ***NOTE*** **IF THE CD MENU DOES NOT APPEAR**
>
> **If the CD menu or the license agreement does not appear, AutoRun might be disabled on your computer. Refer to the Readme.txt file on the CD for alternate installation instructions.**

2. Click Practice Tests and follow the instructions on the screen.

How to Use the Practice Tests

To start the practice test software, follow these steps:

1. Click Start | All Programs, and then select Microsoft Press Training Kit Exam Prep.

   A window appears that shows all the Microsoft Press Training Kit exam prep suites installed on your computer.

2. Double-click the practice test you want to use.

When you start a practice test, you choose whether to take the test in Certification Mode, Study Mode, or Custom Mode:

- **Certification Mode** Closely resembles the experience of taking a certification exam. The test has a set number of questions. It is timed, and you cannot pause and restart the timer.
- **Study Mode** Creates an untimed test during which you can review the correct answers and the explanations after you answer each question.
- **Custom Mode** Gives you full control over the test options so that you can customize them as you like.

In all modes, when you are taking the test, the user interface is basically the same but with different options enabled or disabled depending on the mode.

When you review your answer to an individual practice test question, a "References" section is provided that lists where in the Training Kit you can find the information that relates to that question and provides links to other sources of information. After you click Test Results to score your entire practice test, you can click the Learning Plan tab to see a list of references for every objective.

How to Uninstall the Practice Tests

To uninstall the practice test software for a Training Kit, use the Program And Features option in Windows Control Panel.

Acknowledgments

A book is put together by many more people than the authors whose names are listed on the title page. We'd like to express our gratitude to the following people for all the work they have done in getting this book into your hands: Miloš Radivojević (technical editor) and Fritz Lechnitz (project manager) from SolidQ, Russell Jones (acquisitions and developmental editor) and Holly Bauer (production editor) from Kathy Krause (copyeditor) and Jaime Odell (proofreader) from OTSI. In addition, we would like to give thanks to Matt Masson (member of the SSIS team), Wee Hyong Tok (SSIS team program manager), and Elad Ziklik (DQS group program manager) from Microsoft for the technical support and for unveiling the secrets of the new SQL Server 2012 products. There are many more people involved in writing and editing practice test questions, editing graphics, and performing other activities; we are grateful to all of them as well.

Support & Feedback

The following sections provide information on errata, book support, feedback, and contact information.

Errata

We've made every effort to ensure the accuracy of this book and its companion content. Any errors that have been reported since this book was published are listed on our Microsoft Press site:

http://www.microsoftpressstore.com/title/9780735666092.

If you find an error that is not already listed, you can report it to us through the same page.

If you need additional support, email Microsoft Press Book Support at:

mspinput@microsoft.com

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

We Want to Hear from You

At Microsoft Press, your satisfaction is our top priority, and your feedback our most valuable asset. Please tell us what you think of this book at:

http://www.microsoft.com/learning/booksurvey

The survey is short, and we read every one of your comments and ideas. Thanks in advance for your input!

Stay in Touch

Let's keep the conversation going! We are on Twitter: *http://twitter.com/MicrosoftPress*.

Preparing for the Exam

Microsoft certification exams are a great way to build your resume and let the world know about your level of expertise. Certification exams validate your on-the-job experience and product knowledge. While there is no substitution for on-the-job experience, preparation through study and hands-on practice can help you prepare for the exam. We recommend that you round out your exam preparation plan by using a combination of available study materials and courses. For example, you might use the training kit and another study guide for your "at home" preparation, and take a Microsoft Official Curriculum course for the classroom experience. Choose the combination that you think works best for you.

Note that this training kit is based on publicly available information about the exam and the authors' experience. To safeguard the integrity of the exam, authors do not have access to the live exam.

CHAPTER 1

Data Warehouse Logical Design

Exam objectives in this chapter:

- Design and Implement a Data Warehouse
  - Design and implement dimensions.
  - Design and implement fact tables.

Analyzing data from databases that support line-of-business (LOB) applications is usually not an easy task. The normalized relational schema used for an LOB application can consist of thousands of tables. Naming conventions are frequently not enforced. Therefore, it is hard to discover where the data you need for a report is stored. Enterprises frequently have multiple LOB applications, often working against more than one database. For the purposes of analysis, these enterprises need to be able to merge the data from multiple databases. Data quality is a common problem as well. In addition, many LOB applications do not track data over time, though many analyses depend on historical data.

IMPORTANT

Have you read page xxxii?

It contains valuable information regarding the skills you need to pass the exam.

A common solution to these problems is to create a *data warehouse (DW)*. A DW is a centralized data silo for an enterprise that contains merged, cleansed, and historical data. DW schemas are simplified and thus more suitable for generating reports than normalized relational schemas. For a DW, you typically use a special type of logical design called a Star schema, or a variant of the Star schema called a Snowflake schema. Tables in a Star or Snowflake schema are divided into dimension tables (commonly known as *dimensions*) and fact tables.

Data in a DW usually comes from LOB databases, but it's a transformed and cleansed copy of source data. Of course, there is some latency between the moment when data appears in an LOB database and the moment when it appears in a DW. One common method of addressing this latency involves refreshing the data in a DW as a nightly job. You use the refreshed data primarily for reports; therefore, the data is mostly read and rarely updated.

Queries often involve reading huge amounts of data and require large scans. To support such queries, it is imperative to use an appropriate physical design for a DW.

DW logical design seems to be simple at first glance. It is definitely much simpler than a normalized relational design. However, despite the simplicity, you can still encounter some advanced problems. In this chapter, you will learn how to design a DW and how to solve some of the common advanced design problems. You will explore Star and Snowflake schemas, dimensions, and fact tables. You will also learn how to track the source and time for data coming into a DW through auditing—or, in DW terminology, *lineage information*.

Lessons in this chapter:

- Lesson 1: Introducing Star and Snowflake Schemas
- Lesson 2: Designing Dimensions
- Lesson 3: Designing Fact Tables

Before You Begin

To complete this chapter, you must have:

- An understanding of normalized relational schemas.
- Experience working with Microsoft SQL Server 2012 Management Studio.
- A working knowledge of the Transact-SQL language.
- The AdventureWorks2012 and AdventureWorksDW2012 sample databases installed.

Lesson 1: Introducing Star and Snowflake Schemas

Before you design a data warehouse, you need to understand some common design patterns used for a DW, namely the Star and Snowflake schemas. These schemas evolved in the 1980s. In particular, the Star schema is currently so widely used that it has become a kind of informal standard for all types of business intelligence (BI) applications.

After this lesson, you will be able to:

- Understand why a normalized schema causes reporting problems.
- Understand the Star schema.
- Understand the Snowflake schema.
- Determine granularity and auditing needs.

Estimated lesson time: 40 minutes

Reporting Problems with a Normalized Schema

This lesson starts with normalized relational schema. Let's assume that you have to create a business report from a relational schema in the AdventureWorks2012 sample database. The report should include the sales amount for Internet sales in different countries over multiple years. The task (or even challenge) is to find out which tables and columns you would need to create the report. You start by investigating which tables store the data you need, as shown in Figure 1-1, which was created with the diagramming utility in SQL Server Management Studio (SSMS).

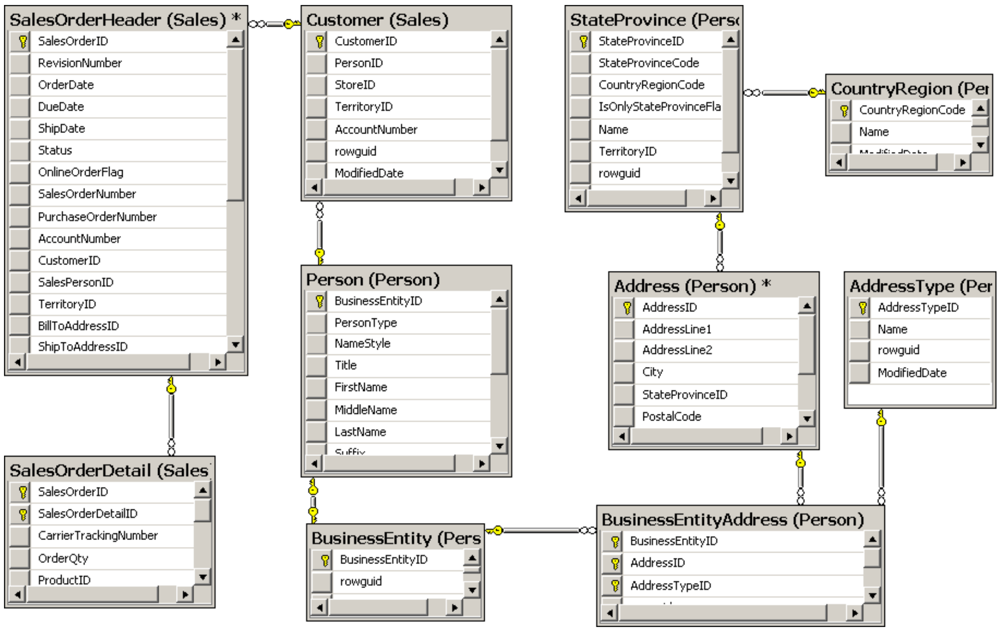


FIGURE 1-1 A diagram of tables you would need for a simple sales report.

Even for this relatively simple report, you would end up with 10 tables. You need the sales tables and the tables containing information about customers. The AdventureWorks2012 database schema is highly normalized; it's intended as an example schema to support LOB applications. Although such a schema works extremely well for LOB applications, it can cause problems when used as the source for reports, as you'll see in the rest of this section.

Normalization is a process in which you define entities in such a way that a single table represents exactly one entity. The goal is to have a complete and non-redundant schema. Every piece of information must be stored exactly once. This way, you can enforce data integrity. You have a place for every piece of data, and because each data item is stored only once, you do not have consistency problems. However, after a proper normalization, you typically wind up with many tables. In a database that supports an LOB application for an enterprise, you might finish with thousands of tables!

Finding the appropriate tables and columns you need for a report can be painful in a normalized database simply because of the number of tables involved. Add to this the fact that nothing forces database developers to maintain good naming conventions in an LOB database. It's relatively easy to find the pertinent tables in AdventureWorks2012, because the tables and columns have meaningful names. But imagine if the database contained tables named *Table1*, *Table2*, and so on, and columns named *Column1*, *Column2*, and so on. Finding the objects you need for your report would be a nightmare. Tools such as SQL Profiler might help. For example, you could create a test environment, try to insert some data through an LOB application, and have SQL Profiler identify where the data was inserted. A normalized schema is not very narrative. You cannot easily spot the storage location for data that measures something, such as the sales amount in this example, or the data that gives context to these measures, such as countries and years.

In addition, a query that joins 10 tables, as would be required in reporting sales by countries and years, would not be very fast. The query would also read huge amounts of data—sales over multiple years—and thus would interfere with the regular transactional work of inserting and updating the data.

Another problem in this example is the fact that there is no explicit lookup table for dates. You have to extract years from date or date/time columns in sales tables, such as *OrderDate* from the *SalesOrderHeader* table in this example. Extracting years from a date column is not such a big deal; however, the first question is, does the LOB database store data for multiple years? In many cases, LOB databases are purged after each new fiscal year starts. Even if you have all of the historical data for the sales transactions, you might have a problem showing the historical data correctly. For example, you might have only the latest customer address (from which you extract customer's current country), which might prevent you from calculating historical sales by country correctly.

The AdventureWorks2012 sample database stores all data in a single database. However, in an enterprise, you might have multiple LOB applications, each of which might store data in its own database. You might also have part of the sales data in one database and part in another. And you could have customer data in both databases, without a common identification. In such cases, you face the problems of how to merge all this data and how to identify which customer from one database is actually the same as a customer from another database.

Finally, data quality could be low. The old rule, "garbage in garbage out," applies to analyses as well. Parts of the data could be missing; other parts could be wrong. Even with good data, you could still have different representations of the same data in different databases. For example, gender in one database could be represented with the letters F and M, and in another database with the numbers 1 and 2.

The problems listed in this section are indicative of the problems that led designers to create different schemas for BI applications. The Star and Snowflake schemas are both simplified and narrative. A data warehouse should use Star and/or Snowflake designs. You'll also sometimes find the term *dimensional model* used for a DW schema. A dimensional model actually consists of both Star and Snowflake schemas. This is a good time to introduce the Star and Snowflake schemas.

Star Schema

Often, a picture is worth more than a thousand words. Figure 1-2 shows a *Star schema*, a diagram created in SSMS from a subset of the tables in the AdventureWorksDW2012 sample database.

In Figure 1-2, you can easily spot how the Star schema got its name—it resembles a star. There is a single central table, called a *fact table*, surrounded by multiple tables called *dimensions*. One Star schema covers one business area. In this case, the schema covers Internet sales. An enterprise data warehouse covers multiple business areas and consists of multiple Star (and/or Snowflake) schemas.

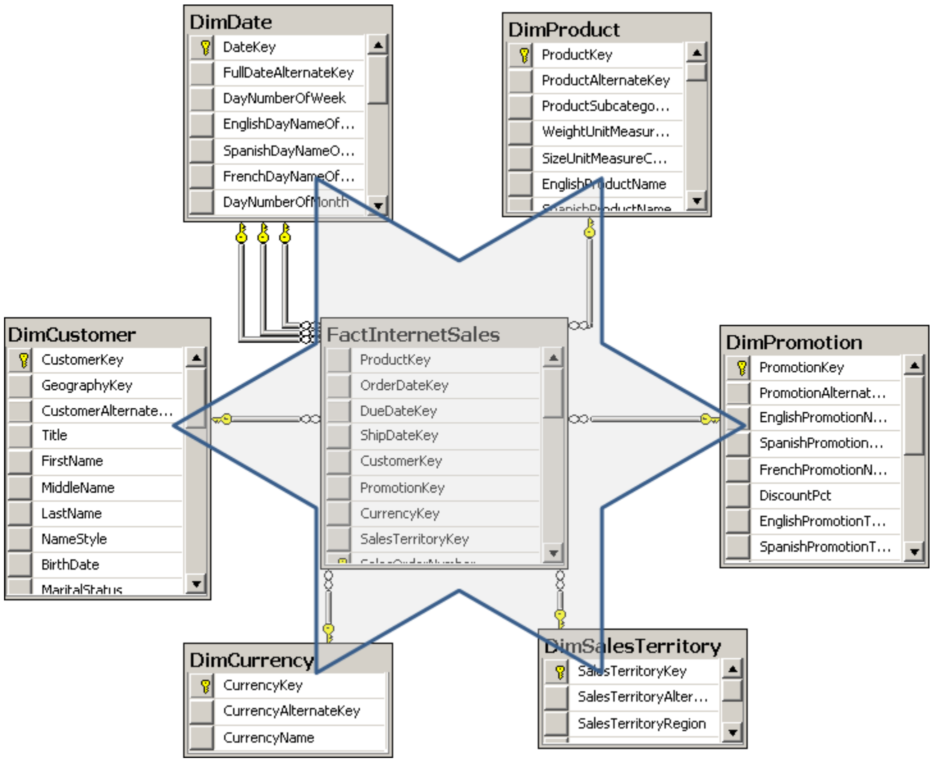


FIGURE 1-2 A Star schema example.

The fact table is connected to all the dimensions with foreign keys. Usually, all foreign keys taken together uniquely identify each row in the fact table, and thus collectively form a unique key, so you can use all the foreign keys as a composite primary key of the fact table. You can also add a simpler key. The fact table is on the "many" side of its relationships with the dimensions. If you were to form a proposition from a row in a fact table, you might express it with a sentence such as, "Customer A purchased product B on date C in quantity D for amount E." This proposition is a fact; this is how the fact table got its name.

The Star schema evolved from a conceptual model of a cube. You can imagine all sales as a big box. When you search for a problem in sales data, you use a divide-and-conquer technique: slicing the cube over different categories of customers, products, or time. In other words, you slice the cube over its dimensions. Therefore, customers, products, and time represent the three dimensions in the conceptual model of the sales cube. Dimension tables (dimensions) got their name from this conceptual model. In a logical model of a Star schema, you can represent more than three dimensions. Therefore, a Star schema represents a multi-dimensional hypercube.

As you already know, a data warehouse consists of multiple Star schemas. From a business perspective, these Star schemas are connected. For example, you have the same customers in sales as in accounting. You deal with many of the same products in sales, inventory, and production. Of course, your business is performed at the same time over all the different business areas. To represent the business correctly, you must be able to connect the multiple Star schemas in your data warehouse. The connection is simple – you use the same dimensions for each Star schema. In fact, the dimensions should be shared among multiple Star schemas. Dimensions have foreign key relationships with multiple fact tables. Dimensions with connections to multiple fact tables are called *shared* or *conformed dimensions*. Figure 1-3 shows a conformed dimension from the AdventureWorksDW2012 sample database with two different fact tables sharing the same dimension.

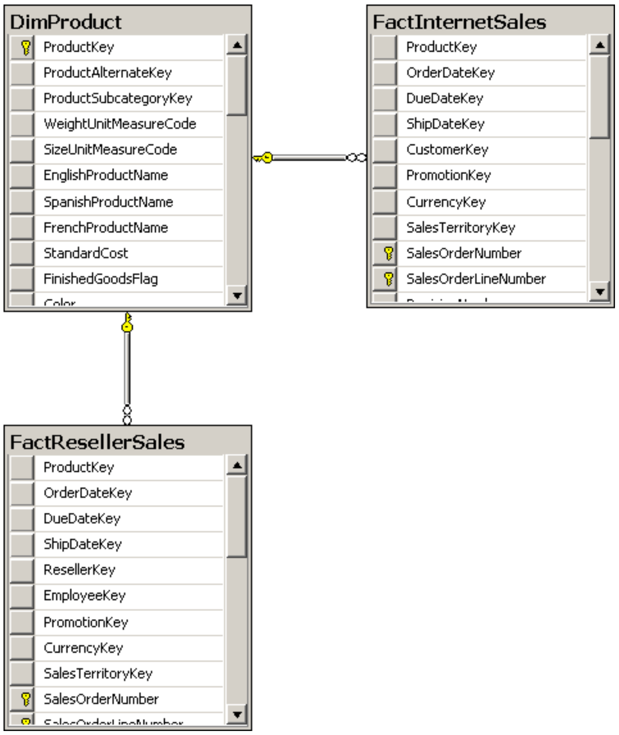


FIGURE 1-3 *DimProduct* is a shared dimension.

In the past, there was a big debate over whether to use shared or private dimensions. Private dimensions are dimensions that pertain to only a single Star schema. However, it is quite simple to design shared dimensions; you do not gain much from the design-time perspective by using private dimensions. In fact, with private dimensions, you lose the connections between the different fact tables, so you cannot compare the data in different fact tables over the same dimensions. For example, you could not compare sales and accounting data for the same customer if the sales and accounting fact tables didn't share the same customer dimension. Therefore, unless you are creating a small proof-of-concept (POC) project that covers only a single business area where you do not care about connections with different business areas, you should always opt for shared dimensions.

A data warehouse is often the source for specialized analytical database management systems, such as SQL Server Analysis Services (SSAS). SSAS is a system that performs specialized analyses by drilling down and is used for analyses that are based on the conceptual model of a cube. Systems such as SSAS focus on a single task and fast analyses, and they're considerably more optimized for this task than general systems such as SQL Server. SSAS enables analysis in real time, a process called *online analytical processing* (OLAP). However, to get such performance, you have to pay a price. SSAS is out of the scope of this book, but you have to know the limitations of SSAS to prepare a data warehouse in a way that is useful for SSAS. One thing to remember is that in an SSAS database, you can use shared dimensions only. This is just one more reason why you should prefer shared to private dimensions.

Snowflake Schema

Figure 1-4 shows a more detailed view of the *DimDate* dimension from the AdventureWorks-DW2012 sample database.

The highlighted attributes show that the dimension is denormalized. It is not in third normal form. In third normal form, all non-key columns should nontransitively depend on the key. A different way to say this is that there should be no functional dependency between non-key columns. You should be able to retrieve the value of a non-key column only if you know the key. However, in the *DimDate* dimension, if you know the month, you obviously know the calendar quarter, and if you know the calendar quarter, you know the calendar semester.

In a Star schema, dimensions are denormalized. In contrast, in an LOB normalized schema, you would split the table into multiple tables if you found a dependency between non-key columns. Figure 1-5 shows such a normalized example for the *DimProduct*, *DimProduct-Subcategory* and *DimProductCategory* tables from the AdventureWorksDW2012 database.

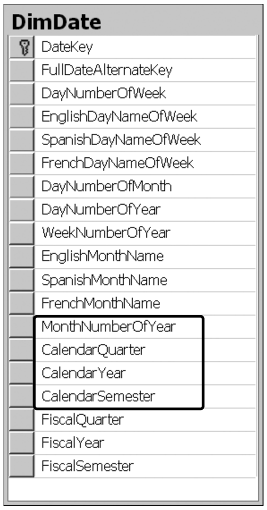


FIGURE 1-4 The *DimDate* denormalized dimension.

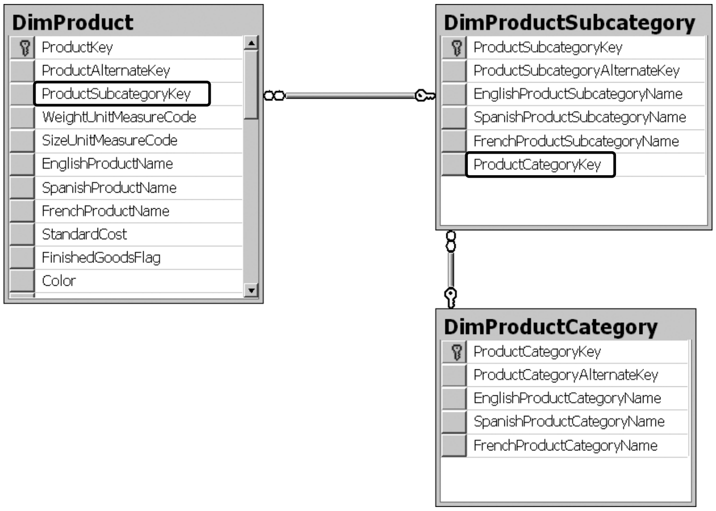


FIGURE 1-5 The *DimProduct* normalized dimension.

The *DimProduct* dimension is not denormalized. The *DimProduct* table does not contain the subcategory name, only the ProductSubcategoryKey value for the foreign key to the *DimProductSubcategory* lookup table. Similarly, the *DimProductSubcategory* table does not contain a category name; it just holds the foreign key ProductCategoryKey from the *Dim-ProductCategory* table. This design is typical of an LOB database schema.

You can imagine multiple dimensions designed in a similar normalized way, with a central fact table connected by foreign keys to dimension tables, which are connected with foreign keys to lookup tables, which are connected with foreign keys to second-level lookup tables.

In this configuration, a star starts to resemble a snowflake. Therefore, a Star schema with normalized dimensions is called a *Snowflake schema.*

In most long-term projects, you should design Star schemas. Because the Star schema is simpler than a Snowflake schema, it is also easier to maintain. Queries on a Star schema are simpler and faster than queries on a Snowflake schema, because they involve fewer joins. The Snowflake schema is more appropriate for short POC projects, because it is closer to an LOB normalized relational schema and thus requires less work to build.

EXAM TIP

If you do not use OLAP cubes and your reports query your data warehouse directly, then using a Star instead of a Snowflake schema might speed up the reports, because your reporting queries involve fewer joins.

In some cases, you can also employ a hybrid approach, using a Snowflake schema only for the first level of a dimension lookup table. In this type of approach, there are no additional levels of lookup tables; the first-level lookup table is denormalized. Figure 1-6 shows such a partially denormalized schema.

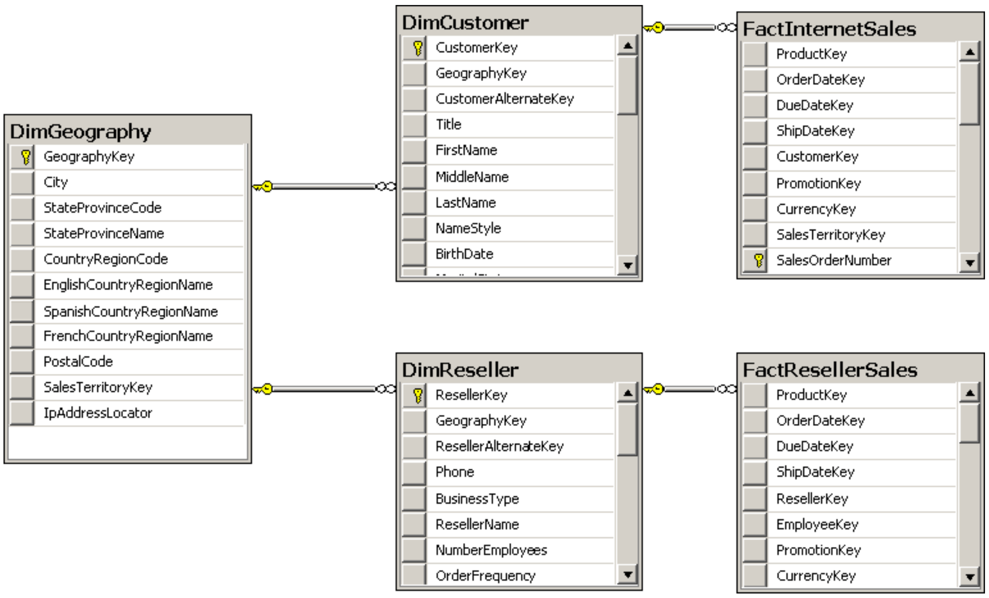


FIGURE 1-6 Partially denormalized dimensions.

In Figure 1-6, the *DimCustomer* and *DimReseller* dimensions are partially normalized. The dimensions now contain only the GeographyKey foreign key. However, the *DimGeography* table is denormalized. There is no additional lookup table even though a city is in a region and a region is in a country. A hybrid design such as this means that geography data is written only once and needs to be maintained in only a single place. Such a design is appropriate when multiple dimensions share the same attributes. In other cases, you should use the simpler Star schema. To repeat: you should use a Snowflake schema only for quick POC projects.

Quick Check

- **How do you connect multiple Star schemas in a DW?**

Quick Check Answer

- **You connect multiple Star schemas through shared dimensions.**

Granularity Level

The number of dimensions connected with a fact table defines the level of granularity of analysis you can get. For example, if no products dimension is connected to a sales fact table, you cannot get a report at the product level—you could get a report for sales for all products only. This kind of granularity is also called the *dimensionality* of a Star schema.

But there is another kind of granularity, which lets you know what level of information a dimension foreign key represents in a fact table. Different fact tables can have different granularity in a connection to the same dimension. This is very typical in budgeting and planning scenarios. For example, you do not plan that customer A will come on date B to store C and buy product D for amount E. Instead, you plan on a higher level—you might plan to sell amount E of products C in quarter B in all stores in that region to all customers in that region. Figure 1-7 shows an example of a fact table that uses a higher level of granularity than the fact tables introduced so far.

In the AdventureWorksDW2012 database, the *FactSalesQuota* table is the fact table with planning data. However, plans are made for employees at the per-quarter level only. The plan is for all customers, all products, and so on, because this Star schema uses only the *DimDate* and *DimEmployee* dimensions. In addition, planning occurs at the quarterly level. By investigating the content, you could see that all plans for a quarter are bound to the first day of a quarter. You would not need to use the DateKey; you could have only *CalendarYear* and *CalendarQuarter* columns in the *FactSalesQuota* fact table. You could still perform joins to *DimDate* by using these two columns—they are both present in the *DimDate* table as well. However, if you want to have a foreign key to the *DimDate* dimension, you do need the DateKey. A foreign key must refer to unique values on the "one" side of the relationship. The combination of *CalendarYear* and *CalendarQuarter* is, of course, not unique in the *DimDate* dimension; it repeats approximately 90 times in each quarter.

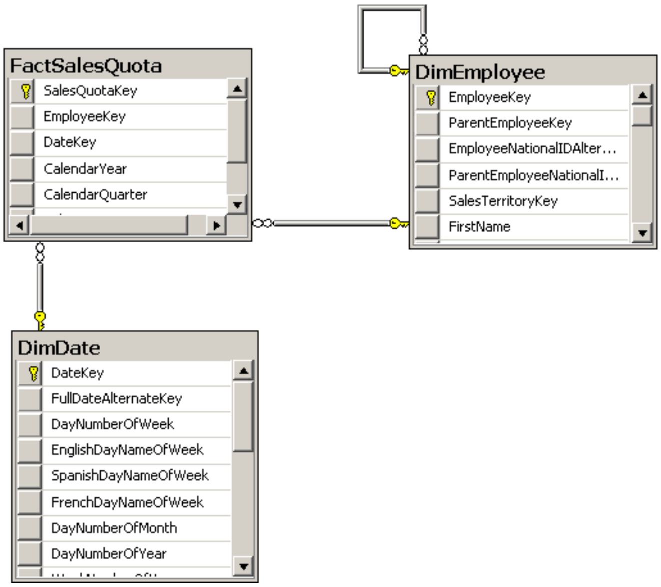


FIGURE 1-7 A fact table with a higher level of granularity.

Auditing and Lineage

In addition to tables for reports, a data warehouse may also include auditing tables. For every update, you should audit who made the update, when it was made, and how many rows were transferred to each dimension and fact table in your DW. If you also audit how much time was needed for each load, you can calculate the performance and take action if it deteriorates. You store this information in an auditing table or tables. However, you should realize that auditing does not help you unless you analyze the information regularly.

Auditing tables hold batch-level information about regular DW loads, but you might also want or need to have more detailed information. For example, you might want to know where each row in a dimension and/or fact table came from and when it was added. In such cases, you must add appropriate columns to the dimension and fact tables. Such detailed auditing information is also called *lineage* in DW terminology. To collect either auditing or lineage information, you need to modify the extract-transform-load (ETL) process you use for DW loads appropriately.

If your ETL tool is SQL Server Integration Services (SSIS), then you should use SSIS logging. SSIS has extensive logging support. In addition, SSIS also has support for lineage information.

PRACTICE Reviewing the AdventureWorksDW2012 Internet Sales Schema

The AdventureWorksDW2012 sample database is a good example of a data warehouse. It has all the elements needed to allow you to see examples of various types of dimensional modeling.

EXERCISE 1 Review the AdventureWorksDW2012 Database Schema

In this exercise, you review the database schema.

1. Start SSMS and connect to your instance of SQL Server. Expand the Databases folder and then the AdventureWorksDW2012 database.
2. Right-click the Database Diagrams folder and select the New Database Diagram option. If no diagrams were ever created in this database, you will see a message box informing you that the database has no support objects for diagramming. If that message appears, click Yes to create the support objects.
3. From the Add Table list, select the following tables (click each table and then click the Add button):
   - *DimCustomer*
   - *DimDate*
   - *DimGeography*
   - *DimProduct*
   - *DimProductCategory*
   - *DimProductSubcategory*
   - *FactInternetSales*

   Your diagram should look similar to Figure 1-8.

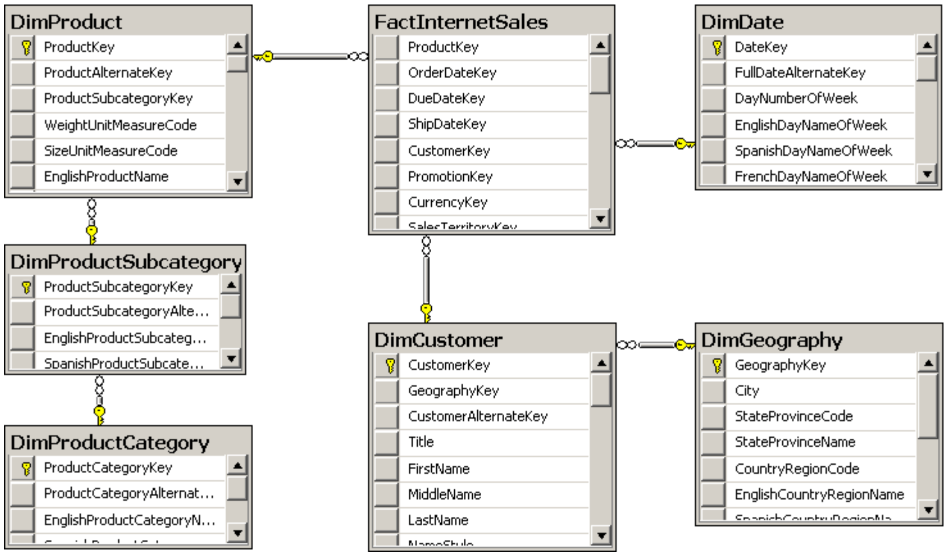


FIGURE 1-8 The AdventureWorksDW2012 Internet Sales Schema.

4. Thoroughly analyze the tables, columns, and relationships.
5. Save the diagram with the name **Practice_01_01_InternetSales**.

EXERCISE 2 Analyze the Diagram

Review the AdventureWorksDW2012 schema to note the following facts:

- The *DimDate* dimension has no additional lookup tables associated with it and therefore uses the Star schema.
- The *DimProduct* table is snowflaked; it uses the *DimProductSubcategory* lookup table, which further uses the *DimProductCategory* lookup table.
- The *DimCustomer* dimension uses a hybrid schema—the first level of the Snowflake schema only through the *DimGeography* lookup table. The *DimGeography* table is denormalized; it does not have a relationship with any other lookup table.
- There are no specific columns for lineage information in any of the tables.

Close the diagram.

NOTE **CONTINUING WITH PRACTICES**

Do not exit SSMS if you intend to continue immediately with the next practice.

Lesson Summary

- The Star schema is the most common design for a DW.
- The Snowflake schema is more appropriate for POC projects.
- You should also determine the granularity of fact tables, as well as auditing and lineage needs.

Lesson Review

Answer the following questions to test your knowledge of the information in this lesson. You can find the answers to these questions and explanations of why each answer choice is correct or incorrect in the "Answers" section at the end of this chapter.

1. Reporting from a Star schema is simpler than reporting from a normalized online transactional processing (OLTP) schema. What are the reasons for wanting simpler reporting? (Choose all that apply.)
   - **A.** A Star schema typically has fewer tables than a normalized schema. Therefore, queries are simpler because they require fewer joins.
   - **B.** A Star schema has better support for numeric data types than a normalized relational schema; therefore, it is easier to create aggregates.
   - **C.** There are specific Transact-SQL expressions that deal with Star schemas.
   - **D.** A Star schema is standardized and narrative; you can find the information you need for a report quickly.
2. You are creating a quick POC project. Which schema is the most suitable for this kind of a project?
   - **A.** Star schema
   - **B.** Normalized schema
   - **C.** Snowflake schema
   - **D.** XML schema
3. A Star schema has two types of tables. What are those two types? (Choose all that apply.)
   - **A.** Lookup tables
   - **B.** Dimensions
   - **C.** Measures
   - **D.** Fact tables

Lesson 2: Designing Dimensions

Star and Snowflake schemas are the de facto standard. However, the standard does not end with schema shapes. Dimension and fact table columns are part of this informal standard as well and are introduced in this lesson, along with natural hierarchies, which are especially useful as natural drill-down paths for analyses. Finally, the lesson discusses a common problem with handling dimension changes over time.

After this lesson, you will be able to:

- Define dimension column types.
- Use natural hierarchies.
- Understand and resolve the slowly changing dimensions problem.

Estimated lesson time: 40 minutes

Dimension Column Types

Dimensions give context to measures. Typical analysis includes pivot tables and pivot graphs. These pivot on one or more dimension columns used for analysis—these columns are called *attributes* in DW and OLAP terminology. The naming convention in DW/OLAP terminology is a little odd; in a relational model, every column represents an attribute of an entity. Don’t worry too much about the correctness of naming in DW/OLAP terminology. The important point here is for you to understand what the word “attribute” means in a DW/OLAP context.

Pivoting makes no sense if an attribute’s values are continuous, or if an attribute has too many distinct values. Imagine how a pivot table would look if it had 1,000 columns, or how a pivot graph would look with 1,000 bars. For pivoting, discrete attributes with a small number of distinct values is most appropriate. A bar chart with more than 10 bars becomes difficult to comprehend. Continuous columns or columns with unique values, such as keys, are not appropriate for analyses.

If you have a continuous column and you would like to use it in analyses as a pivoting attribute, you should discretize it. *Discretizing* means grouping or binning values to a few discrete groups. If you are using OLAP cubes, SSAS can help you. SSAS can discretize continuous attributes. However, automatic discretization is usually worse than discretization from a business perspective. Age and income are typical attributes that should be discretized from a business perspective. One year makes a big difference when you are 15 years old, and much less when you are 55 years old. When you discretize age, you should use narrower ranges for younger people and wider ranges for older people.

IMPORTANT **AUTOMATIC DISCRETIZATION**

Use automatic discretization for POC projects only. For long-term projects, always discretize from a business perspective.

Columns with unique values identify rows. These columns are *keys*. In a data warehouse, you need keys just like you need them in an LOB database. Keys uniquely identify entities. Therefore, keys are the second type of columns in a dimension.

After you identify a customer, you do not refer to that customer with the key value. Having only keys in a report does not make the report very readable. People refer to entities by using their names. In a DW dimension, you also need one or more columns that you use for naming an entity.

A customer typically has an address, a phone number, and an email address. You do not analyze data on these columns. You do not need them for pivoting. However, you often need information such as the customer's address on a report. If that data is not present in a DW, you will need to get it from an LOB database, probably with a distributed query. It is much simpler to store this data in your data warehouse. In addition, queries that use this data perform better, because the queries do not have to include data from LOB databases. Columns used in reports as labels only, not for pivoting, are called *member properties*.

You can have naming and member property columns in multiple languages in your dimension tables, providing the translation for each language you need to support. SSAS can use your translations automatically. For reports from a data warehouse, you need to manually select columns with appropriate language translation.

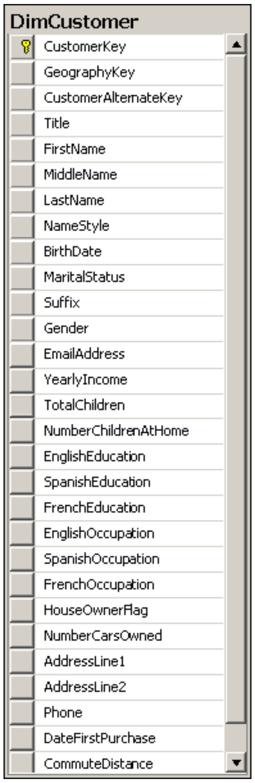
In addition to the types of dimension columns already defined for identifying, naming, pivoting, and labeling on a report, you can have columns for lineage information, as you saw in the previous lesson. There is an important difference between lineage and other columns: lineage columns are never exposed to end users and are never shown on end users' reports.

To summarize, a dimension may contain the following types of columns:

- **Keys** Used to identify entities
- **Name columns** Used for human names of entities
- **Attributes** Used for pivoting in analyses
- **Member properties** Used for labels in a report
- **Lineage columns** Used for auditing, and never exposed to end users

Hierarchies

Figure 1-9 shows the *DimCustomer* dimension of the AdventureWorksDW2012 sample database.



DimCustomer

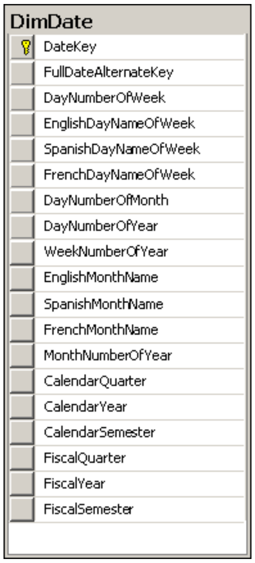
CustomerKey
GeographyKey
CustomerAlternateKey
Title
FirstName
MiddleName
LastName
NameStyle
BirthDate
MaritalStatus
Suffix
Gender
EmailAddress
YearlyIncome
TotalChildren
NumberChildrenAtHome
EnglishEducation
SpanishEducation
FrenchEducation
EnglishOccupation
SpanishOccupation
FrenchOccupation
HouseOwnerFlag
NumberCarsOwned
AddressLine1
AddressLine2
Phone
DateFirstPurchase
CommuteDistance

FIGURE 1-9 The *DimCustomer* dimension.

In the figure, the following columns are attributes (columns used for pivoting):

- *BirthDate* (after calculating age and discretizing the age)
- *MaritalStatus*
- *Gender*
- *YearlyIncome* (after discretizing)
- *TotalChildren*
- *NumberChildrenAtHome*
- *EnglishEducation* (other education columns are for translations)
- *EnglishOccupation* (other occupation columns are for translations)
- *HouseOwnerFlag*
- *NumberCarsOwned*
- *CommuteDistance*

All these attributes are unrelated. Pivoting on *MaritalStatus*, for example, is unrelated to pivoting on *YearlyIncome*. None of these columns have any functional dependency between them, and there is no natural drill-down path through these attributes. Now look at the *DimDate* columns, as shown in Figure 1-10.


DimDate

DateKey
FullDateAlternateKey
DayNumberOfWeek
EnglishDayNameOfWeek
SpanishDayNameOfWeek
FrenchDayNameOfWeek
DayNumberOfMonth
DayNumberOfYear
WeekNumberOfYear
EnglishMonthName
SpanishMonthName
FrenchMonthName
MonthNumberOfYear
CalendarQuarter
CalendarYear
CalendarSemester
FiscalQuarter
FiscalYear
FiscalSemester

FIGURE 1-10 The *DimDate* dimension.

Some attributes of the *DimDate* dimension include the following (not in the order shown in the figure):

- *FullDateAlternateKey* (denotes a date in date format)
- *EnglishMonthName*
- *CalendarQuarter*
- *CalendarSemester*
- *CalendarYear*

You will immediately notice that these attributes are connected. There is a functional dependency among them, so they break third normal form. They form a hierarchy. Hierarchies are particularly useful for pivoting and OLAP analyses—they provide a natural drill-down path. You perform divide-and-conquer analyses through hierarchies.

Hierarchies have levels. When drilling down, you move from a parent level to a child level. For example, a calendar drill-down path in the *DimDate* dimension goes through the following levels: *CalendarYear* → *CalendarSemester* → *CalendarQuarter* → *EnglishMonthName* → *FullDateAlternateKey*.

At each level, you have members. For example, the members of the month level are, of course, January, February, March, April, May, June, July, August, September, October, November, and December. In DW and OLAP jargon, rows on the leaf level—the actual dimension rows—are called *members*. This is why dimension columns used in reports for labels are called *member properties*.

In a Snowflake schema, lookup tables show you levels of hierarchies. In a Star schema, you need to extract natural hierarchies from the names and content of columns. Nevertheless, because drilling down through natural hierarchies is so useful and welcomed by end users, you should use them as much as possible.

Note also that attribute names are used for labels of row and column groups in a pivot table. Therefore, a good naming convention is crucial for a data warehouse. You should always use meaningful and descriptive names for dimensions and attributes.

Slowly Changing Dimensions

There is one common problem with dimensions in a data warehouse: the data in the dimension changes over time. This is usually not a problem in an OLTP application; when a piece of data changes, you just update it. However, in a DW, you have to maintain history. The question that arises is *how* to maintain it. Do you want to update only the changed data, as in an OLTP application, and pretend that the value was always the last value, or do you want to maintain both the first and intermediate values? This problem is known in DW jargon as the *Slowly Changing Dimension (SCD)* problem.

The problem is best explained in an example. Table 1-1 shows original source OLTP data for a customer.

TABLE 1-1 Original OLTP Data for a Customer

CustomerId	FullName	City	Occupation
17	Bostjan Strazar	Vienna	Professional

The customer lives in Vienna, Austria, and is a professional. Now imagine that the customer moves to Ljubljana, Slovenia. In an OLTP database, you would just update the *City* column, resulting in the values shown in Table 1-2.

TABLE 1-2 OLTP Data for a Customer After the City Change

CustomerId	FullName	City	Occupation
17	Bostjan Strazar	Ljubljana	Professional

If you create a report, all the historical sales for this customer are now attributed to the city of Ljubljana, and (on a higher level) to Slovenia. The fact that this customer contributed to sales in Vienna and in Austria in the past would have disappeared.

In a DW, you can have the same data as in an OLTP database. You could use the same key, such as the business key, for your *Customer* dimension. You could update the *City* column when you get a change notification from the OLTP system, and thus overwrite the history.

This kind of change management is called *Type 1 SCD*. To recapitulate, Type 1 means overwriting the history for an attribute and for all higher levels of hierarchies to which that attribute belongs.

But you might prefer to maintain the history, to capture the fact that the customer contributed to sales in another city and country or region. In that case, you cannot just overwrite the data; you have to insert a new row containing new data instead. Of course, the values of other columns that do not change remain the same. However, that creates a new problem. If you simply add a new row for the customer with the same key value, the key would no longer be unique. In fact, if you tried to use a primary key or unique constraint as the key, the constraint would reject such an insert. Therefore, you have to do something with the key. You should not modify the business key, because you need a connection with the source system. The solution is to introduce a new key, a *data warehouse key*. In DW terminology, this kind of key is called a *surrogate key*.

Preserving the history while adding new rows is known as *Type 2 SCD*. When you implement Type 2 SCD, for the sake of simpler querying, you typically also add a flag to denote which row is current for a dimension member. Alternatively, you could add two columns showing the interval of validity of a value. The data type of the two columns should be Date, and the columns should show the values Valid From and Valid To. For the current value, the *Valid To* column should be NULL. Table 1-3 shows an example of the flag version of Type 2 SCD handling.

TABLE 1-3 An SCD Type 2 Change

DWCId	CustomerId	FullName	City	Occupation	Current
17	17	Bostjan Strazar	Vienna	Professional	0
289	17	Bostjan Strazar	Ljubljana	Professional	1

You could have a mixture of Type 1 and Type 2 changes in a single dimension. For example, in Table 1-3, you might want to maintain the history for the *City* column but overwrite the history for the *Occupation* column. That raises yet another issue. When you want to update the *Occupation* column, you may find that there are two (and maybe more) rows for the same customer. The question is, do you want to update the last row only, or all the rows? Table 1-4 shows a version that updates the last (current) row only, whereas Table 1-5 shows all of the rows being updated.

TABLE 1-4 An SCD Type 1 and Type 2 Mixture, Updating the Current Row Only

DWCId	CustomerId	FullName	City	Occupation	Current
17	17	Bostjan Strazar	Vienna	Professional	0
289	17	Bostjan Strazar	Ljubljana	Management	1

TABLE 1-5 An SCD Type 1 and Type 2 Mixture, Updating All Rows

DWCId	CustomerId	FullName	City	Occupation	Current
17	17	Bostjan Strazar	Vienna	Management	0
289	17	Bostjan Strazar	Ljubljana	Management	1

Although Type 1 and Type 2 handling are most common, other solutions exist. Especially well-known is *Type 3 SCD*, in which you manage a limited amount of history through additional historical columns. Table 1-6 shows Type 3 handling for the *City* column.

TABLE 1-6 SCD Type 3

CustomerId	FullName	CurrentCity	PreviousCity	Occupation
17	Bostjan Strazar	Ljubljana	Vienna	Professional

You can see that by using only a single historical column, you can maintain only one historical value per column. So Type 3 SCD has limited usability and is far less popular than Types 1 and 2.

Which solution should you implement? You should discuss this with end users and subject matter experts (SMEs). They should decide for which attributes to maintain the history, and for which ones to overwrite the history. You should then choose a solution that uses Type 2, Type 1, or a mixture of Types 1 and 2, as appropriate.

However, there is an important caveat. To maintain customer history correctly, you must have some attribute that uniquely identifies that customer throughout that customer's history, and that attribute must not change. Such an attribute should be the original—the business key. In an OLTP database, business keys should not change.

Business keys should also not change if you are merging data from multiple sources. For merged data, you usually have to implement a new, surrogate key, because business keys from different sources can have the same value for different entities. However, business keys should not change; otherwise you lose the connection with the OLTP system. Using surrogate keys in a data warehouse for at least the most common dimensions (those representing customers, products, and similar important data), is considered a best practice. Not changing OLTP keys is a best practice as well.

EXAM TIP

Make sure you understand why you need surrogate keys in a data warehouse.

PRACTICE Reviewing the AdventureWorksDW2012 Dimensions

The AdventureWorksDW2012 sample database has many dimensions. In this practice, you will explore some of them.

EXERCISE 1 Explore the AdventureWorksDW2012 Dimensions

In this exercise, you create a diagram for the dimensions.

1. If you closed SSMS, start it and connect to your SQL Server instance. Expand the Databases folder and then the AdventureWorksDW2012 database.
2. Right-click the Database Diagrams folder, and then select the New Database Diagram option.
3. From the Add Table list, select the following tables (click each table and then click the Add button):
   - *DimProduct*
   - *DimProductCategory*
   - *DimProductSubcategory*

   Your diagram should look like Figure 1-11.
4. Try to figure out which columns are used for the following purposes:
   - Keys
   - Names
   - Translations
   - Attributes
   - Member properties
   - Lineage
   - Natural hierarchies
5. Try to figure out whether the tables in the diagram are prepared for a Type 2 SCD change.
6. Add the *DimSalesReason* table to the diagram.
7. Try to figure out whether there is some natural hierarchy between attributes of the *DimSalesReason* dimension. Your diagram should look like Figure 1-12.
8. Save the diagram with the name **Practice_01_02_Dimensions**.

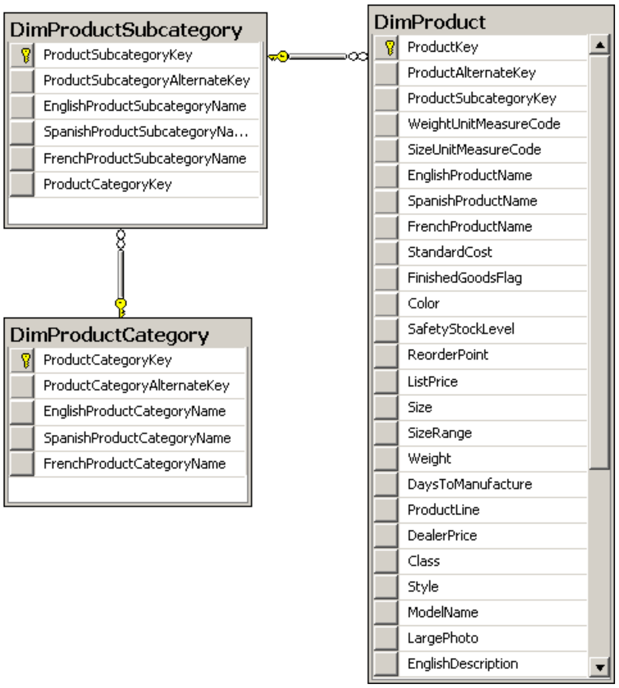


FIGURE 1-11 *DimProduct* and related tables.

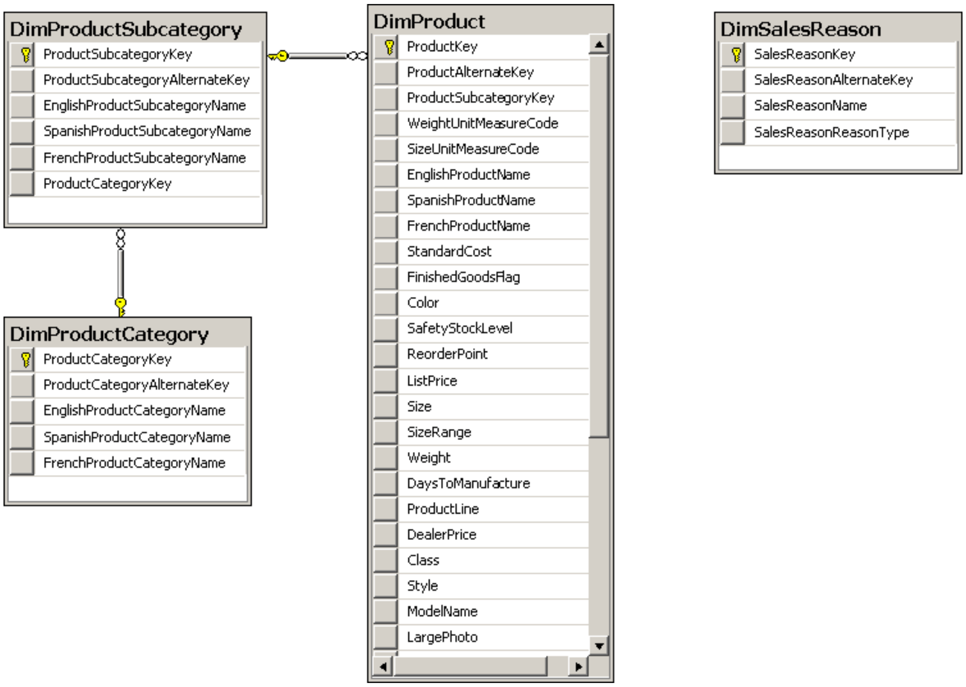


FIGURE 1-12 Adding *DimSalesReason*.

EXERCISE 2 Further Analyze the Diagram

In this exercise, review the database schema from the previous exercise to learn more:

- The *DimProduct* dimension has a natural hierarchy: *ProductCategory* → *ProductSubcategory* → *Product*.
- The *DimProduct* dimension has many additional attributes that are useful for pivoting but that are not a part of any natural hierarchy. For example, Color and Size are such attributes.
- Some columns in the *DimProduct* dimension, such as the *LargePhoto* and *Description* columns, are member properties.
- *DimSalesReason* uses a Star schema. In a Star schema, it is more difficult to spot natural hierarchies. Though you can simply follow the lookup tables in a Snowflake schema and find levels of hierarchies, you have to recognize hierarchies from attribute names in a Star schema. If you cannot extract hierarchies from column names, you could also check the data. In the *DimSalesReason* dimension, it seems that there is a natural hierarchy: *SalesReasonReasonType* → *SalesReasonName*.

Close the diagram.

***NOTE* CONTINUING WITH PRACTICES**

Do not exit SSMS if you intend to continue immediately with the next practice.

Lesson Summary

- In a dimension, you have the following column types: keys, names, attributes, member properties, translations, and lineage.
- Some attributes form natural hierarchies.
- There are standard solutions for the Slowly Changing Dimensions (SCD) problem.

Lesson Review

Answer the following questions to test your knowledge of the information in this lesson. You can find the answers to these questions and explanations of why each answer choice is correct or incorrect in the "Answers" section at the end of this chapter.

1. You implement a Type 2 solution for an SCD problem for a specific column. What do you actually do when you get a changed value for the column from the source system?
   - **A.** Add a column for the previous value to the table. Move the current value of the updated column to the new column. Update the current value with the new value from the source system.

   - **B.** Insert a new row for the same dimension member with the new value for the updated column. Use a surrogate key, because the business key is now duplicated. Add a flag that denotes which row is current for a member.
   - **C.** Do nothing, because in a DW, you maintain history, you do not update dimension data.
   - **D.** Update the value of the column just as it was updated in the source system.

2. Which kind of a column is not a part of a dimension?
   - **A.** Attribute
   - **B.** Measure
   - **C.** Key
   - **D.** Member property
   - **E.** Name

3. How can you spot natural hierarchies in a Snowflake schema?
   - **A.** You need to analyze the content of the attributes of each dimension.
   - **B.** Lookup tables for each dimension provide natural hierarchies.
   - **C.** A Snowflake schema does not support hierarchies.
   - **D.** You should convert the Snowflake schema to the Star schema, and then you would spot the natural hierarchies immediately.

Lesson 3: Designing Fact Tables

Fact tables, like dimensions, have specific types of columns that limit the actions that can be taken with them. Queries from a DW aggregate data; depending on the particular type of column, there are some limitations on which aggregate functions you can use. Many-to-many relationships in a DW can be implemented differently than in a normalized relational schema.

After this lesson, you will be able to:

- Define fact table column types.
- Understand the additivity of a measure.
- Handle many-to-many relationships in a Star schema.

Estimated lesson time: 30 minutes

Fact Table Column Types

Fact tables are collections of measurements associated with a specific business process. You store measurements in columns. Logically, this type of column is called a *measure*. Measures are the essence of a fact table. They are usually numeric and can be aggregated. They store values that are of interest to the business, such as sales amount, order quantity, and discount amount.

From Lesson 1 in this chapter, you already saw that a fact table includes foreign keys from all dimensions. These foreign keys are the second type of column in a fact table. A fact table is on the "many" side of the relationships with dimensions. All foreign keys together usually uniquely identify each row and can be used as a composite primary key.

You often include an additional surrogate key. This key is shorter and consists of one or two columns only. The surrogate key is usually the business key from the table that was used as the primary source for the fact table. For example, suppose you start building a sales fact table from an order details table in a source system, and then add foreign keys that pertain to the order as a whole from the *Order Header* table in the source system. Tables 1-7, 1-8, and 1-9 illustrate an example of such a design process.

Table 1-7 shows a simplified example of an *Orders Header* source table. The *OrderId* column is the primary key for this table. The *CustomerId* column is a foreign key from the *Customers* table. The *OrderDate* column is not a foreign key in the source table; however, it becomes a foreign key in the DW fact table, for the relationship with the explicit date dimension. Note, however, that foreign keys in a fact table can—and usually are—replaced with DW surrogate keys of DW dimensions.

TABLE 1-7 The Source Orders Header Table

OrderId	CustomerId	Orderdate
12541	17	2012/02/21

Table 1-8 shows the source *Order Details* table. The primary key of this table is a composite one and consists of the *OrderId* and *LineItemId* columns. In addition, the source *Order Details* table has the *ProductId* foreign key column. The *Quantity* column is the measure.

TABLE 1-8 The Source Order Details Table

OrderId	LineItemId	ProductId	Quantity
12541	2	5	47

Table 1-9 shows the *Sales Fact* table created from the *Orders Header* and *Order Details* source tables. The *Order Details* table was the primary source for this fact table. The *OrderId*, *LineItemId*, and *Quantity* columns are simply transferred from the source *Order Details* table. The *ProductId* column from the source *Order Details* table is replaced with a surrogate DW *ProductKey* column. The *CustomerId* and *OrderDate* columns take the source *Orders Header* table; these columns pertain to orders, not order details. However, in the fact table, they are replaced with the surrogate DW keys CustomerKey and OrderDateKey.

TABLE 1-9 The Sales Fact Table

OrderId	LineItemId	CustomerKey	OrderDateKey	ProductKey	Quantity
12541	2	289	444	25	47

You do not need the *OrderId* and *LineItemId* columns in this sales fact table. For analyses, you could create a composite primary key from the *CustomerKey*, *OrderDateKey*, and *Product-Key* columns. However, you should keep the *OrderId* and *LineItemId* columns to make quick controls and comparisons with source data possible. In addition, if you were to use them as the primary key, then the primary key would be shorter than one composed from all foreign keys.

The last column type used in a fact table is the lineage type, if you implement the lineage. Just as with dimensions, you never expose the lineage information to end users. To recapitu-late, fact tables have the following column types:

- Foreign keys
- Measures
- Lineage columns (optional)
- Business key columns from the primary source table (optional)

Additivity of Measures

Additivity of measures is not exactly a data warehouse design problem. However, you should consider which aggregate functions you will use in reports for which measures, and which ag-gregate functions you will use when aggregating over which dimension.

The simplest types of measures are those that can be aggregated with the SUM aggregate function across all dimensions, such as amounts or quantities. For example, if sales for product A were $200.00 and sales for product B were $150.00, then the total of the sales was $350.00. If yesterday's sales were $100.00 and sales for the day before yesterday were $130.00, then the total sales amounted to $230.00. Measures that can be summarized across all dimensions are called *additive measures*.

Some measures are not additive over any dimension. Examples include prices and percent-ages, such as a discount percentage. Typically, you use the AVERAGE aggregate function for such measures, or you do not aggregate them at all. Such measures are called *non-additive measures*. Often, you can sum additive measures and then calculate non-additive measures from the additive aggregations. For example, you can calculate the sum of sales amount and then divide that value by the sum of the order quantity to get the average price. On higher levels of aggregation, the calculated price is the average price; on the lowest level, it's the data itself—the calculated price is the actual price. This way, you can simplify queries.

For some measures, you can use SUM aggregate functions over all dimensions but time. Some examples include levels and balances. Such measures are called *semi-additive measures*. For example, if customer A has $2,000.00 in a bank account, and customer B has $3,000.00, together they have $5,000.00. However, if customer A had $5,000.00 in an account yesterday but has only $2,000.00 today, then customer A obviously does not have $7,000.00 altogether. You should take care how you aggregate such measures in a report. For time measures, you can calculate average value or use the last value as the aggregate.

Quick Check

- **You are designing an accounting system. Your measures are debit, credit, and balance. What is the additivity of each measure?**

Quick Check Answer

- **Debit and credit are additive measures, and balance is a semi-additive measure.**

Additivity of Measures in SSAS

SSAS is out of the scope of this book; however, you should know some facts about SSAS if your data warehouse is the source for SSAS databases. SSAS has support for semi-additive and non-additive measures. The SSAS database model is called the *Business Intelligence Semantic Model (BISM)*. Compared to the SQL Server database model, BISM includes much additional metadata.

SSAS has two types of storage: *dimensional* and *tabular*. Tabular storage is quicker to develop, because it works through tables like a data warehouse does. The dimensional model more properly represents a cube. However, the dimensional model includes even more metadata than the tabular model. In BISM dimensional processing, SSAS offers semi-additive aggregate functions out of the box. For example, SSAS offers the LastNonEmpty aggregate function, which properly uses the SUM aggregate function across all dimensions but time, and defines the last known value as the aggregate over time. In the BISM tabular model, you use the Data Analysis Expression (DAX) language. The DAX language includes functions that let you build semi-additive expressions quite quickly as well.

Many-to-Many Relationships

In a relational database, the many-to-many relationship between two tables is resolved through a third intermediate table. For example, in the AdventureWorksDW2012 database, every Internet sale can be associated with multiple reasons for the sale—and every reason can be associated with multiple sales. Figure 1-13 shows an example of a many-to-many relationship between *FactInternetSales* and *DimSalesReason* through the *FactInternetSalesReason* intermediate table in the AdventureWorksDW2012 sample database.

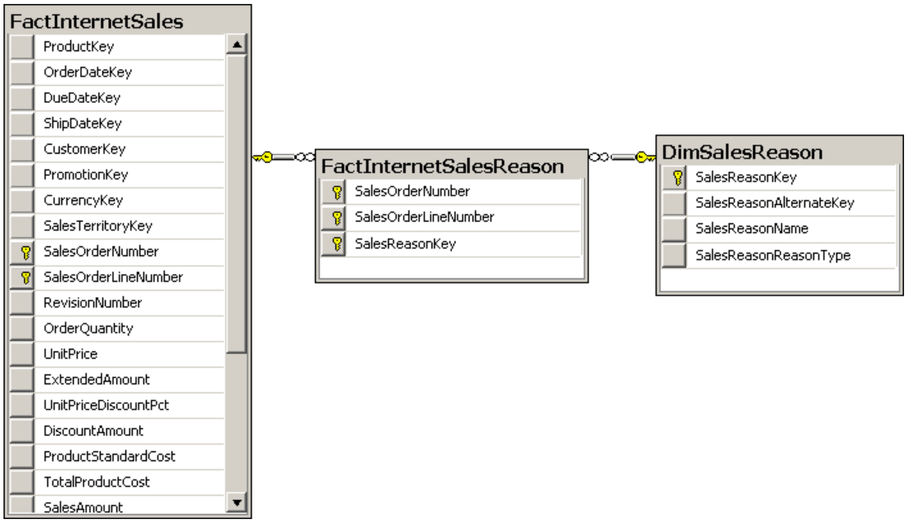


FIGURE 1-13 A classic many-to-many relationship.

For a data warehouse in a relational database management system (RDBMS), this is the correct model. However, SSAS has problems with this model. For reports from a DW, it is you, the developer, who writes queries. In contrast, reporting from SSAS databases is done by using client tools that read the schema and only afterwards build a user interface (UI) for selecting measures and attributes. Client tools create multi-dimensional expression (MDX) queries for the SSAS dimensional model, and DAX or MDX queries for the SSAS tabular model. To create the queries and build the UI properly, the tools rely on standard Star or Snowflake schemas. The tools expect that the central table, the fact table, is always on the "many" side of the relationship.

A quick look at Figure 1-13 reveals that the *FactInternetSales* fact table is on the "one" side of its relationship with the *FactInternetSalesReason* fact table. SSAS with a BISM tabular model does not support many-to-many relationships at all in its current version. In SSAS with a BISM dimensional model, you can solve the problem by creating an intermediate dimension between both fact tables. You create it from the primary key of the *FactInternetSales* table. Let's call this dimension *DimFactInternetSales*. Then you put it on the "one" side of the relationships with both fact tables. This way, both fact tables are always on the "many" side of any relationship. However, you have to realize that the relationship between the *FactInternetSales* and the new *DimFactInternetSales* dimension is de facto one to one.

EXAM TIP

Note that you create an intermediate dimension between two fact tables that supports SSAS many-to-many relationship from an existing fact table, and not directly from a table from the source transactional system.

You can generate such intermediate dimensions in your data warehouse and then just inherit them in your SSAS BISM dimensional database. (Note that SSAS with BISM in a tabular model does not recognize many-to-many relationships, even with an additional intermediate dimension table.) This way, you can have the same model in your DW as in your BISM dimensional database. In addition, when you recreate such a dimension, you can expose it to end users for reporting. However, a dimension containing key columns only is not very useful for reporting. To make it more useful, you can add additional attributes that form a hierarchy. Date variations, such as year, quarter, month, and day are very handy for drilling down. You can get these values from the *DimDate* dimension and enable a drill-down path of year → quarter → month → day → sales order in this dimension. Figure 1-14 shows a many-to-many relationship with an additional intermediate dimension.

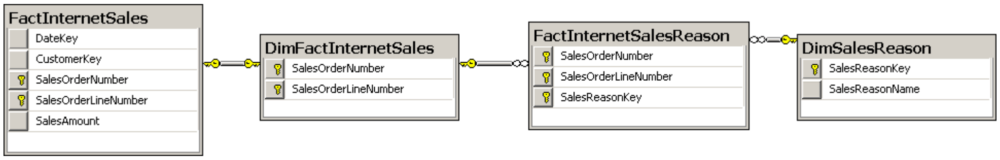


FIGURE 1-14 A many-to-many relationship with two intermediate tables.

Note that SSMS created the relationship between *DimFactInternetSales* and *FactInternet-Sales* as one to one.

PRACTICE Reviewing the AdventureWorksDW2012 Fact Tables

The AdventureWorksDW2012 sample database has many types of fact tables as well, in order to show all possible measures. In this practice, you are going to review one of them.

EXERCISE 1 Create a Diagram for an AdventureWorksDW2012 Fact Table

In this exercise, you create a database diagram for a fact table and two associated dimensions.

1. If you closed SSMS, start it and connect to your SQL Server instance. Expand the Databases folder and then the AdventureWorksDW2012 database.
2. Right-click the Database Diagrams folder and select the New Database Diagram option.
3. From the Add Table list, select the following tables (click each table and then click the Add button):
   - *DimProduct*
   - *DimDate*
   - *FactProductInventory*

Your diagram should look like Figure 1-15.

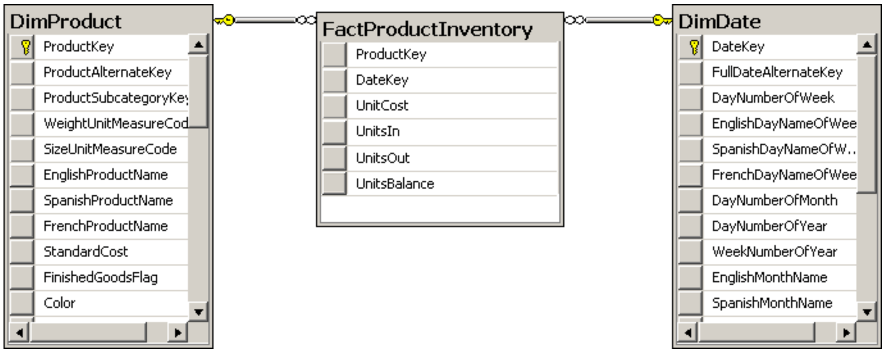


FIGURE 1-15 FactProductInventory and related tables.

EXERCISE 2 Analyze Fact Table Columns

In this exercise, you learn more details about the fact table in the schema you created in the previous exercise. Note that you have to conclude these details from the names of the measure columns; in a real-life project, you should check the content of the columns as well.

- Knowing how an inventory works, you can conclude that the UnitsIn and UnitsOut are additive measures. Using the SUM aggregate function for these two columns is reasonable for aggregations over any dimension.
- The UnitCost measure is a non-additive measure. Summing it over any dimension does not make sense.
- The UnitsBalance measure is a semi-additive measure. You can use the SUM aggregate function over any dimension but time.

Save the diagram using the name **Practice_01_03_ProductInventory**. Close the diagram and exit SSMS.

Lesson Summary

- Fact tables include measures, foreign keys, and possibly an additional primary key and lineage columns.
- Measures can be additive, non-additive, or semi-additive.
- For many-to-many relationships, you can introduce an additional intermediate dimension.

Lesson Review

Answer the following questions to test your knowledge of the information in this lesson. You can find the answers to these questions and explanations of why each answer choice is correct or incorrect in the "Answers" section at the end of this chapter.

1. Over which dimension can you not use the SUM aggregate function for semi-additive measures?
   - **A.** Customer
   - **B.** Product
   - **C.** Date
   - **D.** Employee
2. Which measures would you expect to be non-additive? (Choose all that apply.)
   - **A.** Price
   - **B.** Debit
   - **C.** SalesAmount
   - **D.** DiscountPct
   - **E.** UnitBalance
3. Which kind of a column is not part of a fact table?
   - **A.** Lineage
   - **B.** Measure
   - **C.** Key
   - **D.** Member property

Case Scenarios

In the following case scenarios, you apply what you've learned about Star and Snowflake schemas, dimensions, and the additivity of measures. You can find the answers to these questions in the "Answers" section at the end of this chapter.

Case Scenario 1: A Quick POC Project

You are hired to implement a quick POC data warehousing project. You have to prepare the schema for sales data. Your customer's SME would like to analyze sales data over customers, products, and time. Before creating a DW and tables, you need to make a couple of decisions and answer a couple of questions:

1. What kind of schema would you use?
2. What would the dimensions of your schema be?
3. Do you expect additive measures only?

Case Scenario 2: Extending the POC Project

After you implemented the POC sales data warehouse in Case Scenario 1, your customer was very satisfied. In fact, the business would like to extend the project to a real, long-term data warehouse. However, when interviewing analysts, you also discovered some points of dissatisfaction.

Interviews

Here's a list of company personnel who expressed some dissatisfaction during their interviews, along with their statements:

- **Sales SME** "I don't see correct aggregates over regions for historical data."
- **DBA Who Creates Reports** "My queries are still complicated, with many joins."

You need to solve these issues.

Questions

1. How would you address the Sales SME issue?
2. What kind of schema would you implement for a long-term DW?
3. How would you address the DBA's issue?

Suggested Practices

To help you successfully master the exam objectives presented in this chapter, complete the following tasks.

Analyze the AdventureWorksDW2012 Database Thoroughly

To understand all kind of dimensions and fact tables, you should analyze the AdventureWorksDW2012 sample database thoroughly. There are cases for many data warehousing problems you might encounter.

- **Practice 1** Check all fact tables. Find all semi-additive measures.
- **Practice 2** Find all hierarchies possible for the *DimCustomer* dimension. Include attributes in the dimension and attributes in the lookup *DimGeography* table.

Check the SCD and Lineage in the AdventureWorksDW2012 Database

Although the AdventureWorksDW2012 database exemplifies many cases for data warehousing, not all possible problems are covered. You should check for what is missing.

- **Practice 1** Is there room for lineage information in all dimensions and fact tables? How would you accommodate this information?
- **Practice 2** Are there some important dimensions, such as those representing customers and products, that are not prepared for a Type 2 SCD solution? How would you prepare those dimensions for a Type 2 SCD solution?

Answers

This section contains answers to the lesson review questions and solutions to the case scenarios in this chapter.

Lesson 1

1. **Correct Answers: A and D**
   - **A. Correct:** A Star schema typically has fewer tables than a normalized schema.
   - **B. Incorrect:** The support for data types depends on the database management system, not on the schema.
   - **C. Incorrect:** There are no specific Transact-SQL expressions or commands for Star schemas. However, there are some specific optimizations for Star schema queries.
   - **D. Correct:** The Star schema is a de facto standard for data warehouses. It is narrative; the central table—the fact table—holds the measures, and the surrounding tables, the dimensions, give context to those measures.
2. **Correct Answer: C**
   - **A. Incorrect:** The Star schema is more suitable for long-term DW projects.
   - **B. Incorrect:** A normalized schema is appropriate for OLTP LOB applications.
   - **C. Correct:** A Snowflake schema is appropriate for POC projects, because dimensions are normalized and thus closer to source normalized schema.
   - **D. Incorrect:** An XML schema is used for validating XML documents, not for a DW.
3. **Correct Answers: B and D**
   - **A. Incorrect:** Lookup tables are involved in both Snowflake and normalized schemas.
   - **B. Correct:** Dimensions are part of a Star schema.
   - **C. Incorrect:** Measures are columns in a fact table, not tables by themselves.
   - **D. Correct:** A fact table is the central table of a Star schema.

Lesson 2

1. **Correct Answer: B**
   - **A. Incorrect:** This is Type 3 SCD management.
   - **B. Correct:** This is how you handle changes when you implement a Type 2 SCD solution.
   - **C. Incorrect:** Maintaining history does not mean that the content of a DW is static.
   - **D. Incorrect:** This is Type 1 SCD management.

2. **Correct Answer: B**
    - **A. Incorrect:** Attributes are part of dimensions.
    - **B. Correct:** Measures are part of fact tables.
    - **C. Incorrect:** Keys are part of dimensions.
    - **D. Incorrect:** Member properties are part of dimensions.
    - **E. Incorrect:** Name columns are part of dimensions.
3. **Correct Answer: B**
    - **A. Incorrect:** You need to analyze the attribute names and content in order to spot the hierarchies in a Star schema.
    - **B. Correct:** Lookup tables for dimensions denote natural hierarchies in a Snowflake schema.
    - **C. Incorrect:** A Snowflake schema supports hierarchies.
    - **D. Incorrect:** You do not need to convert a Snowflake to a Star schema to spot the hierarchies.

Lesson 3

1. **Correct Answer: C**
    - **A. Incorrect:** You can use SUM aggregate functions for semi-additive measures over the *Customer* dimension.
    - **B. Incorrect:** You can use SUM aggregate functions for semi-additive measures over the *Product* dimension.
    - **C. Correct:** You cannot use SUM aggregate functions for semi-additive measures over the *Date* dimension.
    - **D. Incorrect:** You can use SUM aggregate functions for semi-additive measures over the *Employee* dimension.
2. **Correct Answers: A and D**
    - **A. Correct:** Prices are not additive measures.
    - **B. Incorrect:** Debit is an additive measure.
    - **C. Incorrect:** Amounts are additive measures.
    - **D. Correct:** Discount percentages are not additive measures.
    - **E. Incorrect:** Balances are semi-additive measures.

3. **Correct Answer: D**
    - **A. Incorrect:** Lineage columns can be part of a fact table.
    - **B. Incorrect:** Measures are included in a fact table.
    - **C. Incorrect:** A fact table includes key columns.
    - **D. Correct:** Member property is a type of column in a dimension.

Case Scenario 1

1. For a quick POC project, you should use the Snowflake schema.
2. You would have customer, product, and date dimensions.
3. No, you should expect some non-additive measures as well. For example, prices and various percentages, such as discount percentage, are non-additive.

Case Scenario 2

1. You should implement a Type 2 solution for the slowly changing customer dimension.
2. For a long-term DW, you should choose a Star schema.
3. With Star schema design, you would address the DBA's issue automatically.

CHAPTER 9

Implementing Dynamic Packages

Exam objectives in this chapter:

- Extract and Transform Data
  - Define connection managers.
- Load Data
  - Implement package logic by using SSIS variables and parameters.

When you are developing Microsoft SQL Server Integration Services (SSIS) packages, it is a good practice to make each task or transformation as dynamic as possible. This enables you to move your packages from one environment to another (for example, from development to a test environment, and then to a production environment) without opening and changing the package. You can also configure your packages to set different properties at run time.

To do this, you can use the new features in SQL Server 2012, such as parameters and project-level connection managers, or you can use the package configurations that were first made available in the package deployment model in earlier versions of SQL Server. This chapter discusses both possibilities for designing and configuring your package to dynamically set values at run time. Using dynamic packages eliminates the need to make changes as you move from one environment to the other or to open the project when you want to run your packages using different property or variable values.

Lessons in this chapter:

- Lesson 1: Package-Level and Project-Level Connection Managers and Parameters
- Lesson 2: Package Configurations

Before You Begin

To complete this chapter, you must have:

- Basic knowledge of SQL Server 2012 SISS control flow and data flow features and components.
- Experience working with SQL Server 2012 Management Studio (SSMS).
- Experience working with SQL Server Data Tools (SSDT) or SQL Server Business Intelligence Development Studio (BIDS).
- An understanding of the basics of file copying and security.
- Knowledge of system environment variables.

Lesson 1: Package-Level and Project-Level Connection Managers and Parameters

In SQL Server 2012, SSIS introduces parameters and project-level connection managers. Parameters enable you to assign values to properties within packages at the time of package execution. Project-level connection managers allow you to define the data source connection once and use it in all packages that are part of the project. Both features are available only to projects developed for the project deployment model. This means that SSIS packages created with prior versions of SQL Server have to be upgraded to the new project deployment model if you want to use these new features.

Both functionalities are a replacement for the package configurations used in previous versions of SQL Server. In SQL Server 2012, SSIS also takes advantage of *build configurations* from Microsoft Visual Studio, which enable you to define multiple configurations and set the values of your parameters per configuration. This makes it easier to debug and deploy packages in SQL Server Data Tools against different SSIS servers.

After this lesson, you will be able to:

- Understand the difference between package-level and project-level connection managers.
- Implement parameters.
- Use property expressions to make your packages dynamic.

Estimated lesson time: 30 minutes

Using Project-Level Connection Managers

Project-level connection managers allow you to set up a connection manager on the project level. This means that you can define a connection for the whole project and use it in all packages that are part of the project. In prior versions of SQL Server, the connections were always contained within each package.

To create a project-level connection, define a new connection manager by right-clicking Connection Managers in Solution Explorer under your project and selecting the New Connection Manager option. An alternative method is to convert an existing package-level connection to a project-level connection. You can do this by opening the package, right-clicking the connection you want to convert in the Connection Managers window, and selecting Convert To Project Connection, as shown in Figure 9-1.

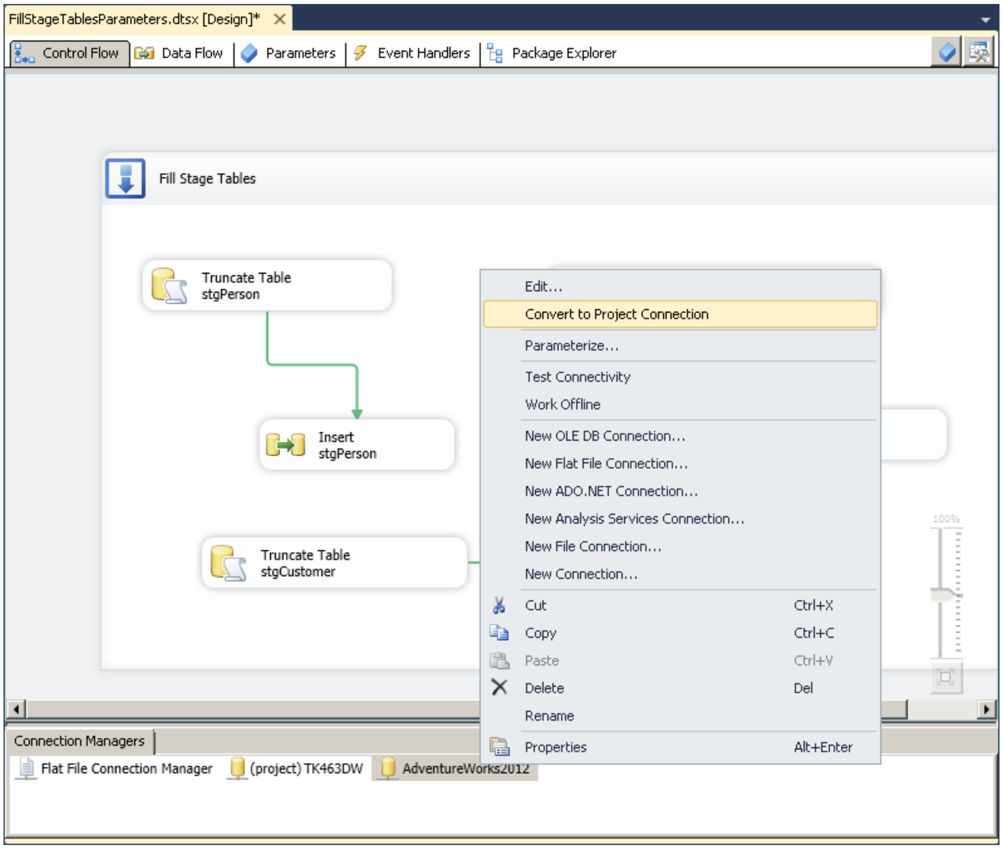


FIGURE 9-1 Converting a package-level connection to a project-level connection.

Project-level connections are a very useful new feature when it comes to developing and maintaining your packages. Note that in cases when you need to have a connection available only for the specific package, you can still create a connection at the package scope.

Parameters

In SQL Server 2012, SSIS introduces parameters. Parameters allow you to assign values to properties within packages at the time of package execution. They can also be used in SSIS expressions—for example, to use an Expression task to set a variable value based on the specific value of the parameter. There are two types of parameters: project parameters, which are created on the project level, and package parameters, which are created at the package level. You can use and set parameters only in projects developed for the project deployment model. When you are using the project deployment model, projects are deployed to the *Integration Services catalog*. Details regarding deployment possibilities are explained later in Chapter 11, "Installing SSIS and Deploying Packages."

Using Parameters

You can use a single parameter to assign a value to multiple package properties, and you can use a parameter in multiple SSIS expressions. Depending on where the project is in the project deployment life cycle, you can assign up to three different types of values to each parameter. The three types are listed in the order in which they can be applied to the parameter:

- **Design default value** The default value assigned when the project is created or edited in SQL Server Data Tools (SSDT). This value persists with the project.
- **Server default value** The default value assigned during project deployment or later, while the project resides in the SSIS catalog. This value overrides the design default.
- **Execution value** The value that is assigned in reference to a specific instance of package execution. This assignment overrides all other values but applies to only a single instance of package execution.

Note that the last two types of values become relevant after the SSIS project has been deployed to the Integration Services catalog.

Defining Parameters

You add project or package parameters by using SSDT. Usually you create a project parameter by selecting Project.params in Solution Explorer under your project name, as shown in Figure 9-2. To add a package parameter, you must first open the package and then select the parameters tab in the package design area.

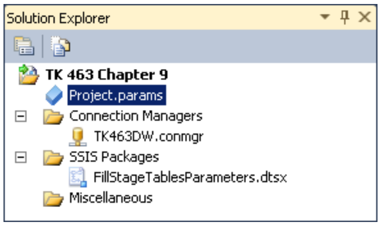
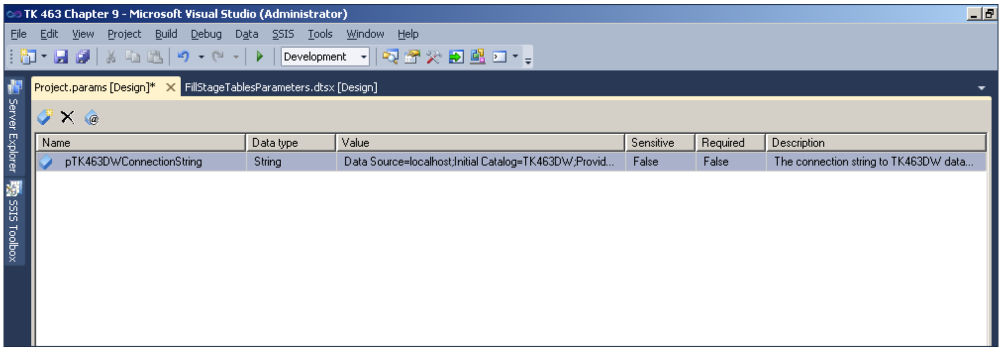


FIGURE 9-2 Solution Explorer with Project.params selected.

When you open the package or project parameters window, you can define a new parameter by clicking the Add Parameter icon (the first icon on the left; it looks like a blue cube). Figure 9-3 shows a parameter window with one parameter called pTK463DWConnectionString.


TK 463 Chapter 9 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Data SSIS Tools Window Help
Development
Project.params [Design]*
FillStageTablesParameters.dtsx [Design]
Server Explorer
SSIS Toolbox

Name	Data type	Value	Sensitive	Required	Description
pTK463DWConnectionString	String	Data Source=localhost;Initial Catalog=TK463DW;Provid...	False	False	The connection string to TK463DW data...

FIGURE 9-3 The parameters window.

When you create a parameter in SSDT, there are several properties to specify:

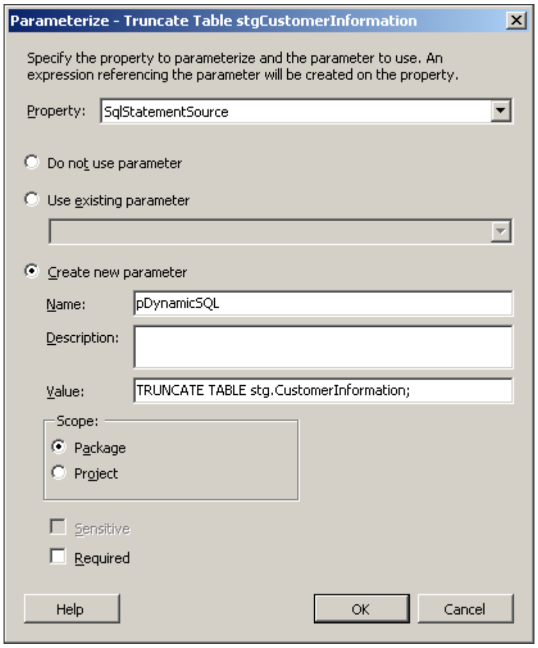
- **Name** The name of the parameter. The first character of the name must be a letter or an underscore.
- **Data type** The data type of the parameter.
- **Value** The default value for the parameter assigned at design time. This is also known as the design default.
- **Sensitive** If the value of this property is True, parameter values are encrypted in the catalog and appear as NULL when viewed with Transact-SQL or SQL Server Management Studio.
- **Required** Requires that a value other than the design default be specified before the package can be executed.
- **Description** For maintainability, the description of the parameter. In SSDT, set the parameter description in the Visual Studio Properties window when the parameter is selected in the applicable parameters window.

EXAM TIP

Parameter design values are stored in the project file.

You can edit parameters in the list in the parameter window, or you can use the Properties window to modify the values of parameter properties. You can delete a parameter by using the Delete Parameter toolbar button. By using the Add Parameters To Configurations toolbar button (the last button on the right), you can specify a value for a parameter for a specific build configuration that is used only when you execute the package in SQL Server Data Tools. Build configurations are explained in the next topic.

Another approach to creating a parameter is to do so implicitly by right-clicking a control flow task or a connection manager and selecting the Parameterize option. A Parameterize dialog box opens, as shown in Figure 9-4. Here you can specify which property should be dynamically evaluated at run time and whether the parameter should be created (the latter is specified by selecting the Create New Parameter option).


Parameterize - Truncate Table stgCustomerInformation

Specify the property to parameterize and the parameter to use. An expression referencing the parameter will be created on the property.

Property: SqlStatementSource

Do not use parameter

Use existing parameter

[✓ Create new parameter]

Name: pDynamicSQL

Description:

Value: TRUNCATE TABLE stg.CustomerInformation;

Scope: [✓ Package] Project

Sensitive

Required

FIGURE 9-4 The Parameterize dialog box.

EXAM TIP

You cannot change the value of a parameter while a package is running.

Build Configurations in SQL Server 2012 Integration Services

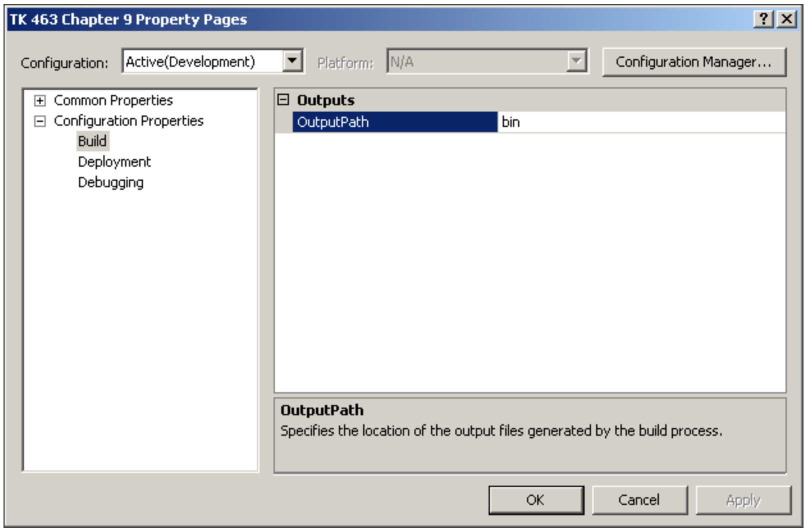
Build configurations in Visual Studio provide a way to store multiple versions of solution and project properties. The active configuration can be quickly accessed and changed, allowing you to easily build multiple configurations of the same project. Two levels of build configurations can be defined in Visual Studio: solution configurations and project configurations.

The new project deployment model in SQL Server 2012 takes advantage of more possibilities for build configurations from Visual Studio than earlier versions did. You can now set project and package parameters to get values from build configurations. Also, some of the project-level properties can be set via build configurations (for example, deployment server name and server project path).

Creating Build Configurations

A project can have a set of defined project properties for every unique combination of a configuration and platform. You can create an additional project or solution configuration by using the Configuration Manager:

1. Select the Configuration Manager option from the Configurations drop-down list on the Standard toolbar to open the Configuration Manager dialog box. Alternatively, you can first open the project's Property Pages dialog box (right-click the project name in Solution Explorer and select Properties), as shown in Figure 9-5, and click the Configuration Manager button.



OutputPath bin

OutputPath
Specifies the location of the output files generated by the build process.

FIGURE 9-5 The project Property Pages dialog box.

2. In the Active Solution Configuration drop-down list, select New.
3. In the New Solution Configuration dialog box, enter the name of the solution configuration you would like to create. Select the Create New Project Configurations check box to also create the project configuration.

You can now change the active configuration directly from the Solution Configurations drop-down list on the Standard toolbar or from the Configuration Manager dialog box.

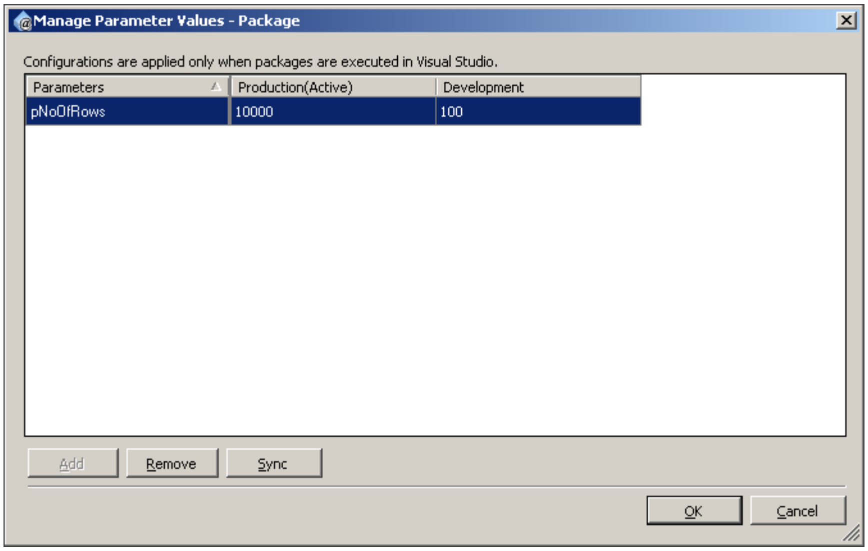
Using Build Configurations

You can bind parameter values to build configurations by using the parameter window:

1. Open the project or package parameters window.
2. In the parameters window, select the last toolbar button on the left (Add Parameters To Configurations).

3. In the Manage Parameter Values dialog box that appears, you can add values for each parameter for each configuration. Click the Add button to add a parameter to the configuration.
4. In the Add Parameters window, select parameters and click OK.
5. Set the appropriate values of parameters for your configurations, as shown in Figure 9-6.

Figure 9-6 shows that the pNoOfRows parameter will have a value of 10,000 when the package is executed using the Production configuration and 100 when using the Development configuration. This means that if you have multiple parameters and need to change the value of the parameters at design time when you are developing or debugging the SSIS project, you just need to switch between build configurations to have all parameter values changed at once.

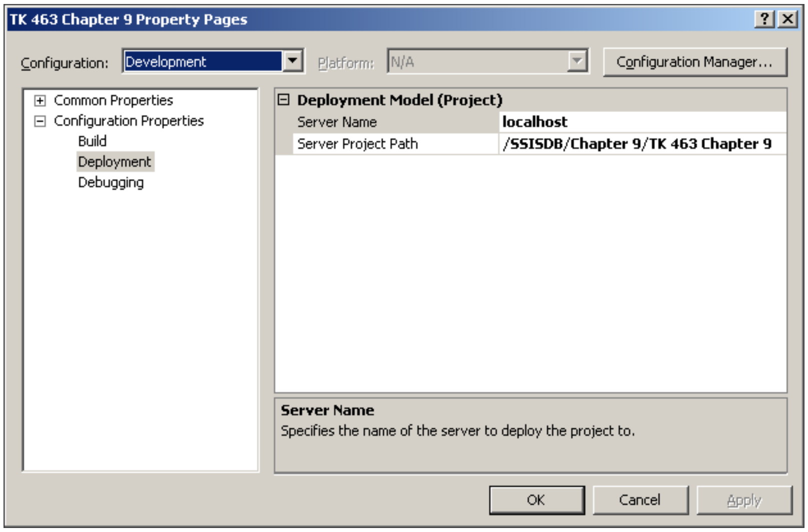


Configurations are applied only when packages are executed in Visual Studio.

Parameters	Production(Active)	Development
pNoOfRows	10000	100

FIGURE 9-6 Managing parameter values.

You can also use build configurations to set build, deployment, and debugging configuration properties for a project. To assign different project properties based on configuration, right-click the project in Solution Explorer and select Properties. Figure 9-7 shows the deployment properties inside the project's Property Pages dialog box. If you have to deploy your project to multiple servers, you can set different values for the Server Name and Server Project Path properties for each configuration, so that you can quickly switch between deployment environments at design time by using configurations.



Deployment Model (Project)

Server Name	localhost
Server Project Path	/SSISDB/Chapter 9/TK 463 Chapter 9

Server Name
Specifies the name of the server to deploy the project to.

FIGURE 9-7 Deployment properties in a project's Property Pages dialog box.

Property Expressions

In previous chapters, you saw some examples of setting connection managers or control flow task properties at run time. The SSIS expressions used to update properties of the control flow during package execution are called *property expressions*. You can apply a property expression in two ways. First, you can set the property as an expression through the properties window by clicking the ellipsis (…) button of the Expressions property. This will open the Property Expression Editor, as shown in Figure 9-8. In this dialog box, you can select a property from the drop-down list and then type an expression.

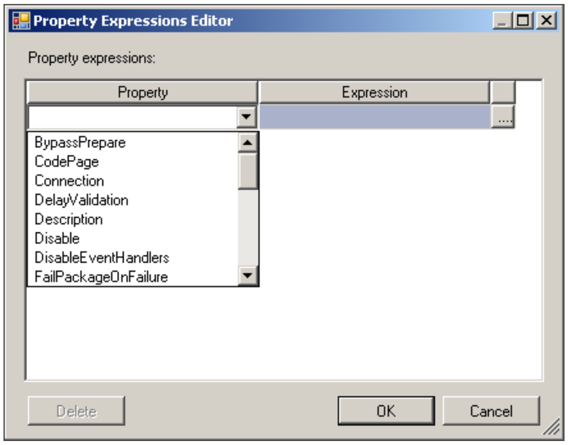


FIGURE 9-8 The Property Expressions Editor.

The second way to set property expressions is to use the task or container editors, which offer one or more ways to set properties through expressions. Figure 9-9 shows the Execute SQL Task Editor. On the Expressions tab, you can specify property expressions.

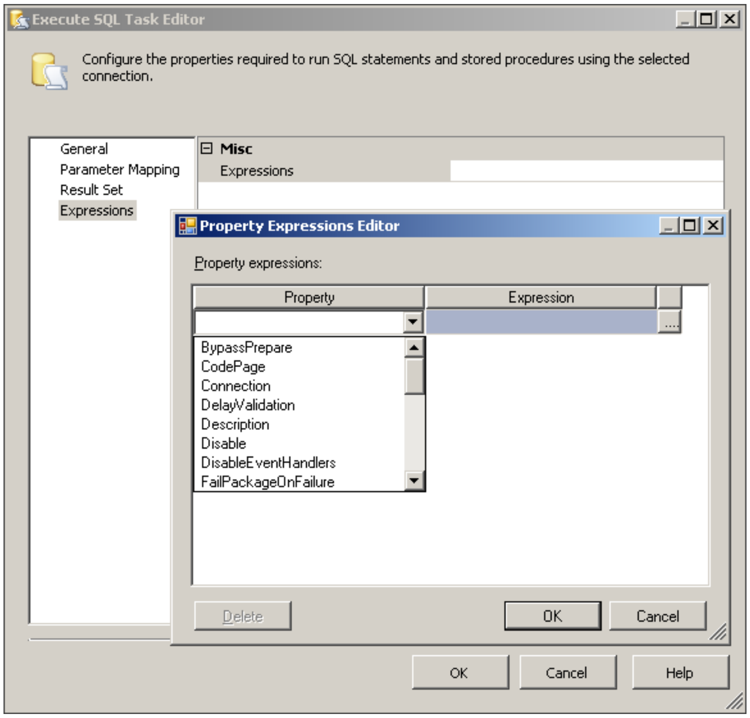


FIGURE 9-9 The Property Expressions Editor opened from the Expressions tab.

Quick Check

- When are property expressions evaluated as a package is running?

Quick Check Answer

- Unlike parameters that are read at the start of package execution, property expressions are updated when the property is accessed by the package during package execution. A property expression can change the value of a property in the middle of package execution, so that the new value is read when the property is needed by the package.

EXAM TIP

You can also update the value of connection properties while a package is running. This capability is especially useful when you have a Foreach Loop that is iterating over files. You can use a variable to indicate that the full path is being captured and update the connection for the file with a property expression. Remember that you cannot use a parameter in this case to store the full file path, because parameter values cannot change while a package is running.

PRACTICE Implementing Parameters

In this practice, you will use parameters to make your packages dynamic. In the first exercise you will parameterize the connection string, and in the second exercise you will use a parameter value to filter the source query in the data flow task. The third exercise focuses on setting up an additional build configuration to test project execution against another database.

If you encounter a problem completing an exercise, you can install the completed projects from the Solution folder that is provided with the companion content for this chapter and lesson.

EXERCISE 1 Set a Parameter for a Connection String

In this exercise, you parameterize a connection string by using a project parameter.

1. If you are missing the database objects from Chapter 5, "Designing and Implementing Data Flow," execute the needed SQL code from that chapter to have all the stage and dimension tables available in the TK463DW database.
2. Start SQL Server Data Tools, open the TK 463 Chapter 9 project in the Starter folder, and then open the FillStageTablesParameters.dtsx package.

   Notice that this package is using two connections, AdventureWorks2012 and TK463DW.
3. In the Connection Managers window, right-click the TK463DW connection and select Convert To Project Connection. You have changed the connection from the package level to the project level and can now see this connection in the Solution Explorer window in the Connection Managers folder. The TK463DW connection can now be used by any other package within the same SSIS project.
4. Right-click Project.params in Solution Explorer, and then click Open or double-click Project.params to open it.
5. Click the Add Parameter button on the toolbar.
6. Name the parameter by setting the Name property to **pTK463DWConnString** and set the Data Type property to String. In the Value property field, type **Data Source=localhost;Initial Catalog=TK463DW;Provider=SQLNCLI11.1;Integrated Security=SSPI;**.

7. Close the Project.params window.
8. Inside the FillStageTablesParameters package, right-click the (project) TK463DW connection in the Connection Managers window and select Parameterize.
9. In the Parameterize dialog box, select the Use Existing Parameter check box and select the pTK463DWConnString parameter in the drop-down list. Notice that the project parameter name starts with $Project::. Click OK to close the Parameterize dialog box.
10. Look at the (project) TK463DW connection in the Connection Managers window and notice a small icon next to it reflecting that this connection is parameterized.

EXERCISE 2 Use a Parameter in the Data Flow Task

In this exercise, you create a package-level parameter that will be used to filter source data. You will use this parameter in the data flow task to filter only rows from the source for which the year of the modified date is equal to or greater than the parameter value.

1. If necessary, start SQL Server Data Tools, open the TK 463 Chapter 9 project, and then open the FillStageTablesParameters.dtsx package from the previous exercise for editing.
2. Select the Parameters tab and click the Add Parameter button on the toolbar.
3. Name the parameter by setting the Name property to **pYear** and set the Data Type property to Int16. For the Value property, enter **2002**.
4. Click the Data Flow tab and open the Person OLE DB Source adapter.
5. In the OLE DB Source Editor, change the data access mode to SQL Command and enter the following SELECT statement to retrieve the necessary rows and use a parameter placeholder inside the query.

```
SELECT
  BusinessEntityID,
  PersonType,
  NameStyle,
  Title,
  FirstName,
  MiddleName,
  LastName,
  Suffix,
  EmailPromotion,
  AdditionalContactInfo,
  Demographics,
  rowguid,
    ModifiedDate
FROM
  Person.Person
WHERE
  YEAR(ModifiedDate) >= ?
```

6. Click the Parameters button to open the Set Query Parameters dialog box.

7. For the Variables property, select the $Package::pYear parameter as the source for the query parameter. Click OK twice to close the window and the OLE DB Source Editor.

8. Execute the package and observe the number of rows displayed in the data flow area.

9. Change the parameter value to **2008** and execute the package. Notice that fewer rows are read from the OLE DB Source adapter in the data flow area.

EXERCISE 3 Use Build Configurations

In this exercise, you create an additional database, TK463DWProd, and then create a new build configuration in Visual Studio to use this database when running the SSIS package build from the previous exercise.

1. Start SSMS and connect to your SQL Server instance. Open a new query window by clicking the New Query button.

2. You will create a new database and the *stg.Person* table so that you can execute the SSIS package created in the previous exercise. Execute the provided T-SQL code to create the database and the table.

```
USE master;
IF DB_ID('TK463DWProd') IS NOT NULL
  DROP DATABASE TK463DWProd;
GO
CREATE DATABASE TK463DWProd
 ON PRIMARY
 (NAME = N'TK463DWProd', FILENAME = N'C:\TK463\TK463DWProd.mdf',
  SIZE = 307200KB , FILEGROWTH = 10240KB )
 LOG ON
 (NAME = N'TK463DWProd_log', FILENAME = N'C:\TK463\TK463DWProd_log.ldf',
  SIZE = 51200KB , FILEGROWTH = 10%);
GO
ALTER DATABASE TK463DWProd SET RECOVERY SIMPLE WITH NO_WAIT;
GO
USE TK463DWProd;
GO
CREATE SCHEMA stg AUTHORIZATION dbo;
GO
CREATE TABLE stg.Person
(
 BusinessEntityID INT          NULL,
 PersonType       NCHAR(2)     NULL,
 Title            NVARCHAR(8)  NULL,
 FirstName        NVARCHAR(50) NULL,
 MiddleName       NVARCHAR(50) NULL,
 LastName         NVARCHAR(50) NULL,
 Suffix           NVARCHAR(10) NULL,
 ModifiedDate     DATETIME     NULL
);
```

3. If necessary, start SQL Server Data Tools, open the TK 463 Chapter 9 project, and then open the FillStageTablesParameters.dtsx package from the previous exercise for editing.

4. Select the Configuration Manager option in the Solution Configurations drop-down list.
5. In the Configuration Manager dialog box, select the New option in the Active Solution Configuration drop-down list.
6. In the New Solution Configuration dialog box, enter **Production** as the name of the configuration and click OK. Close the Configuration Manager dialog box.
7. Right-click Project.params in Solution Explorer, and then click Open (or double-click Project.params).
8. Click the Add Parameter To Configurations button on the toolbar.
9. In the Manage Parameter Values dialog box, click Add. Select the pTK463DWConnString parameter and click OK. Notice that the value of the parameter was copied to both configurations.
10. Change the Production configuration to use the newly created database, TK463DWProd. The value should look like this—**Data Source=localhost;Initial Catalog=TK463DWProd;Provider=SQLNCLI11.1;Integrated Security=SSPI;**.
11. Click OK and save the SSIS project to store the values for the configurations.
12. Execute the package first under the Development configuration and then under the Production configuration by selecting the different values in the Solution Configurations drop-down list. Look at the *stg.Person* table in the TK463DWProd database to see if it contains data.

NOTE **CONTINUING WITH PRACTICES**

Do not exit SSMS or SSDT if you intend to continue immediately with the next practice.

Lesson Summary

- Use parameters to set up connection properties at run time.
- Parameters and project-level connection mangers can only be used with the new project deployment model introduced with SSIS in SQL Server 2012.
- Use property expressions to change the control flow properties at run time.

Lesson Review

Answer the following questions to test your knowledge of the information in this lesson. You can find the answers to these questions and explanations of why each answer choice is correct or incorrect in the "Answers" section at the end of this chapter.

1. Which parameter types are available in SSIS in SQL Server 2012? (Choose all that apply.)
   - **A.** Project-level parameters
   - **B.** Solution-level parameters
   - **C.** Control flow–level parameters
   - **D.** Package parameters
2. Which properties can be set by using build configurations? (Choose all that apply.)
   - **A.** Parameter values
   - **B.** Variable values
   - **C.** Control flow task properties
   - **D.** The Deployment Server Name property
3. Which properties can be set by using property expressions? (Choose all that apply.)
   - **A.** SQL statement for the Execute SQL task
   - **B.** Variable values
   - **C.** Data flow task properties
   - **D.** The Lookup transformation SqlCommand property

Lesson 2: Package Configurations

In versions of SSIS before SQL Server 2012, you had to use package configurations to update properties, variable values, and connections at run time. You could have the package look to an external source for configuration information that changed the settings within the package when it executed. Package configurations are optional, but in real-life scenarios they are almost mandatory, because they provide a way for you to update package settings without having to open each package in Business Intelligence Development Studio (BIDS) or SSDT. For example, by using package configurations, you can maintain connection strings and variable settings for all of your packages in a single location.

After this lesson, you will be able to:

- Implement package configurations.

Estimated lesson time: 40 minutes

Implementing Package Configurations

When you are executing a package in a package deployment model, the first action the package takes is to look at its configurations and overwrite the package's current settings with the new settings from the configurations. Common elements that are configured by using package configurations are:

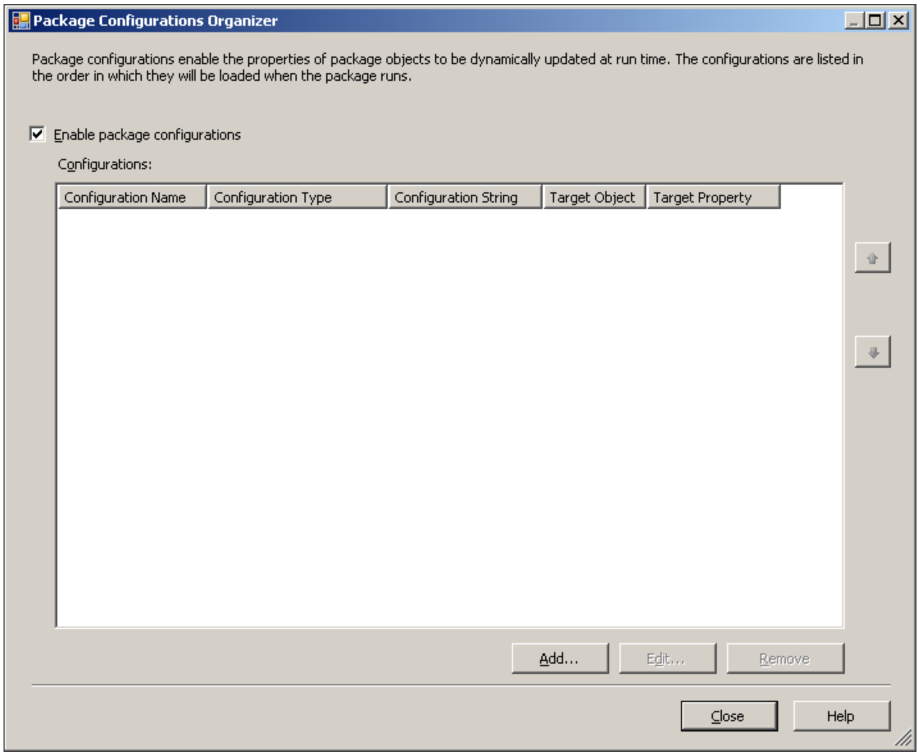
- **Connection properties** These include properties that set the connection string, the server name, the user name, and the password.
- **Package variable properties** You can set variable values, variable descriptions, and the Raise Change Event property.
- **Package properties** These include any property on the package level, such as checkpoint and security settings.

Before you can enable package configuration, you must convert your SSIS project to the package deployment model. By default, a new package in the SQL Server 2012 version of SSIS is set up for the project deployment model. You can change this by selecting Convert To Package Deployment Model under the project's name on the main toolbar. Note that you can convert only projects that do not have any parameters or project-level connection managers defined.

By default, each package has its package configuration turned off. To enable and set up configurations, you use the Package Configuration Organizer, with which you can perform the following tasks:

- Enable or disable a package's package configurations
- Add or remove configurations assigned to the package
- Define the order in which the configurations are applied

To open the Package Configurations Organizer, open the package for which you want to turn on configurations, and then choose SSIS Configurations from the SSIS menu. To enable configurations, select the Enable Package Configurations check box at the top of the dialog box. Figure 9-10 shows the Package Configurations Organizer dialog box with package configurations enabled.


Package Configurations Organizer

Package configurations enable the properties of package objects to be dynamically updated at run time. The configurations are listed in the order in which they will be loaded when the package runs.

☑ Enable package configurations

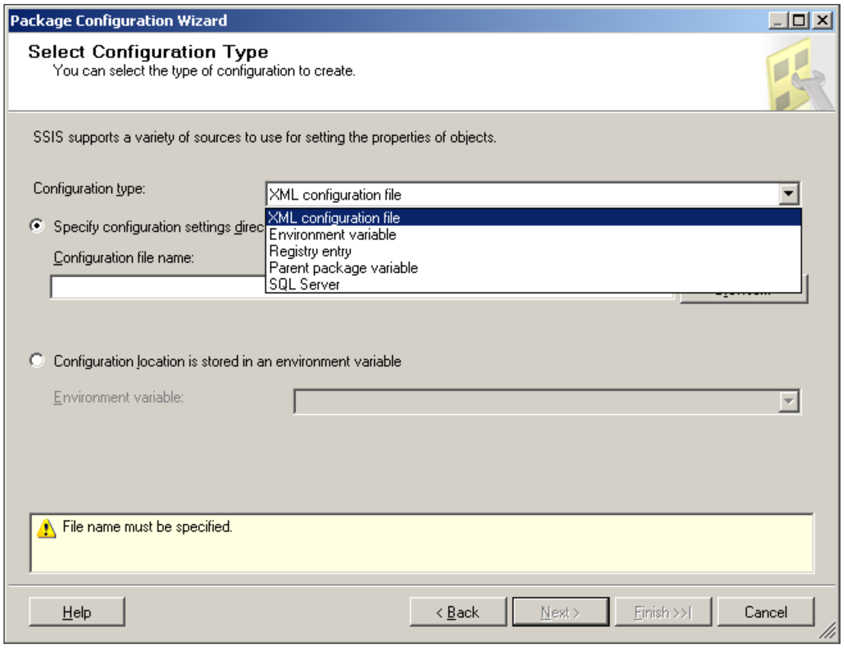
Configurations:

Configuration Name	Configuration Type	Configuration String	Target Object	Target Property

FIGURE 9-10 The Package Configurations Organizer.

Creating a Configuration

To create a new configuration, click the Add button in the Package Configurations Organizer dialog box to start the Package Configuration Wizard. First you must specify the configuration type by selecting the appropriate value from the drop-down list, as shown in Figure 9-11. SSIS supports different package configuration types; Table 9-1 describes the configuration types you can use.


Package Configuration Wizard

Select Configuration Type

You can select the type of configuration to create.

SSIS supports a variety of sources to use for setting the properties of objects.

Configuration type: XML configuration file

- XML configuration file
- Environment variable
- Registry entry
- Parent package variable
- SQL Server

(●) Specify configuration settings direc

Configuration file name:

( ) Configuration location is stored in an environment variable

Environment variable:

File name must be specified.

FIGURE 9-11 Selecting a configuration type by using the Package Configuration Wizard.

TABLE 9-1 Package Configuration Types

Type	Description
XML Configuration File	Stores configuration settings in an XML file in the file system. Use this option if you are comfortable working with configuration files and your project requirements let you store configuration information in a file system file. Note that you can store multiple configurations in a single XML file.
Environment Variable	Saves the configuration information inside the system's global variables collection, which is called an *environment variable*. Only one property can be stored in each Environment Variable configuration.
Registry Entry	Lets you save package properties and settings in your computer's registry.
Parent Package Variable	Provides a way for your package to inherit the value of a variable from a parent package. When a package is executed from another SSIS package by using the Execute Package task, the values of its variables are available to the child package through the Parent Package Variable configuration. With this configuration type, you can choose only one package property setting at a time.
SQL Server	Stores configuration settings in a SQL Server table. You can store multiple configurations in a single table.

EXAM TIP

In a package deployment model, you can pass a variable value from one package to another only by using package configurations and selecting the Parent Package Variable package configuration type.

Choose the most appropriate configuration for your environment and your project requirements. Ensure that you consider how the package will be supported in a production environment and how other technologies are supported and configured. Take care to evaluate any security and compliance requirements when you are storing connection information such as server name, user name, or password information.

IMPORTANT **PARAMETERS OR PACKAGE CONFIGURATIONS?**

If you are using SQL Server 2012, use the new project deployment model with parameters and project-level connection managers to support moving your solution from one environment to another. These new features provide better package management and flexibility in package development compared to package configurations. These new features are positioned as an evolution of Integration Services deployment and administration in SQL Server.

The most commonly used configuration types are the XML Configuration File and SQL Server configurations. The next section looks more closely at each of these types.

Creating an XML File Configuration

When you choose the XML Configuration File type, you can specify the location for your configuration file. There are two ways to specify the location of the XML file:

- Enter the location of the file directly in the Configuration File Name text box. Use this when you intend to always use the same location and name for your configuration file.
- Use an environment variable that contains the location and name of the configuration file. To use this approach, you must create a system variable in your computer's system properties. The value of the variable must contain the full path, name, and extension of the file.

  Using an environment variable for the file location pointer is called the *indirect file location approach* and is very valuable if your XML file location or file name might change in the future or already changes between environments. If you choose to use the environment variable, be sure to add it to the servers on which the package will run.

EXAM TIP

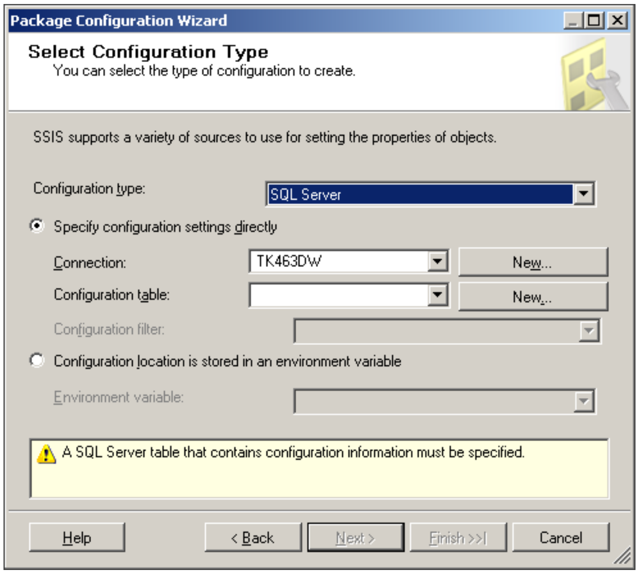
Indirect configurations are useful when the location of the file changes between the development and the production server. To use the indirect configuration, you first need to create the file by using the wizard and then go back and edit the configuration to assign the environment variable.

As with all of the configuration types, more than one package can use the same XML Configuration File. If you have several packages that have common properties, such as connection strings, you might want to have all of them use one XML file for configuration.

After you have defined the location and name of the file, you define the server settings and properties that the XML Configuration File should contain. Because these are common among all configuration types, this chapter reviews the SQL Configuration setup before describing the server settings and property definitions.

Creating a SQL Server Configuration

To store your package configurations in a SQL Server table, select SQL Server from the Configuration Type drop-down list in the Package Configuration Wizard. Using SQL Server as the storage mechanism for your configurations requires a different group of settings than those used by the other configuration types, such as XML Configuration File. Figure 9-12 shows the SQL Server configuration options available for setting up configurations.



Package Configuration Wizard

Select Configuration Type

You can select the type of configuration to create.

SSIS supports a variety of sources to use for setting the properties of objects.

Configuration type: SQL Server

(●) Specify configuration settings directly

Connection: TK463DW

Configuration table:

Configuration filter:

( ) Configuration location is stored in an environment variable

Environment variable:

A SQL Server table that contains configuration information must be specified.

FIGURE 9-12 The Package Configuration Wizard for a SQL Server table configuration.

Just as with the XML Configuration File type, you can specify an environment variable as the location of your configuration (for example, the data source name for the SQL Server configuration), or you can specify configuration settings directly. There are three settings that define the table location details:

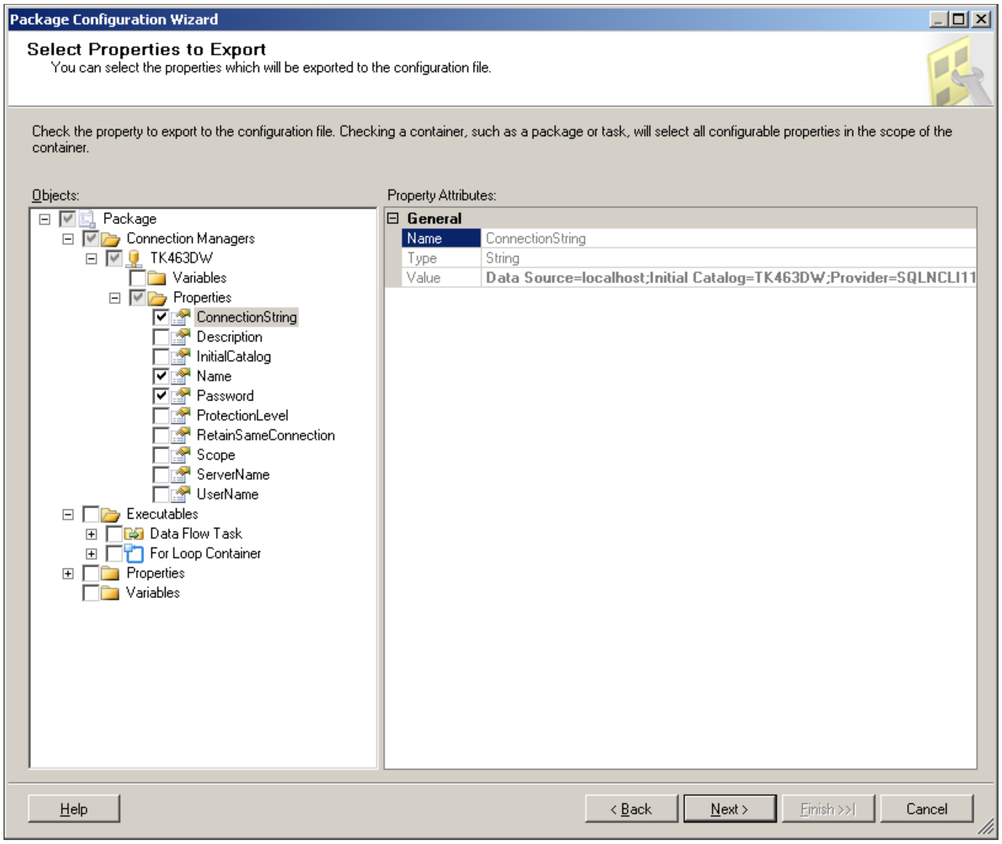
- **Connection** This must be a SQL Server–based connection that sets the server and database in which your configurations will be stored and read. If you did not define the connection you need, you can click New next to Connection to open the Configure OLE DB Connection Manager dialog box.
- **Configuration Table** This is the name of the table in which the configurations will reside. This table has predefined column name and data type definitions that cannot be changed. To create the table, you click New next to the Configuration Table text box to open the Create Table dialog box, in which you can change the name of the table and execute the table creation statement for the connection that you specified in the previous setting.
- **Configuration Filter** Multiple SQL Server configurations can share the same table, and you can specify the configuration you want by using the Configuration Filter drop-down list. You can enter a new filter or use an existing one. The name you select or enter for this property is used as a value in the Configuration Filter column in the underlying table.

Adding Properties to Your Configuration

No matter which SSIS configuration type you are using, you can select Properties To Export on the next page of the wizard to select the SSIS package and object properties that are to be used in the configuration. After you define the configuration type properties in the Package Configuration Wizard, click Next.

At this point, SSIS prompts you to verify whether configuration entries already exist for the configuration type you selected. If they do, SSIS prompts you to either reuse the configuration entries or overwrite them. If you see this dialog box, you will probably want to share the existing configurations between packages. If you do, click the Reuse Existing button. If you want to clear the existing entries and create new ones, click Overwrite.

If configuration entries do not already exist in this configuration, or if you clicked Overwrite, you will see the Select Properties To Export page, as shown in Figure 9-13.


Package Configuration Wizard

Select Properties to Export

You can select the properties which will be exported to the configuration file.

Check the property to export to the configuration file. Checking a container, such as a package or task, will select all configurable properties in the scope of the container.

Objects:

```
⊟ ☑ Package
   ⊟ ☑ Connection Managers
      ⊟ ☑ TK463DW
            ☐ Variables
         ⊟ ☑ Properties
               ☑ ConnectionString
               ☐ Description
               ☐ InitialCatalog
               ☑ Name
               ☑ Password
               ☐ ProtectionLevel
               ☐ RetainSameConnection
               ☐ Scope
               ☐ ServerName
               ☐ UserName
   ⊟ ☐ Executables
      ⊞ ☐ Data Flow Task
      ⊞ ☐ For Loop Container
   ⊞ ☐ Properties
     ☐ Variables
```

Property Attributes:

General	
Name	ConnectionString
Type	String
Value	Data Source=localhost;Initial Catalog=TK463DW;Provider=SQLNCLI11

FIGURE 9-13 The Select Properties To Export page of the Package Configuration Wizard.

The Select Properties To Export page uses a tree view structure of your package properties, allowing you to select the properties for the SSIS configuration you have selected. Properties are grouped within the following folders:

- **Variables** Lists all of the package variables you can select for configuration entries, along with their properties.
- **Connection Managers** Lists all of the package connections, from which you can choose the specific properties for your connections.
- **Log Providers** Lets you dynamically set the log configuration.
- **Properties** Displays a list of all package-level properties that you can use to configure your package.
- **Executables** Contains the tree structure of your tasks and containers. By navigating through this tree, you can configure the specific properties of your tasks and containers.

If you are using an XML Configuration File, SQL Server, or Registry Entry configuration, you can set multiple configuration properties at one time by selecting multiple property check boxes.

Sharing, Ordering, and Editing Your Configurations

If you have several configurations in your list, you can define the order in which configurations are applied in a package. The configurations are called in the order in which they are listed in the Package Configuration Organizer. This is an important consideration if you have multiple configurations that will update the same property or if you have configurations that have a dependency on prior configurations. For example, you might have a configuration that updates a connection string, which is then used as the location of the configuration entries in a second configuration. Note that the last-applied property update will be the value that is used in the package.

A common approach is to share configurations between packages. If you do this, you might have configuration entries that apply to one package and not another. This does not affect package execution, but you will receive a warning to indicate that a configuration property does not exist in the package.

As a final note, you can modify all SSIS configuration entries you have made by simply editing the file, SQL Server, registry, or environment variable value. Look for the Configured Value property and change it as necessary.

REAL WORLD **MULTIPLE CONFIGURATIONS**

In previous versions of SQL Server, the use of configurations was almost mandatory for moving from a development to a test environment and then to a production environment. Because user names and passwords for connection strings should not be stored as clear text in an XML file, most of the clients store configuration properties in SQL Server. To have the flexibility to move from one environment to another, you also need to put the information about the SQL Server instance used for storing configurations in a configuration setup. A common solution is to first use an XML configuration that contains the location of the file stored as an environment variable and that includes only the connection string property for the SQL Server configuration that will hold other configuration values. Then you create a second configuration that is a SQL Server configuration, and you use that configuration for all configuration values.

PRACTICE Using Package Configurations

In this practice, you will create an XML Configuration File and share it between two packages.

If you encounter a problem completing an exercise, you can install the completed projects from the Solution folder that is provided with the companion content for this chapter and lesson.

EXERCISE Create an XML Configuration

In this exercise, you use SSIS configurations to create an SSIS XML Configuration File that contains the connection string property of the AdventureWorks2012 and TK463 databases. You then share this configuration with another package.

1. Start SQL Server Data Tools, open the TK 463 Chapter 9 project, and open the FillStageTables_1.dtsx package.
2. Now you need to convert the project to a package deployment model. Click Project on the toolbar and select Convert To Project Deployment Model. Click OK in the dialog box.
3. In the Convert To Package Deployment Model dialog box, every step should show the Result value as Passed. Click OK to convert the project to a package deployment model.
4. Choose Package Configurations from the SSIS menu.
5. Select the Enable Package Configurations check box in the Package Configurations Organizer dialog box.
6. Click Add to create a new configuration.
7. Click Next on the Welcome To The Package Configuration Wizard page.
8. In the Configuration Type drop-down list, select XML Configuration File.
9. Click the Browse button next to the Configuration File Name box, browse to the \Chapter09\Lesson2\Starter\ installed files folder, and then type **SSIS_Conn.dtsConfig**. Click Save to save the file name and path.
10. Click Next in the Package Configuration Wizard to go to the Select Properties To Export page.
11. Under Objects, expand the Connection Managers folder, expand the Properties folder for the AdventureWorks2012 connection, and select the check box next to the ConnectionString property. Repeat the process for the TK463DW connection. Click Next.
12. Name the configuration **MainXMLConfiguration** and close the Configuration Organizer dialog box.
13. Save and close the FillStageTables_1.dtsx package.
14. Open the FillStageTables_2.dtsx package and repeat steps 4 through 9. Select the file you created in step 9, and click Next. You will be prompted to overwrite the existing file or reuse the configuration that it contains. Click the Reuse Existing button.
15. Name the configuration **MainXMLConfiguration** and close the Configuration Organizer dialog box.

16. Save and close the FillStageTables_2.dtsx package.
17. Execute the packages and try to change the XML file so that the connection string points to the TK463DWProd database created in the previous lesson, and execute the packages again.

Lesson Summary

- Package configurations are available in the package deployment model.
- Use package configurations if you are using previous versions of SSIS to set connection properties at run time.
- Use a combination of XML and SQL Server configurations to provide additional portability for your packages.

Lesson Review

Answer the following questions to test your knowledge of the information in this lesson. You can find the answers to these questions and explanations of why each answer choice is correct or incorrect in the "Answers" section at the end of this chapter.

1. Which configuration types can you use to store configuration values? (Choose all that apply.)
   - **A.** XML Configuration File
   - **B.** SQL Server
   - **C.** Any relational database system
   - **D.** Registry entry
2. On which objects can you set dynamic properties by using package configurations? (Choose all that apply.)
   - **A.** Parameters
   - **B.** Variables
   - **C.** Data flow transformations
   - **D.** The Sequence Container task
3. Which SSIS elements can be configured by using a package configuration? (Choose all that apply.)
   - **A.** Connection properties
   - **B.** Package variable properties
   - **C.** Parameter properties
   - **D.** Package properties

Case Scenario

In the following case scenarios, you apply what you've learned about implementing dynamic packages. You can find the answers to these questions in the "Answers" section at the end of this chapter.

Case Scenario: Making SSIS Packages Dynamic

You are creating a set of SSIS packages to move data from flat files and different databases to a data warehouse. Because of strict development, test, and production procedures, your SSIS project must support these possibilities:

1. You need to test different deployment options for your SSIS packages in the development phase by using SSDT.
2. You have a development, test, and production environment and would like to minimize any changes to packages when they are ready for the test phase.
3. You need to parameterize the location of the flat files but still be able to dynamically set the correct file name when using the Foreach Loop container to read all the files inside the specified folder.

How would you address these issues? What would your solution look like?

Suggested Practices

To help you successfully master the exam objectives presented in this chapter, complete the following tasks.

Use a Parameter to Incrementally Load a Fact Table

In order to practice what you have learned in this chapter, you will create a package to load a fact table that will store Internet sales data based on the AdventureWorks2012 database.

- **Practice 1** Create the necessary package to load the fact table. At each execution of the package, do the full load and truncate the table at the beginning.
- **Practice 2** Add a project parameter that will accept the incremental date and use it to load only data that is newer than the supplied value. Replace the TRUNCATE T-SQL statement at the beginning with the DELETE T-SQL statement.
- **Practice 3** Parameterize all connection strings.

Answers

This section contains answers to the lesson review questions and solutions to the case scenario in this chapter.

Lesson 1

1. **Correct Answers: A and D**
   - A. **Correct:** Project-level parameters are available.
   - B. **Incorrect:** Solution-level parameters are not available.
   - C. **Incorrect:** Parameters can be defined on the project or package level.
   - D. **Correct:** SSIS supports package-level parameters.
2. **Correct Answers: A and D**
   - A. **Correct:** Parameter values can be bound to build configurations.
   - B. **Incorrect:** Variables cannot be bound to build configurations.
   - C. **Incorrect:** Control flow task properties cannot be bound to build configurations.
   - D. **Correct:** Project-level properties can be set for each build configuration.
3. **Correct Answers: A and C**
   - A. **Correct:** You can set the SQL statement of the Execute SQL task at run time by using property expressions.
   - B. **Incorrect:** Variable properties cannot be set by using property expressions.
   - C. **Correct:** General data flow task properties can be set by using property expressions.
   - D. **Incorrect:** You can only change the SQL command of the Lookup task when you are using property expressions for the data flow task that has a Lookup task inside it.

Lesson 2

1. **Correct Answers: A, B, and D**
   - A. **Correct:** You can use XML as a configuration file.
   - B. **Correct:** You can store the configuration values in a SQL Server database.
   - C. **Incorrect:** You can only use a SQL Server database.
   - D. **Correct:** You can use registry entries to store configuration values.

2. **Correct Answers: B and D**
   - **A. Incorrect:** Parameters can be used in the project deployment model.
   - **B. Correct:** You can set variable properties.
   - **C. Incorrect:** You can only set data flow task properties, and not for a specific transformation.
   - **D. Correct:** You can dynamically set properties for a Sequence Container task by using configurations.
3. **Correct Answers: A, B, and D**
   - **A. Correct:** Connection properties can be set by using package configurations.
   - **B. Correct:** You can set variable properties.
   - **C. Incorrect:** You cannot use parameters when using package configurations.
   - **D. Correct:** You can dynamically set properties for the package by using package configurations.

Case Scenario

1. Add parameters for the connection strings. Create development, test, and production build configurations and bind parameters to each of them with different data source values. This will allow you to execute the package from SSDT against different configurations without manually changing the value of parameters.
2. Add parameters and parameterize all needed connection managers.
3. Create a parameter that will hold the value of the file location folder. Create a new variable to store the current file name. Using SSIS expression language, set the new variable value to combine the value from the parameter with the value you get from the Foreach Loop container. Use the new variable as the property expression for the Foreach Loop to dynamically change the fully qualified file name while the package is running.

Index

Symbols

32-bit data providers, 137
64-bit data providers, 137
64-bit vs. 32-bit environments
  Excel files and, 647
  in project validation, 471
  SSIS component installation, 427
@data_filename, 508
@dataflow_path_id_string, 507
@data_tap_id argument, 508
@execution_id, 507
@max_rows, 508
@task_package_path, 507

A

abstract entities, 467
abstraction, defined, 534
access permissions, 429
accounts. *See* storage accounts
accuracy, measurement of, 532
AcquireConnections method, 702, 722, 724
acting, in data mining, 669
Active Directory Domain Services, browsing, 701
Active Directory users, 617
Active Operations dialog box, 469
Active Solution Configuration, 359
adaptability, 240
adapters
  ADO Net source
    dynamic SQL, 302–304
  CDC source, 305
  data flow destination, 178–179
    defining, 184–187
  data flow source, 178–179
    adding with Source Assistant, 181–182
    defining, 180–183
  OLE DB Destination
    column mapping, 230
  OLE DB Source
    dynamic SQL, 300–302
Add Copy Of Existing Package dialog box, 113
Add Existing Item command, 407
additivity of measures, 29–31
Add New Item command, 407
Add New Source dialog box, 181
Add Parameter button, 364
Add Parameter icon, 357
Add Parameters To Configurations toolbar button, 357
Add Parameters window, 360
Add SSIS Connection Manager dialog box, 140, 141
Add SSIS Connection Manager window, 192
Add Users page, 621
Add Variable dialog box, 158, 159, 244
Add Watch, 502
administration operations (SQL Server), 132
administration tasks, 151
administrative stored procedures, accessibility of, 481
administrators, in MDS, 617
Administrators user group, 617
ADO connection manager, 134
ADO.NET connection manager, 134, 136
ADO.NET dataset consumer, Object type variable, 247
ADO.NET destination, 184–185
ADO.NET source, 180
ADO Net source adapter, dynamic SQL, 302–304
advanced configuration, SQL server installation, 428
advanced data preparation transformations, 204
Advanced Editor
  Fast Parse property, 189–190
  for Calculate Checksum dialog box, 729
  Input and Output Properties tab, 188
  specifying sorted output data, 225

Advanced Editor dialog box, 189
AdventureWorks2012 sample database
  configuring data flow source adapters, 182–183
  extracting data from views and loading into tables, 91–98
  implementing transactions, 334
AdventureWorksDW2012 sample database
  aggregating columns in tables, 59–62
  conformed dimensions, 8
  database schema, 14–15
  DimCustomer dimension, 19–20, 49
    attributes, 19–20
  DimDate dimension, 9
    attributes, 20
  dimensions
    diagrams, 24–26
  fact tables, 32–33
  lineage, 36–37
  many-to-many relationships, 31–33
  master data, 573
  normalized dimensions, 9–10
  reporting problems with normalized relational schemas, 5–7
  SCD (Slowly Changing Dimension) problem, 36–37
  Star schema, 7–10
Aggregate data flow, 398
aggregated tables, indexed views, 58–61
aggregate functions, 29–30
  AVERAGE, 29
  COUNT_BIG, 59
  LastNonEmpty, 30
  SUM, 29
Aggregate transformation, 202, 399, 400, 758
aggregation, 146
aggregation functions, 399
alerts
  for package failure, 347
  specificity of, 381
algorithms
  for data mining, 668
  hash functions
    implementing in SSIS, 291–292
aligned indexes, 72
All Connections report, 506
All Connections validation report, 469
All environment references, validation mode, 467
All Executions report, 506
All Executions validation report, 469, 470
All Operations report, 506
All Operations validation report, 469
All process mode option (CDC), 306
All Validations report, 506
All Validations validation report, 469
All with old values process mode option (CDC), 306
ALTER INDEX command, 56
Analysis Services connection manager, 134
Analysis Services Execute DDL task, 153
Analysis Services Processing task, 153
analysis services tasks, 153
analytical applications
  dimensional model, 567
  hierarchical data, 566
analytical systems, presentation quality of, 533
annotations, adding, 626
Application log providers, 384, 385
application programming interface (API), SSIS management, 423
applications
  analytical
    dimensional model, 567
    hierarchical data, 566
  LOB
    transactional data, 566
approximate merging, 737
approximate string matching, 737, 756
architecture
  ETL (extract-transform-load), 217–218
  MDS (Master Data Services), 576–577
artificial intelligence, data mining algorithms, 668
A single environment reference
  validation mode, 467
asynchronous output, 709, 718
asynchronous transformation
  blocking vs. partial-blocking, 513
  execution trees, 513–514
atomic units, 328, 395
attribute completeness, 532
Attribute groups metadata, 606
Attribute Properties button, 627
attributes, 17–18, 18–19
  Business Key, 284
  checking changes, 291–292
  Code, 590, 591
  columns, 589
    DimCustomer dimension (AdventureWorksDW2012 sample database), 19–20
  completeness of, 654

defining attribute types, 284
DimDate dimension (AdventureWorksDW2012 sample database), 20
discretizing, 17
automatic discretization, 18
domain-based, 590
entities, 590, 591
file, 590
Fixed, 284
free-form, 590
Manager, 590
Name, 590
naming conventions, 21
pivoting, 17
Type 1 SCD, 22, 284
set-based updates, 292–293
updating dimensions, 290–293
Type 2 SCD, 22, 284
set-based updates, 292–293
updating dimensions, 290–293
Type 3 SCD, 23
Attributes metadata, 606
audit data
correlating with SSIS logs, 401–402
storing, 396
Audit data flow transformation, implementing, 402–405
auditing
correlation with SSIS logs, 401–402
customized for ETL, 342, 344, 347
data lineage, 73
defined, 394
implementing, 394–395
implementing row-level, 402
Integration Services Auditing, 382
master data, 568, 569
native, 399
tables, 13
techniques, 395–402
varied levels, 412
vs. logging, 398
Audit transformation, 199
elementary auditing, 401
system variables, 400
Audit Transformation Editor, 404
authoritative sources, for de-duplication for, 738
authority, for MDM (master data mangement), 572
autogrowing, prevention of, 43
automated execution, 456, 462–464
automatic cleansing methods, 537
automatic discretization, 18
automation
adaptability, 240
determinism and, 239
predictability, 240
Auto Shrink database option, 43–45
autoshrinking, prevention of, 43
Average aggregate function, 29, 399

B

backpressure mechanism, 514
Back Up Database task, 152
balanced tree. *See* B-trees
base entities, 590
Basic logging level, 468, 506
Batch_ID column, in stg.entityname_Leaf table, 608
batch processing, 62–64
BatchTag column, in stg.entityname_Leaf table, 608
BEGIN TRAN statement, 332
BIDS (SQL Server Business Intelligence Development Studio), 102, 179–180
binary large object (BLOB), 516
BISM (Business Intelligence Semantic Model), 30–31
bitmap filtered hash joins, 57
blocking
selecting data transformations, 198–199
transformations, 203
blocking (asynchronous) transformations, 513
blocking technique, 739
blocking transformations, 199
Boolean data type, 246
Boolean expressions, in precedence contraints, 256
break conditions, 500–501
breakpoints
debugging with, 500–503
in script task, 702
B-trees (balanced trees), 56–57
Buffer Memory counter, 520
buffers, data
allocation with execution trees, 513–514
architecture of, 512–513
changing setting, 516
counters for monitoring, 520
fixed limit of, 514
optimization of, 515–516
synchronous vs. asynchronous components and, 513

Buffers in Use counters, 520
buffers, rows, 178
Buffers Spooled counter, 520
BufferWrapper, 707
build configurations, 358–361
  creating, 359
  uses for, 354, 358
  using, 359–361, 365–366
Build Model group, 627
Bulk Insert task, 150
Bulk Logged recovery model, 43
Business Intelligence Development Studio (BIDS), updating package settings in, 367
Business Intelligence Semantic Model (BISM), 30–31
business key, 23, 28–29
Business Key attribute, 284
business problems, schema completeness, 534
business rules
  data compliance, 529
  documentation of, 534
Business rules metadata, 606
Byte data type, 246

C

Cache connection manager, 218, 517
cache modes, 218–219
CACHE option, creating sequences, 46
Cache Transform transformation, 199, 223–224
Calculate Checksum dialog box, advanced editor for, 729
calculation (data), 146
Call Stack debugging window, 503
Canceled operation status, 466
Candidate Key, profiling of, 657
canonical row, in fuzzy transformations, 758
case scenarios
  batch data editing, 633
  connection manager parameterization, 125
  copying production data to development, 125
  data cleansing, 731
  data-driven execution, 277
  data flow, 232
  data warehouse administration problems, 79
  data warehouse not used, 559
  deploying packages to multiple environments, 491
  improving data quality, 660, 765
  loading fact tables, 322
  loading large dimensions, 322
  MDM solutions, 600
  MDS (Master Data Services), 600–601
  prediction improvement, 694
  remote executions, 491
  single vs. multiple packages, 276–277
  slow data warehouse reports, 79
  Snowflake schema, 34–38
  SQL Server installation, 451
  Star schema, 34–38
  strictly structured deployments, 451
  tuning SSIS packages, 523
case sensitivity, in Lookup transformation, 220
cases, in data mining, 668, 688, 689
catalog.add_data_tap stored procedure, 507
catalog.configure_catalog stored procedure, 442, 481
catalog.create_execution stored procedure, 442, 507
catalog.deploy_project stored procedure, 442
catalog.extended_operation_info stored procedure, 465
catalog.grant_permission stored procedure, 481
catalog.operation_messages catalog view, 465
catalog.operations catalog view, 465
Catalog Properties window, 506
catalog.restore_project operation, 442
catalog.revoke_permission stored procedure, 481
catalog SQL Server views, 654
catalog.start_execution stored procedure, 507
catalog.stop_operation operation stored procedure, 442
catalog.validate_package stored procedure, 442
catalog.validate_project stored procedure, 442
catalog views, 63–64
cdc.captured_columns system table, 305
CDC (change data capture), 299
  components, 305–306
  enabling on databases, 304–305
  implementing with SSIS, 304–307, 308–316
  packages
    LSN (log sequence number) ranges, 305
  processing mode options, 306
cdc.change_tables system table, 305
CDC Control task, 149, 305
CDC Control Task Editor, 306
cdc.ddl_history system table, 305
cdc.index_columns system table, 305

cdc.lsn_time_mapping system table, 305
CDCSalesOrderHeader table, 304–305
CDC source, 180
CDC source adapter, 305
CDC Source Editor dialog box, 306
CDC splitter, 305
CDC Splitter transformation, 201
cdc.stg_CDCSalesOrderHeader_CT, 305
central MDM (master data management), 571
central metadata storage, 571
  continuously merged data, 571
  identity mapping, 571
change data capture. *See* CDC (change data capture)
Changed column, on create views page, 609
Changing Attributes Updates output, 290
Chaos property, IsolationLevel, 331
Character Map transformation, 199
Char data type, 246
Check Database Integrity task, 152
CheckPointFileName property, 337, 340
checkpoints
  creating restartable, 336–339
  setting and observing, 340–341
CheckpointUsage property, 337, 339, 340
child packages, 20, 328, 330, 344
  logging, 411
  parameterization of, 269
CI (compression information) structure, 62
Class Library template, 718
Clean Logs Periodically property, 439, 470
cleansing (data), 145–147, 570, 571
cleansing projects, 646
  quick start, 639
  stages of, 648
Cleansing Project setting, 552
Cleanup method, customization of, 724
clean values, 759
Client Tools SDK, development vs. production environments, 425
closed-world assumption, 532
CLR integration, 445
clustered indexes, 56–57
clustering keys, 56
code. *See also* listings
  adding foreign keys to InternetSales table, 75
  adding partitioning column to FactInternetSales table, 74
  creating columnstore index for InternetSales table, 75
  creating partition schemes, 74
  Customers dimension, 50
  dbo.Customers dimension table, 209
  enabling CDC on databases, 304–305
  InternetSales fact table, 53
  loading data to InternetSales table, 75
  loading data to partitioned tables, 76–78
  Products dimension, 51
Code attribute, 590, 591
Code column, in stg.entityname_Leaf table, 608
code snippets, 706, 714
collections, 591
collection settings (Foreach Loop Editor), 157
Collections metadata, 606
Column Length Distribution, profiling of, 656
column mappings, in fuzzy lookup, 761
Column Mappings window, 95
Column Null Ratio, profiling of, 656
Column Pattern, profiling of, 656
columns
  aggregating in AdventureWorksDW2012 sample database tables, 59–62
  <Attribute name>, 609
  attributes, 17–18, 18–19, 589
    DimCustomer dimension (AdventureWorksDW2012 sample database), 19–20
  computed, 46
  Customers dimension, 49–50, 65
  Dates dimension, 52, 67
  dimensions, 17–18
  fact tables, 28–29, 47–48
    analyzing, 33–34
    lineage type, 29
  hash functions, implementing in SSIS, 291–292
  identity
    compared to sequences, 45–46
  in fuzzy lookup output, 758
  in stg.entityname_Leaf table, 608
  InternetSales Fact table, 67
  keys, 18
  language translation, 18
  lineage information, 18
  mapping
    OLE DB Destination adapter, 214, 230
    OLE DB Destination Editor, 187
  mappings, 95
  member properties, 18, 18–19
  name, 18

- package-level audit, 398
- Products dimension, 51, 66
- references
  - errors, 208
  - resolving, 207–208
- Remarks, 50
- third normal form, 9
- Type 2 SCD, 22
- Type 3 SCD, 23
- updating, 229–231
- Valid From, 284
- Valid To, 284

Column Statistics, profiling of, 657
columnstore indexes, 62–68
- catalog views, 63–64
- fact tables, 64
- InternetSales fact table, 68–69
- segments, 63

Column Value Distribution, profiling of, 657
column values, 400
Combine Data button, 626
combining data flows vs. isolating, 120
commands
- ALTER INDEX, 56
- OLE DB, 226
- TRUNCATE TABLE, 71
- T-SQL
  - data lineage, 73

COMMIT TRAN statement, 332
common language runtime [CLR]) code. *See* Microsoft.NET
communication, importance of, 537
compatibility, data types and, 247
complete auditing, 395, 396
Completed operation status, 466
completeness
- measurement of, 531–532
- of attributes, 654

completion constraints, 165
complex data movements, 88, 145–147
complexity, of master data, 568
compliance with theoretical models, measurement of, 534
components, parameterization of, 254
ComponentWrapper project item, 707, 711
composite domains, 638, 640
composite joins, Lookup transformations and, 220
compression. *See* data compression
compression information (CI) structure, 62
computations, elementary, 256
computed columns, 46
computedPackageID variable, 271
Computer-assisted cleansing stage, 648
Computer event property, 386
concepts, 567
conceptual schema, documentation of, 534
concrete instance, 467
Conditional Split transformation, 201, 214, 291, 694
confidence level, in data mining, 689
Configuration File Name text box, 371
Configuration Filter setting, table location, 373
Configuration Manager, creating additional projects with, 359
configuration properties
- setting multiple, 375
- storage of, 375

Configuration Table setting, 373
configuration types, for packages, 370
configuration usage, described in debugging, 499
Configured Value property, 375
Configure Error Output dialog box, 318
Configure Project dialog box, 487
configuring
- connections for SSIS deployment, 120–123
- data destination adapters, 185–186
- data source adapters, 182–183
- Flat File connection manager, 138–140
- OLE DB connection manager, 140–142

conformed dimensions, 8
connection context, of execution errors, 506
Connection Manager Conversion Confirmation dialog box, 143
Connection Manager dialog box, 141
connection manager editor, 121–122
connection managers, 133–144
- 64-bit data providers, 137
- accessing, 702
- ADO, 134
- ADO.NET, 134, 136
- Analysis Services, 134
- Cache
  - Lookup transformation, 218
- creating, 138–143
- event handlers, 345
- Excel, 134
- File, 134

file formatting, 140
Flat File, 134
assigning property values dynamically, 160–161
creating, 138–140
for 64-bit data provider, 427
FTP, 134
how to use, 702
HTTP, 134
manually handling transaction, 332
MSMQ, 135
Multiple Files, 134
Multiple Flat Files, 134
names, 136
ODBC, 135
OLE DB, 135
creating, 140–142
Lookup transformation, 218
package-scoped, 136–137
package templates, 407
parameterization, 137
case scenario, 125
parameterization of, 254
parameters and, 354
project-level, 354–355
project-scoped, 136–137
SMO, 135
SMTP, 135
SQL Server Compact Edition, 135
WMI, 135
Connection Managers pane, 120, 142, 143
Connection Managers property, 374
Connection Managers window, 355
Connection Manager tab, 186, 471, 473
connection properties
in package configurations, 368
updating, 363
connections
defining, 355
SSIS (SQL Server Integration Services) deployment, 111
Connections collection class, 711
connection setting, table location, 373
connection string paramaterization, 363–364
ConnectionString property, 702
connection strings
Flat File, 142
OLE DB, 142
sharing, 372
storing in a single location, 367
consistency
measurement of, 532
solutions for, 537
consolidated members, 591
constraints
foreign keys, 47
NOT NULL, 186
precedence, 164–169
completion, 165
creating, 118
failure, 165, 167–169
success, 165
container editors, property expression with, 362
Container property (Foreach Loop Editor), 159
containers, 145, 155–163
Foreach Loop, 156
For Loop, 156
logging configuration, 388–391
Sequence, 156
container-scoped variables, 248, 249, 251
Continuous Values, 688
control flow. *See also* data flow
debugging, 498, 500–503, 505
determining, 156–163
external processes, 171
master package concept, 265–266
parallel execution management, 517
tasks, 145, 147–155
administration, 151
analysis services, 153
Analysis Services Execute DDL, 153
Analysis Services Processing, 153
Back Up Database, 152
Bulk Insert, 150
CDC Control, 149, 305
Check Database Integrity, 152
custom, 154–155
data flow, 177, 178–197, 190–192
Data Flow, 150
Data Mining Query, 153
data movement, 150
data preparation, 148
Data Profiling, 148
Execute Package, 149
Execute Process, 149
Execute SQL, 150, 226, 292, 292–293
Execute SQL Server Agent Job, 152
Execute T-SQL Statement, 152
Expression, 149

File System, 148, 161, 167, 168
FTP, 148
History Cleanup, 152
maintenance, 151–152
Maintenance Cleanup, 152
Message Queue, 149
Notify Operator, 152
precedence constraints, 164–169
Rebuild Index, 152
Reorganize Index, 152
Script, 154
Send Mail, 149
Shrink Database, 152
Transfer Database, 151
Transfer Error Messages, 151
Transfer Jobs, 151
Transfer Logins, 151
Transfer Master Stored Procedures, 151
Update Statistics, 152
Web Service, 148
WMI Data Reader, 149
WMI Event Watcher, 149
workflow, 148–149
XML, 148
Control Flow Designer, 105–107
Control Flow tab, 336, 340, 502
Convert To Package Deployment Model, 368
Copy Column transformation, 199
copying
data
compared to specifying queries, 94
production data to development, 125
Corrected status, in cleansing, 648
Count aggregate function, 399, 656
COUNT_BIG aggregate function, 59
Count distinct aggregate function, 399
Count distinct operation, 400
Country/Region domains, in default KB, 639
covered queries, 56
CPU utilization condition, job scheduling with, 463
Create A Domain button, 642
Create A New Data Source To SQL Server dialog box, 192
Create Catalog dialog box, 445
Created operation status, 466
Create Entities button, 627
Create Environment editor, 485
Create New Data Source dialog box, 192
Create New Parameter option, 358
CREATE_OBJECTS permission, 482
Create Views tab, 609
Create Website window, 584
credit risk management, defining cases for, 688
CRM (Customer Relationship Management) applications
identity mapping, 735
integrating master data, 569
master data, 567
cross-domain rule, 640
cross joins, problems with, 737
CRUD cycle (create, read, update, and delete)
master data, 568
MDM (master data management), 569
custom code, 699–730
custom components, 716–730
script component, 707–715
script task, 700–707
custom components
developing, 718
interaction with SSIS developer, 719
planning, 717–718
vs. script components, 716
custom data flow transformation component
configuring, 728–730
deploying, 727–728
developing, 725–727
custom data flow transformations, 226
custom data source, script component as, 708, 709
custom destinations, in custom components, 717
customer complaints, 531
Customer dimension table
columns, 49–50, 65
creating, 49
loading data to DW, 209–212
loading data with SCD tranformation, 293–296
loading into DW, 212–214
set-based update logic, 292–293
updating, 229–231
Customer Relationship Management (CRM) applications. *See* CRM (Customer Relationship Management) applications
Customers entity, schema correctness, 534
custom logging events, 469
custom reports, for SSIS monitoring, 470
custom source, in custom components, 717
custom SSIS components, 291
custom tasks, 154–155
custom transformation, in custom components, 717

D

daily frequency, job schedudle, 463
data. *See also* master data
  auditing, 395
  automatic removal, 402
  batch editing, 624–626, 633
  batch import, 612–614
  cleansing, 646–648
  combining, 626
  copying
    compared to specifying queries, 94
  copying production data to development, 125
  correcting with DQS, 746
  correlating audit with logs, 401
  currency of, 533
  de-duplicating, 626
  determining correct automatically, 738
  editing and publishing, 627
  extracting from views, 94
  extracting from views and loading into tables, 91–98
  filtering, 629–630
  finding inaccurate, 654
  hierarchical, 566
  importing/exporting master data, 606–616
  importing from flat files, 194–197
  improving quality, 660
  improving quality of, 765
  isolating inaccurate, 532
  loading, 64–68
    auditing with data lineage, 73
    changing from T-SQL procedures to SSIS, 227
    creating data warehouses, 42–44
    data lineage, 73
    foreign key constraints, 47
    incremental loads, 299–316
    into Customer dimension table with SCD tranformation, 293–296
    preparing data transformations, 209–212
    to InternetSales table, 75
  loading with Excel, 629–630
  management in buffers, 512–513
  master, 566
    AdventureWorksDW2012 sample database, 573
    auditing, 568, 569
    complexity, 568
    CRM (Customer Relationship Management) applications, 567
    CRUD cycle (create, read, update, and delete), 568
    defining, 573–574
    definition, 567–568
    dimensions, 567
    ERP (enterprise resource planning) applications, 567
    hierarchical data, 569
    HRM (Human Resources Management) applications, 567
    MDM (master data management), 569–572
    nouns, 567
    reusage, 568
    versioning, 568, 569
    volatility, 568
  merging from multiple sources, 736
  metadata, 566
  partition switching, 72
  preparation of, 667, 687–690
  profiling, 654–660
  reading
    dynamic SQL, 299–304
  retaining audit, 401–402
  reusing with master data management, 609
  semi-structured, 566
  source
    incremental loads, 299
  sources for DQS projects, 647
  splitting for test set, 690
  status modification during cleansing, 648
  transactional, 566
    verbs, 567
  unstructured, 566
  validating, 646
  validating in advance, 626
  viewing in pipeline, 503–505
  XML, 566–567
data aggregation, 146
Data Analysis Expression (DAX), 30
Database Engine
  SQL Server feature choice, 424
  SSIS solutions storage, 423, 429
  upgrading, 430
Database Engine Services, limited SSIS functionality of, 425
database roles, administrative tasks, 481

databases
  AdventureWorks2012 sample
    extracting data from views and loading into tables, 91–98
  AdventureWorksDW2012 sample
    aggregating columns in tables, 59–62
    conformed dimensions, 8
    database schema, 14–15
    DimCustomer dimension, 19–20, 49
    DimDate dimension, 9
    dimensions, 24–26
    fact tables, 32–33
    lineage, 36–37
    many-to-many relationships, 31–33
    master data, 573
    normalized dimensions, 9–10
    reporting problems with normalized relational schemas, 5–7
    SCD, 36–37
    Star schema, 7–10
  creating, 42–44
  enabling CDC on, 304–305
  in Data Quality Server, 541
  log providers, 385
  of Data Quality Services, 547
  relational
    master data, 567
  shrinking, 153
  space
    calculating requirements, 43
  SQL Server
    creating, 48–49
  TK463DW
    enabling CDC on, 304–305
database schemas
  AdventureWorksDW2012 sample database, 14–15
  unintended usage of, 533
data buffers
  allocation with execution trees, 513–514
  architecture of, 512–513
  changing settings, 516
  counters for monitoring, 520
  fixed limit of, 514
  optimization of, 515–516
  synchronous vs. asynchronous components, 513
DataBytes event property, 387
data calculation, 146
data cleansing, 145–147, 537, 541, 551–552, 570, 571
DataCode event property, 387
data compatibility, 247
data compression, 61–62, 64–68
  dictionary compression, 62
  InternetSales fact table, 68–69
  page compression, 62
  prefix compression, 62
  row compression, 61
  Unicode compression, 62
data conflicts, in MDM (master data management), 572
Data Conversion transformation, 199, 247
data destination adapters, 177
  configuring, 185–186
  defining, 184–185
data encryption, in deployed SSIS solutions, 480
data error correction vs. prevention, 536
data extraction, 265, 268, 731
data files, 43, 44
data flow. *See also* control flow
  adding system variables to, 382
  Aggregate, 398
  BIDS (SQL Server Business Intelligence Development Studio), 179–180
  Cache Transform transformation, 223
  controlling with expressions, 262
  data path arrows, 206
  debugging, 498, 505
  defining data sources, 178–197
  defining destinations, 178–197
  design tips, 516–517
  error flows, 317–321
  error output, 382
  error outputs, 317
  Lookup transformations, 218–224
  Row Count, 398
  SCD (Slowly Changing Dimension), 286–291
  set-based updates, 225–227
  Sort transformation, 224–225
  tuning options, 514–516
  visualizing with data viewers, 503–505
data flow buffers
  accessing, 707
  data types, 246
  in scripting, 708
data flow destination adapters, 178, 184–187
Data Flow Path Editor, 503–505, 507
data flow rows, processing with profiling, 711–716
data flows, combining vs. isolating, 120

data flow source adapters, 178
  adding with Source Assistant, 181–182
  defining, 180–182
data flow task, 150, 177
  buffer optimization, 515
  capturing data from, 507
  checkpoints, 339
  components SSIS (SQL Server Integration Services) Toolbox, 178–179
  creating, 178–180
    for Person.Person table (TK463DW), 190–192
  defining data destination adapters, 184–187
  defining data source adapters, 180–183
  design tips, 516–517
  event handlers, 344
  package templates, 407
  parameterization of, 255
  rows, 177, 178
  transactions, 330, 332, 333
data flow transformation components
  Aggregate, 399
  Row Count, 399
data flow transformations, 178
data governance, 537, 570
data history, SCD problem, 21–23
data integration, data types and, 247
data integrity
  accuracy, 570
  defined, 529
  RDBMS (relational database management system), 569–570
data latency, 3
data life cycle, 535
data lineage, 73
data loading, 268
data loads, 265–266
data management, SSIS vs. LOB, 87
DataMarket account ID, 550
data merges, business keys, 23
data mining, 668–687
  algorithms for, 670–671
  defined, 668
  predictions, 671–679
  techniques of, 668
  uses for, 667, 669
data mining (DM) models
  additional uses for, 669
  storage of, 668
Data Mining Extensions (DMX) query, 672
Data Mining Model Training destination, 184
data mining projects
  SSIS sampling, 689
  steps of, 669
data mining queries, 132
Data Mining Query task, 153, 667
Data Mining Query transformation, 204
data modification language (DML) statements. *See* DML (data modification language) statements
data movements
  complex, 145–147
  complex data movements, 88
  DW, 88
  modifying movements, 112–123
    adding existing SSIS packages to projects, 112–114
    configuring connections, 120–123
    editing SSIS packages creating by SQL Server Import and Export Wizard, 114–116
    running SSIS packages in Debug mode, 120–123
  planning, 89–99
  simple data movements, 88
    creating, 91–98
  SQL Server Import and Export Wizard, 88, 89–99
    creating simple data movements, 91–98
    transformations, 89
  SSDT (SQL Server Data Tools), 88
  SSIS (SQL Server Integration Services), 87
  tasks, 150
data normalization, 145–147
data overview, 688
data path arrows, 206
data pivoting, 146
data platforms
  SSIS (SQL Server Integration Services) deployment, 111
data preparation tasks, 148
data profiling, 132, 530, 535, 552
  processing data flow rows with, 711–716
  with SSAS cubes, 656
Data Profiling task, 148
  incorporating results, 701
  using, 703
data providers
  32-bit, 137
  64-bit, 137
  data types, 246
  third-party, 660

data quality
  assessment of, 535–536
  checking for issues, 538–539
  cleansing methods, 537, 551–552
  defined, 529
  dimensions of, 531–532
  goals of master data managment, 530
  improvement plan for, 536–537
  MDM (master data management), 572
  measuring solutions, 537–538
  planning data quality projects, 535
  root causes of bad, 536
  schema dimensions, 534–535
  soft dimensions, 533
Data Quality Client
  administration with, 549–552
  cleansing and mathching, 551–552
  functions of, 541
  installation requirements, 542–543
  location of log file, 552
  log settings, 552–553
  main screen, 547
  monitoring DQS activity, 555–558
data quality data warehouse, schema for, 537. *See also* data quality warehouse
data quality dimensions
  accuracy, 532
  completeness, 531–532
  consistency, 532, 537
  defined, 530
  information, 532
  inspection of, 531
data quality projects
  creating, 646–653
  opening existing, 646
data quality schema dimensions, 534–535
Data Quality Services (DQS)
  architecture of, 540–541
  databases of, 547
  installation of, 542–547
  knowledge bases in, 540
  maintaining with Data Quality Client, 549–552, 555–558
  maintaining with other tools, 553–555
  security administration, 553–554
data quality soft dimensions, measurement of, 533
data quality warehouse, schema for, 538. *See also* data quality data warehouse
DataReader destination, 184
dataset size, estimating, 515
data source adapters, 177
  configuring, 182–183
  defining, 180–183
data sources
  connection managers
    creating, 138–143
    file formatting, 140
    Flat File, 138–140
    OLE DB, 140–142
  defining, 178–197
  selecting, 91
data staging area (DSA), 44
data stewards, 537, 570
data stores
  auditing, 396
  connection managers, 133–144
    64-bit data providers, 137
    ADO, 134
    ADO.NET, 134, 136
    Analysis Services, 134
    Excel, 134
    File, 134
    Flat File, 134
    FTP, 134
    HTTP, 134
    MSMQ, 135
    Multiple Files, 134
    Multiple Flat Files, 134
    names, 136
    ODBC, 135
    OLE DB, 135
    package-scoped, 136–137
    parameterization, 137
    project-scoped, 136–137
    SMO, 135
    SMTP, 135
    SQL Server Compact Edition, 135
    WMI, 135
data taps
  adding, 509–510
  using, 507–508
data tap stored procedure, 507
data transformations, 177, 198–214, 265, 268
  advanced data preparation, 204
  blocking, 199, 203
  Cache Transform
    Lookup transformation, 223–224

Conditional Split, 214, 291
custom, 226
Derived Column, 206, 212
loading data, 209–212
logical-row, 199–200
Lookup, 200–201
cache modes, 218–219
Cache Transform transformation, 223–224
case sensitivity, 220
composite joins, 220
connection managers, 218
ETL (extract-transform-load), 218–224
merging outputs with Union All component, 220–221
replacing Union All transformation, 228–229
rows with no matching entities, 220
Merge Join, 211
multi-input, 200–201
multi-output, 200–201
multi-row, 202–203
non-blocking, 199
partial-blocking, 199
resolving column references, 207–208
SCD (Slowly Changing Dimension), 285–290
Script Component, 291
selecting, 198–204
Sort, 224–225
Union All
replacing multiple Lookup transformation outputs, 228–229
Data type property, 244, 357, 639
data types
Date, 284
DateTime, 284
of variables, 245–248
purpose of, 247
SSIS (SQL Server Integration Services), 187–189
data unpivoting, 146
data validation, 146
data values, variables measurement, 688
data viewers
removing, 505
visualizing data flow with, 503–505
Data Viewer tab, 503–504
Data Viewer window
Detach, 505
Play button, 505
data warehouses (DW), 3
auditing, 396
creating databases, 42–44
data history
SCD (Slowly Changing Dimension) problem, 21–23
data movements, 88
design
queries, 4
Dimensional Models, 6
loading data to, 209–212
maintenance of, 265, 462
metadata storage w/ continuously merged data, 571
naming conventions, 17–18, 21
performance
batch processing, 62–64
columnstore indexes, 62–64
data compression, 61–62
indexed views, 58–61
indexing dimensions, 56–59
indexing fact tables, 56–59
data warehousing
automation of, 239–240
individual elements of, 267
operations included, 265–266
Date data type, 284
date functions, 258
Dates dimension
columns, 52, 67
creating, 52–53
DateTime data type, 246, 284
DAX (Data Analysis Expression), 30
DBNull data type, 246
db_owner database role, 481
debugging
control flow, 498, 500–503
data flow, 498
icons, 498
Debug logging level, 552
Debug menu, 505, 702
Debug mode, for SSIS packages, 120–123
Debug toolbar, 503
Decimal data type, 246
Decision Trees mining model, 671–673
de-duplicating
and data cleansing, 646
and identity mapping, 738
in domain management, 640
problems with, 738
with fuzzy grouping, 758
default knowledge, in DQS, 639

DefaultMaxBufferRows parameter, 515
DefaultMaxBufferSize parameter, 515
default permissions, 483
default value parameter, 356
defining
  attribute types, 284
  data flow destination adapters, 184–187
  data sources, 178–197
  destinations, 178–197
  ETL (extract-transform-load) strategy, 217
delayed validation, 259
DelayValidation property, 259
Delete All Breakpoints, 505
Delete Parameter toolbar button, 357
deleting fact tables, 71–73
delimited files, Flat File source, 196
denormalized dimensions
  DimDate table, 9–10
  partially denormalized dimensions, 11
Deny permission, 618
deployment environment
  SSIS (SQL Server Integration Services)
    development environments, 111
    production environments, 111
  SSIS (SQL Server Integration Services) projects, 110–124
deployment environments, switching between, 360
deployment files, 443
Deployment Wizard, 442–443, 446
Derived Column transformation, 200, 206, 212, 401
Derived Column Transformation Editor, 206
derived hierarchies, 590
derived variables, in data mining, 688
Description property, 244, 357
design
  data flow, 516–517
  dimensions, 17–27
    column types, 17–18
    hierarchies, 19–22
    SCD (Slowly Changing Dimension) problem, 21–24
    Type 1 SCD, 22
    Type 2 SCD, 22
    Type 3 SCD, 23
  fact tables, 27–33
    additivity of measures, 29–30
    columns, 28–29
    many-to-many relationships, 30–32
  logical, 3
    Snowflake schema, 3, 4, 9–11
    Star schema, 3, 4, 7–10, 11
  queries, 4
design environment, data providers for, 137
design flow, parallel execution and, 517
design-time methods, customizatin of, 719–721
design-time troubleshooting, 498–505
design time validation, delaying, 259
destination adapters
  debugging, 505
  execution trees, 513–514
  optimization of, 517
Destination Assistant component, 184
destination environment
  determining, 421, 424
  preparing, 422
destination, script component as, 708, 709
destinations, defining, 178–197
destination tables, truncating, 117
Detach, Data Viewer window, 505
determinism, 239, 240
development environment
  copying production data to, 125
  SSIS installation choices, 424
  SSIS (SQL Server Integration Services), 110
  troubleshooting, 498
development execution, speeding, 505
development vs. production environments
  SSIS package deployment, 421, 437
Diagnostic event, 386
diagrams, dimensions, 24–26
dialog boxes
  Add Copy Of Existing Package, 113
  Add New Source, 181
  Add SSIS Connection Manager, 140, 141
  Add Variable, 158, 159
  Advanced Editor, 189
  CDC Source Editor, 306
  Configure Error Output, 318
  Connection Manager, 141
  Connection Manager Conversion Confirmation, 143
  Create A New Data Source To SQL Server, 192
  Create New Data Source, 192
  Execute SQL Task Editor, 191
  Input Output Selection, 214
  Load Package, 113
  OLE DB Source Editor, 182, 191

Set Breakpoints, 500–501
Set Query Parameters, 300–301
dictionary compression, 62
DimCustomer dimension
AdventureWorksDW2012 sample database, 19–20, 49
attributes, 19–20
DimCustomer.dtsx SSIS package, log confirmation template, 392
DimDate denormalized dimension, 9–10, 20
Dimensional Models, 6
dimensional storage, in SSAS, 30
dimension primary key, joins, 57
Dimension Processing destination, 184
dimensions, 3, 8–9, 567
additivity of measures, 29–30
columns, 17–18
conformed, 8
creating, 49–52
Customers
columns, 49–50, 65
Dates
columns, 52, 67
creating, 52–53
denormalized
DimDate, 9–10
partially denormalized dimensions, 11
design, 17–27
column types, 17–18
hierarchies, 19–22
SCD (Slowly Changing Dimension) problem, 21–24
Type 1 SCD, 22
Type 2 SCD, 22
Type 3 SCD, 23
diagrams, 24–26
DimCustomer (AdventureWorksDW2012 sample database), 19–20
DimDate (AdventureWorksDW2012 sample database)
AdventureWorksDW2012 sample, 9
attributes, 20
DimProduct (AdventureWorksDW2012 sample database), 10
granularity, 12
implementing, 45–47
indexing, 56–59
intermediate, 32
late-arriving, 48, 285
multiple fact tables, 8
naming conventions, 17, 21
private, 9–10
Products
columns, 66
creating, 51
shared, 8–9
DimProduct, 8
POC projects, 9–10
SSAS (SQL Server Analysis Services), 9
Type 1 SCD, 22
Type 2 SCD, 22
surrogate key, 284
Type 3 SCD, 23
updating, 290–293
checking attribute changes, 291–292
set-based updates, 292–293
dimension tables, 396
history, 287
selecting, 287
DimProduct dimension (AdventureWorksDW2012 sample database), 8, 10
direct assignments, 467
directed algorithms, in dining mining, 668
Disable Data Viewer, 505
DisableEventHandlers property, 346
Discrete Values, 688
discretizing
attributes, 17
automatic, 18
groups, 17
disk storage, filegroups, 44
disk-swapping, 516
distributed transactions, 328
distribution of values, determining, 655
DML (data modification language) statements
testing execution of, 43
DMX query builder, 677
DMX Query Editor, 672
documentation
MDM (master data management), 573
measurement of, 534
domain accounts, 426
domain-based attributes, 590
Domain management, 638
Domain Management setting, 552
domain rules, 640
Domain Rules tab, 644

domains
  changing values of, 640
  copying, 640
  creating linked, 640
  default KB, 639
  importing/exporting, 639
  in knowledge bases, 638
  management, 639–641
  manual editing of, 643–645
  MDM (master data management), 573
domain user accounts, insatlling MDS, 577–578
domain values, weighting in DQS matching, 747
Double data type, 246
DQLog.Client.xml, 552
DQS activities, life cycle of, 647
DQS Administrator role, 553
DQS cleansing project, creation of, 649–652
DQS Cleansing transformation, 204
DQS Cleansing transformations
  advanced configuration for, 746
  using, 745–746, 748–751
DQS databases, backup and restore, 554–555
DQS Data knowledge base, default storage, 639
DQS (Data Quality Services), de-duplication with, 626
DQServerLog.DQS_MAIN.log, 552
DQSInstaller application, actions of, 543
DQSInstaller.exe, 543
DQS KB Editor role, 554
DQS KB Operator role, 554
DQS log files
  deleting, 555
  location of, 552
DQS_MAIN, 541
DQS matching, 746–747
  algorithms in, 738
  using, 752–755
  vs. fuzzy transformations, 740, 756
DQS notification, 552
DQS projects
  creation of, 646
  knowledge bases requirements, 647
  management activities, 646–647
DQS_PROJECTS, 541
DQS_STAGING_DATA, 541
drillthrough, enabling in data mining, 668
DSA (data staging area), 44
DTExec command-line utility, 461–462, 518
DTExecUI utility, 458, 474
DTSExecResult enumeration, event properties, 387
Dts object, 700, 701, 702
Dump On Errors check box, 473
duplicate values, as inaccurate data, 532
dynamic packages, uses of, 353
dynamic pivoting, 203
dynamic SQL
  reading data, 299–304
    ADO Net source adapter, 302–304
    OLE DB Source adapter, 300–302

E

early-arriving facts, 285, 288
ease of use, measurement of, 533
Edit Breakpoints, 500
editing, SSIS (SQL Server Integration Services) packages, 99, 114–120
Edit Value, 502
effective permissions, determining, 619
elementary auditing, 395, 396
email, 132
Empty data type, 246
Enable Data Viewer check box, 503
Enable Data Viewers, 503
Enable Inferred Member Support check box, 288
Enable Notifications value, 552
Enable Package Configurations check box, 368
Encryption Algorithm Name property, 439
encryption, in SSISDB catalog data, 438, 439
Ended unexpectedly operation status, 466
EndTime event property, 387
EngineThreads property, 517
English text, in term extraction, 680
enterprise resource planning (ERP) applications. *See* ERP (enterprise resource planning) applications
entities, 589
  attributes, 590, 591
  base, 590
  creating with Excel, 627–628
  de-duplicating, 735
  keys, 18
  MDS (Master Data Services) models, 589–590
  populating, 596–599
  StateProvince
    members, 597
  tables, 589

Entities metadata, 606
entity-relationship diagrams, 534
entity sets, 567
entropy, measurement of, 532
EntryPoint property, 701
Enumerator property (Foreach Loop Editor), 157
Environment list box, 471
environment, overriding values, 467
Environment Properties dialog box, 486
Environment references, 441
environment reference validation, 466–467
Environments node, 485
Environments object, 482
environment variables, 370, 371, 373, 441
ERP (enterprise resource planning) applications
  master data, 567
erroneous values vs. outliers, 688
Error Code column, 505, 608
error codes. *See* HTTP Status Codes
error detection, automatic execution of, 239
error flows, ETL, 317–321
Error logging level, 552
error outputs, 317
  debugging with, 505
  in data flow components, 382
error paths, 317–318, 319
errors
  allowing in package execution, 499
  capturing with continuous monitoring, 506
  column references, 208
  correction vs. prevention, 536
  described in debuging, 499
  identifying in Data Viewer window, 505
  truncation, 508–509
Estimated Row Size parameter, 515
ETL (extract-transform-load), 88, 177
  architecture, 217–218
  defining strategy, 217
  determining strategies/tools, 216–231
    landing zones, 217
    Lookup transformations, 218–224
    set-based updates, 225–227
    Sort transformation, 224–225
  for data preparation, 667
  Lookup transformations, 218–224
ETL (extract-transform-load) process, 13
  batch cleansing during, 646
  custom solutions for, 342–344
  error flows, 317–321
  hash functions, 291–292
  incremental loads, 307–308
EvaluateAsExpression property, 244
event handlers
  defined, 342
  implementing, 344–346
  in debugging, 505
  package templates, 407
  turning off, 346
  types of, 343
  using, 342–344
Event Handlers tab, 342, 345
events
  extending data, 383
  identifying, 382
  logging groups, 386
  possible properties, 386–387
  raising in script component, 711
Events property, exposing, 702
Event Viewer, accessing, 392
Excel Add-in, for MDM, 624–631
Excel connection manager, 134
Excel destination, 184
Excel files, as source for DQS project, 180, 647
Excel Fuzzy Lookup Add-in, 758
Excel, modifying data in, 630
exceptions
  detecting, 382
  identifying, 382
exclamation point tooltip, 552
ExclusionGroup property, 709
exclusion terms, defining, 680
executable component, event handlers, 342
Executables property, 374
EXECUTE AS Transact-SQL statement, 489
EXECUTE_OBJECTS permission, 482
Execute Package dialog box, 458, 473
Execute Package task, 149, 265, 269
Execute Package tasks, 517
Execute Package Utility, default invocation, 427
EXECUTE permission, 482
Execute Process task, 149, 270, 699
Execute SQL Server Agent job task, 269–270
Execute SQL Server Agent Job task, 152

Execute SQL task, 133, 150, 226
  checkpoints, 338
  data types for, 247
  for T-SQL code, 699
  implementing event handlers, 344, 345
  manually handling transactions, 332
  package templates, 407
  property parameterization, 251
  set-based update logic, 292, 292–293
Execute SQL Task Editor, 115, 362
Execute SQL Task Editor dialog box, 191
Execute SQL Task properties, 116
Execute T-SQL Statement task, 152
execution
  automated, 462–463
  monitoring, 465–470
  new execution instance, 467
  observing for troubleshooting, 498–499, 500
  of SSIS solutions, 456
  on-demand, 457–462
execution boundaries, 385, 398
execution boundary events, 386, 387
execution data, storing, 396
execution engine, Integration Services, 421
execution environment, data providers, 137
execution errors
  displaying connection context, 506
  identifying, 382, 384, 386
  jobs vs. master packages, 464
execution events, 342
execution exception events, 386, 398
ExecutionID, event property, 387
execution instance, 467–468, 507
Execution instance GUID variable, 400
Execution Performance report, 519
execution progress events, 386
Execution Results tab, 499
execution speed, establishing, 518
Execution start time variable, 400
execution times, described in debugging, 499
execution trees, 513–514, 520–521
Execution value, parameter value, 356
EXISTS operator, 739
explicit assignment, property parameterization, 251
explicit hierarchies, 591
Explorer area, access to, 618
Export Column transformation, 200
Export stage, of cleansing projects, 648
Expression Builder, 262
expressions
  date and time functions, 258
  mathematical functions, 257
  null functions, 258–259
  operators, 255–256
  property expressions, 259
  setting properties, 303
  string functions, 257–258
Expressions property, 244, 302, 361
Expressions tab, property expressions, 362
Expression task, 149, 259
  parameterization with, 254
  setting variables with, 356
extension points, 699
external audit table schema, 397
external processes, 131, 171, 270
extracting
  data from views, 94
  data from views and loading into tables, 91–98
extraction processes, 267
extract-transform-load. *See* ETL (extract-transform-load)

F

FactInternetSales table, adding partitioning column to, 74
fact tables, 3, 396, 567
  additivity of measures, 29–30
  AdventureWorksDW2012 sample database, 32–33
  business key, 28–29
  columns, 28–29, 47–48
    analyzing, 33–34
    lineage type, 29
  columnstore indexes, 64
  creating, 52–54
  deleting large parts of, 71–73
  design, 27–33
    activity of measures, 29–30
    columns, 28–29
    many-to-many relationships, 30–32
  early-arriving facts, 285, 288
  foreign keys, 28
  granularity, 12
  implementing, 47–48
  indexing, 56–59

inferred members, 47
inserting unresolved records, 285
InternetSales table
creating, 52–53
loading
partitions, 71–73
loading data
guidelines, 308
incremental loads, 299–316
loading incrementally, 378
many-to-many relationships, 30–32
partitioning, 44–45, 74–76
partition switching, 76–78
Star schema, 7–8
surrogate keys, 28
Failed operation status, 466
failover cluster, 426
FailPackageOnFailure property, 337, 338, 340
Fail Transformation option, 317
failure precedence constraints, 165, 167–169
failures
avioding, 381
identifying, 382, 384
jobs vs. master packages, 464
logging, 385
Fast Load option, 517
Fast Parse property (Advanced Editor), 189–190
Fatal logging level, 552
Feature Selection screen, 433
file attributes, 590
File connection manager, 134, 384
File Connection Manager Editor, 162
file formatting, connection managers, 140
filegroups, 44
partition schemes, 72
Primary
creating partition schemes, 74
SQL Server Customer Advisory Team (SQLCAT) white papers, 44
files
delimited
Flat File source, 196
flat
importing data from, 194–197
processing with SSIS (SQL Server Integration Services) packages, 157–159
Files property (Foreach Loop Editor), 157
file systems, 131
File System task, 148, 161
general settings, 161, 167, 168
transaction failure, 330
File Transfer Protocol. *See* FTP
Fill Stage Tables container, 390
FillStageTables.dtsx SSIS package, 389, 392
filtered indexes, 57
filtered nonclustered indexes, 57
filtering method, 739
final destination, described in debugging, 499
FireError method, 721
FireInformation method, 721
FireWarning method, 721
“five whys” method, 536
Fixed attribute, 284
Fixed Attribute output, 290
flags, Type 2 SCD, 22
Flat File connection manager, 134
assigning property values dynamically, 160–161
creating, 138–140
Flat File Connection Manager Editor, 139, 196–199, 346
flat file data sources, column configuration, 515
Flat File destination, 184
Flat File destination adapter
in debugging, 505
in event handling, 345
flat files
importing data from, 194–197
multiple
merging, 225
Flat File source, 180
floating-point numbers, 590
Folder property (File System task), 161, 168
Folder property (Foreach Loop Editor), 157
folders, in SSISDB catalog, 440
Folders object, 482
Foreach File Enumerators, logging, 411
ForEach Loop container, 156
debugging, 501
in debug environment, 499
logging variables, 411
package templates, 407
properties update, 363
TransactionOption, 330
unique identifier, 386
ForEach Loop Editor, 157–159
ForEach loop, generating object lists, 701

foreign keys
  adding to InternetSales tables, 75
  constraints, removing, 47
  fact tables, 28–29
  implementing fact tables, 47–48
foreign languages, term extraction for, 680
For Loop containers, 156
  debugging, 499, 501
  TransactionOption, 330
  unique identifier, 386
Format output property, in domains, 639
FOR XML directive, 248
free-form attributes, 590
Frequency column, in term lookup, 682
frequency distribution of values, 532
Frequency threshold, in term extraction, 681
fresh installation, 428
FTP connection manager, 134
FTP (File Transfer Protocol), 131
FTP task, 148
full-cache lookups, 517
Full Cache mode (Lookup transformations), 218
Full recovery model, 43
functional area access, 618–619
Functional area access permission, 617
Functional Dependency, profiling of, 657
functions
  aggregate, 29–30
  AVERAGE aggregate, 29
  COUNT_BIG aggregate, 59
  hash
    implementing in SSIS, 291–292
  LastNonEmpty aggregate, 30
  NEXT VALUE FOR, 45
  Partition, 72
  SUM aggregate, 29
  T-SQL HASHBYTES, 291
Fuzzy Grouping, 756
Fuzzy Grouping transformation, 204, 740
Fuzzy Lookup, 756
Fuzzy Lookup Add-in for Excel, obtaining, 759
Fuzzy Lookup transformation, 205, 740

G

General tab (Lookup Transformation Editor), 219
governance (data), 570
granularity dimensions, 12
graphs, pivot, 17
Group by aggregate functions, 399, 400
grouping, tasks, 155
groups, discretizing, 17
GUID, audit transformation, 400

H

hard dimensions
  defined, 531
  measurement of, 535, 537
hard disk space requirements, 425
hardware, monitoring consumption of, 519–520
hash functions, implementing in SSIS, 291–292
hash joins
  bitmap filtered, 57
  Query Optimizer, 57
help, in script task, 702
heterogeneous data, integration of, 247
hierarchical data, 566, 569
hierarchies
  derived, 590
  dimensions, 19–22
  explicit, 591
  levels, 20
    members, 20
  recursive, 590–591
  Snowflake schema, 21
  Star schema, 21
  Type 1 SCD, 22
Hierarchies metadata, 606
hierarchy member permissions, 617, 619
high determinism, 240
Historical Attributes Inserts output, 290
history
  defining attribute types, 284
  dimension tables, 287
History Cleanup task, 152
history (data), SCD, 21–23
Hit Count breakpoint, 501
Hit Count Type breakpoint, 501
home pages, Master Data Manager, 585
HRM (Human Resources Management) applications, master data, 567
HTTP connection manager, 134
hypercubes. *See* Star schema

I

icons
  in debug environment, 498–499
  for breakpoints, 501
ID column, in stg.entityname_Leaf table, 608
IdentificationString property, 507
identity columns, compared to sequences, 45–46
identity mapping
  and data merging, 736
  and de-duplicating, 738
  central metadata storage, 571
  preparing data for, 740–744
  problems with, 736–738
  with fuzzy transformations, 758
IDEs (integrated development environment), for SSDT, 110
Ignore Failure option, 222, 317
IIS (Internet Information Services), Master Data Manager, 577
implicit permissions, 483
Import Column transformation, 200
importing, data from flat files, 194–197
ImportStatus_ID, in stg.entityname_Leaf, 608
ImportType, stg.entityname_Leaf, 608
inaccurate data, isolated, 532
IncludeInDebugDump property, 244
incremental loads, 299–316
  creating incremental load packages, 312–314
  ELT (extract-transform-load), 307–308
  implementing CDC with SSIS, 304–307
  reading data with dynamic SQL, 299–304
indexed views, 58–61
indexes
  aligned, 72
  clustered, 56–58
  columnstore, 62–68
    catalog views, 63–64
    fact tables, 64
    InternetSales fact table, 68–69
    segments, 63
  filtered, 57
  nonclustered
    filtered, 57
  partition, 72
indexing
  dimensions, 56–59
  fact tables, 56–59
  testing methods, 79
indirect configurations, using, 372
indirect file location approach, 371
inferred dimension members, 285, 288
inferred members, 47
Inferred Member Updates output, 290
Info logging level, 552
information, measuring quality, 532
INFORMATION_SCHEMA ANSI standard view, 654
Information Theory, 532
inherited permissions, 483, 618
Initial Staging, ETL, 217
in-place upgrade, 428
Input and Output Properties tab (Advanced Editor), 188
inputFileName variable, 249
Input Output Selection dialog box, 214
inputs for script component, 708–709
INSERT statement, 73
installation footprint, minimizing, 424
Installation Type screen, 432
installing
  MDS (Master Data Services), 575–587
    domain user accounts, 577–578
    installation operations, 581
    post-installation tasks, 582–586
    pre-installation tasks, 579–581
    SQL Server Setup, 578
    user accounts, 577
    Windows Web Server roles, 577
  Silverlight 5, 586
Int16 data type, 246
Int32 data type, 246
Int64 data type, 246
integrated development environment. *See* IDEs (integrated development environment)
Integration Management
  access to, 618
  creating subscription views in, 609
Integration Services
  dashboard, 469, 506
  default database choice, 429
  deployment, 371
  Deployment Wizard, 442
  development vs. production environment, 425
  initialization operation, 442
  operations report, 506
  performance tuning, 422
  row level security, 483
  security model, 422
  SQL Server feature choice, 424
  SSIS process hosting, 421, 422

Integration Services Auditing, 382
integration services catalog, 461
  accessing SSISDB catalog, 506
  project deployment, 356
Integration Services Logging, 382
integrity (data)
  accuracy, 570
  RDBMS (relational database management system), 569–570
intention, measurement of, 533
Interactive cleansing stage, 648
interactive survivorship process, 756
intermediary-level audits, 396
intermediate dimensions, 32
Internet Information Services. *See* IIS (Internet Information Services)
InternetSales fact table
  adding foreign keys, 75
  columns, 67
  columnstore indexes, 68–69
  creating, 52–53
  creating columnstore index, 75
  data compression, 68–69
  loading data, 75
Invalid status, in cleansing projects, 648
IsDestinationPathVariable property (File System task), 161, 167
isolating vs. combining data flows, 120
IsolationLevel property, 331
isolation levels, for transactions, 331
IsSourcePathVariable property (File System task), 161, 167
Itanium-based operating systems, feature support on, 427

J

Jaccard similarity coefficient, 738, 756
Jaro-Winkler distance algorithm, 738
jobs, in SQL Server Agent, 463
joined tables
  indexed views, 58–61
  staging area, 227–228
joins
  composite
    Lookup transformations, 220
  dimension primary key, 57
  hash
    bitmap filtered, 57
    Query Optimizer, 57
  merge loops, 58
  nested loops, 58
  non-equi self joins, 60
  SQL Server, 58–59
  SQL Server Query Optimizer, 57

K

keys, 18
  business, 28–29
    merging data, 23
  clustering, 56
  foreign
    adding to InternetSales tables, 75
    fact tables, 28–29
  surrogate, 23–24
    fact tables, 28
    Type 2 SCD, 284
keys, primary and secondary. *See* primary and secondary keys
key terms, retrieving, 680
knowledge base (KB)
  creating additional, 661
  creating and maintaining, 638–645
  for DQS matching, 746–747
Knowledge Base Management screen, 642
knowledge bases, preparation of, 541
knowledge discovery
  initiating, 638
  using, 641–643
KnowledgeDiscovery setting, 552

L

landing zones, ETL, 217
language translation, 18
LastNonEmpty aggregate function, 30
late-arriving dimensions, 48, 285
latency, 3
leading value
  defined, 639
  uses of, 640
leaf-level members, 591

levels (hierarchies), 20
Levenshtein distance algorithm, 738
lineage information, columns, 18
lineage, tables, 13
lineage type, fact table columns, 29
line-of-business. *See* LOB
Line of Business (LOB). *See* LOB (Line of Business)
linked domains, creating, 640
listings. *See also* code
  Flat File connection string, 142
  OLE DB connection string, 142
loading
  data, 64–68
    auditing with data lineage, 73
    changing from T-SQL procedures to SSIS, 227
    creating data warehouses, 42–44
    data lineage, 73
    foreign key constraints, 47
    incremental loads, 299–316
    into Customer dimension table with SCD tranfor-
      mation, 293–296
    preparing data transformations, 209–212
    to InternetSales table, 75
  data into tables, 91–98
  dbo.Customers dimension table, 212–214
  fact tables
    partitions, 71–73
Load Package dialog box, 113
load processes, 267
LOB (Line of Business)
  data latency, 3
  data management operations compared to SSIS, 87
  naming conventions, 3
  OLTP applications, 566
  tables, 5
Locals windows, viewing variables, 502
locked projects, access to, 647
log configuration templates, 388, 392
Log Entry screen, 514
Log Events window, 514
logging
  benchmarking performance with, 518–519
  correlating data, 401, 402
  customized for ETL, 342, 344
  Integration Services Logging, 382
  multiple levels, 411
  packages, 383–388
  production-time troubleshooting, 506
  SSIS (SQL Server Integration Services), 13
  vs. auditing, 398
Logging Level list box, 473
logging levels, 468–469
LoggingMode property, 387–388
logical design, 3
  Snowflake schema, 3, 4, 9–11
    case scenarios, 34–38
    hierarchies, 21
  Star schema, 3, 4, 7–10
    case scenarios, 34–38
    hierarchies, 21
    queries, 11
logical-row transformations, 199–200
logic, controlling w/ variables, 502
log providers
  configuring, 386–388, 388–391
  package templates, 407
  selecting, 384–385
  types of, 383–384
Log Providers property, 374
log sequence number (LSN) ranges, CDC packages, 305
Lookup transformation, 200–201
  cache modes, 218–219
  Cache Transform transformation, 223–224
  case sensitivity, 220
  composite joins, 220
  connection managers, 218
  ETL (extract-transform-load), 218–224
  merging outputs with Union All component, 220–221
  replacing Union All transformation, 228–229
  rows with no matching entities, 220
Lookup Transformation Editor, 213, 218–219
loop containers, TransactionOption, 330
low-level audits, 396
LSN (log sequence number) ranges, CDC packages, 305

M

Machine name variable, 400
Maintenance Cleanup task, 152
maintenance tasks, 132, 151–152
managed service account (MSA), SSIS installation, 426
management activities, on existing projects, 646

Management Tools
  Complete installation option, 427
  development vs. production environment, 425
Manage My DataMarket RDS Providers, 550
MANAGE_OBJECT_PERMISSIONS permission, 482
Manage Parameter Values dialog box, 360
MANAGE_PERMISSIONS permission, 482
Manager attribute, 590
Manage Users page, 621
manual cleansing methods, 537
manual matching, 737
many-to-many relationships
  fact tables, 30–32
  schema correctness and, 534
mapping
  columns, 95
    OLE DB Destination adapter, 230
    OLE DB Destination Adapter, 214
    OLE DB Destination Editor, 187
  optimizing, 739
  parameters to SSIS variables, 301
Mappings tab (OLE DB Destination Editor), 186
Mapping stage, of cleansing projects, 648
Mapping tab, 346
master data, 566. *See also* MDM (master data management)
  AdventureWorksDW2012 sample database, 573
  auditing, 568, 569
  complexity, 568
  CRM (Customer Relationship Management) applications, 567
  CRUD cycle (create, read, update, and delete), 568
  defined, 529
  defining, 573–574
  definition, 567–568
  dimensions, 567
  ERP (enterprise resource planning) applications, 567
  hierarchical data, 569
  HRM (Human Resources Management) applications, 567
  identity mapping/de-duplicating, 735–766
  MDM (master data management), 569–572
    authority, 572
    central, 571
    central metadata storage, 571
    challenges, 572
    data conflicts, 572
    data quality, 572
    data stewards, 570
    documentation, 573
    domain knowledge, 573
    no central MDM, 570
  nouns, 567
  reusage, 568
  versioning, 568, 569
  volatility, 568
master data management. *See* MDM (master data management)
Master Data Manager, 576–577, 623
  home page, 585
  IIS (Internet Information Services), 577
Master Data Manager web application
  creating subscription views with, 609
  functional access for, 618
  initiating staging process, 608
Master Data ribbon tab, 627
Master Data Services. *See* MDS (Master Data Services)
Master Data Services (MDS). *See* MDS (Master Data Services)
Master Data tab, 625
master package concept
  configuring, 274–275
  creating, 270–274
  for advanced control flow, 265–266
  vs. SQL Server Agent jobs, 464
MasterPackageID, 271
match index, 760
matching
  attaining 100% accurate, 737
  attaining the best, 737
  controlling in fuzzy lookup, 758
Matching policy, 638–639
Matching Policy And Matching Project setting, 552
matching policy KB, 746
matching projects, 646, 747
matching rules, 639, 746
mathematical functions, 257
MaxBufferSize parameter, 515
MaxConcurrentExecutables property, 517
Maximum aggregate function, 399
MaximumErrorCount property, 499
Maximum length in term extraction, 681
Maximum Number of Versions per Project property, 439
MaxProjectVersion operation type, 442
MDM (master data management), 569–572

authority, 572
central, 571
central metadata storage, 571
continuously merged data, 571
identity mapping, 571
challenges, 572
data conflicts, 572
data quality, 530, 572
data stewards, 570
documentation, 573
domain knowledge, 573
no central MDM, 570
RDBMS (relational database management system), 569
MDM (master data management) solutions, 605–633
defining security, 616–624
Excel Add-in for, 624–632
importing and exporting data, 606–616
quality data, 529, 537
mdm.tblUser table, 617
mdm.udpSecurityMemberProcessRebuildModel stored procedure, 617, 619
mdq.NGrams function, 738
mdq.SimilarityDate function, 738
MDS Add-in for Excel, using, 763–764
MDS Configuration folder, 607
MDS Configuration Manager, 577–578, 608
MDS connections, in Excel, 625
MDS (Master Data Services), 541
architecture, 576–577
case scenarios, 600–601
creating MDS models, 588–599
installing, 575–587
domain user accounts, 577–578
installation operations, 581
post-installation tasks, 582–586
pre-installation tasks, 579–581
SQL Server Setup, 578
user accounts, 577
Windows Web Server roles, 577
models, 588–589
entities, 589–590
objects, 588–589, 589–592
permissions in, 617
security settings in, 616
MDS metadata, in model deployment, 606
MDS model deployment packages
creating and deploying, 606, 610–611
exporting data, 609
importing batches of data, 607–609
MDSModelDeploy utility, 606–607
MDS Objects, creating in Excel, 627–628
MDS security
defining, 620–622
users and permissions, 617
MDS (SQL Server 2012 Master Data Services), 572
MDS string similarity functions, 739
MDS System Administrator, changing, 617
MDS web service, 609
member properties, 18, 18–19
members
consolidated, 591
hierarchy levels, 20
inferred dimension, 285
setting, 288
StateProvince entity, 597
memory buffers. *See* data buffers
Merge Join transformation, 201, 211
Merge Join Transformation Editor, 211
merge loop joins, 58
MERGE statement, 293
Merge transformation, 201
MERGE T-SQL statement, Execute SQL task, 226
merging
data
business keys, 23
Lookup transformation outputs with Union All component, 220–221
multiple flat files, 225
MessageBox.Show method, in debugging, 505
Message Queue task, 149
MessageText event property, 387
metadata, 566
and master data, 606
configuring for script component, 708
folder, 440
parameter, 441
project, 440
Microsoft CodePlex, custom SSIS components, 291
Microsoft Distributed Transaction Coordinator (MSDTC)
cross-host communication, 330, 332
transaction definition, 328–329
Microsoft.NET Framework 4, 542
Microsoft.Net Framework System.TypeCode enumeration, 245–246

Microsoft Silverlight 5
  installing, 586
  Master Data Manager, 577
Microsoft.SQLServer.DTSPipelineWrap namespace, 718
Microsoft.SQLServer.DTSRuntimeWrap namespace, 718
Microsoft.SQLServer.PipelineHost namespace, 718
Microsoft Visual Studio, breakpoint functionality, 500
Microsoft Visual Studio Tools for Applications (VSTA), scripting with, 700
migration, 429, 451
MinBufferSize parameter, 515
minimalization, measurement of, 534
minimal rights principle, 426
Minimum aggregate function, 399
mining models
  creating, 672
  limiting data feed in, 689
Min Record Score value, 551
Min Score For Auto Corrections value, 551
Min Score For Suggestions value, 551
Model Administrators, 617–618
Model Data Explorer pane, 626
model deployment packages
  creation of, 606, 610–611
  deploying, 611
Model Deployment Wizard, 606–607
model object permissions, 617, 618
model permission, assigning, 622
models, MDS
  creating, 588–599
  entities, 589–590
modifiedRecordCount variable, 271
MODIFY_OBJECTS permission, 482
MODIFY permission, 482
modules, defined, 553
movements (data)
  complex, 145–147
  DW, 88
  modifying movements, 112–123
    adding existing SSIS packages to projects, 112–114
    configuring connections, 120–123
    editing SSIS packages created by SQL Server Import and Export Wizard, 114–116
    running SSIS packages in Debug mode, 120–123
  planning, 89–99
  simple data movements
    creating, 91–98
  SQL Server Import and Export Wizard, 88, 89–99
    creating simple data movements, 91–98
    transformations, 89
  SSDT (SQL Server Data Tools), 88
  SSIS (SQL Server Integration Services), 87
  tasks, 150
msdb database, 429, 443
MSMQ connection manager, 135
Multicast transformation, 201, 400
Multidimensional mode, for data mining, 670
multidimensional processing, 266, 267
multi-input transformations, 200–201
multi-output transformations, 200–201
multiple configurations, 375
multiple fact tables, dimensions, 8
Multiple Files connection manager, 134
Multiple Flat Files connection manager, 134
multiple flat files, merging, 225
multiple result sets, retaining, 248
multiple tasks, execution of, 517
multi-row transformations, 202–203

N

Name attribute, 590
name columns, 18, 608
Name property, 244, 357
Name property (File System task), 161, 167
Name property (Foreach Loop Editor), 157, 159
names, connection managers, 136
Namespace property, 244
Namespace property (Foreach Loop Editor), 159
naming conventions
  attributes, 21
  dimensions, 17, 21
  DW, 17–18, 21
  LOB applications, 3
  OLAP, 17–18
Navigational Access, 619
nested loop joins, 58
nested tables, in data mining, 671
Net process mode option (CDC), 307
Net with merge process mode option (CDC), 307
Net with update mask process mode option (CDC), 307
NewCode column, in stg.entityname_Leaf table, 609
New Connection Manager option, 355
new execution operation, 458
New output, 290
newRecordCount variable, 271
New Solution Configuration dialog box,, 359

New status, in cleansing project, 648
NEXT VALUE FOR function, 45
nGrams algorithm, 738
No Cache mode (Lookup transformations), 219
No environment references, validation mode, 467
non-blocking transformations, 199, 513
nonclustered indexes, filtered, 57
None logging level, 468
non-equi self joins, 60–61
non-key columns, third normal form, 9
normalization, 5, 145–147, 747
normalized schemas
  measurement of quality, 534
  reporting problems, 5–7
Normalize property, in domains, 639
Notify Operator task, 152
NOT NULL constraint, 186
NotSupported property, TransactionOption, 329, 330
nouns, 567
NULL function, 200, 258
nulls, 400, 654
number series, sequences, 45
numbers, floating point, 590

O

Object Browser, SSIS monitoring reports, 469–470
Object data type, 246, 247
Object Explorer, accessing SSISDB catalog, 506
object hierarchy
  inheritance, 387
  permission inheritance, 483
  variable access, 248
objectives, separation of, 267
objects
  icons in debug environment, 498
  MDS (Master Data Services), 588–589, 589–592
  parameterizing with expressions, 261–262
  SSAS (SQL Server Analysis Services), 132
obsolete information, 533
ODBC connection manager, 135
ODBC destination, 184
ODBC source adapter, 180, 181
OLAP cubes, in data mining, 688
OLAP (online analytical processing), 9, 17–18
OLE DB command, 226
OLE DB Command transformation, 204, 290
OLE DB connection manager, 121, 135
  creating, 140–142
  log providers, 384
  Lookup transformation, 218
  parameterization, 122
OLE DB destination, 184
OLE DB Destination Adapter, column mapping, 214, 230
OLE DB Destination Editor
  column mapping, 187
  Connection Manager tab, 186
  Mappings tab, 186
OLE DB source, 180, 181
OLE DB Source adapter, dynamic SQL, 300–302
OLE DB Source Editor, 364
OLE DB Source Editor dialog box, 182, 191, 345
OLE DB transformation, 517
OLTP applications, SCD problem, 21–24
on-demand execution, 456
  DTExecUI utility, 458, 461–462
  programmatic execution, 458–462
  SQL Server Management Studio, 457–458
  uses for, 457
OnError event, 386, 391, 402
OnError event handler, 343, 344, 345
one-to-many relationships, auditing, 397
one-to-one relationships
  auditing, 396
  schema correctness, 534
OnExecStatusChanged event, 343, 386
OnInformation event, 343, 386
online analytical processing (OLAP), 9
on line transactional processing applications. *See* OLTP applications
OnPostExecute event, 343, 386, 391
OnPostValidate event, 343, 386
OnPreExecute event, 343, 386, 391, 501
OnPreValidate event, 343, 386
OnProgress event, 343, 386
OnQueryCancel event, 343, 386
OnTaskFailed event, 343, 386, 391
OnVariableValueChanged event, 343, 386
OnWarning event, 343, 386
open-world assumption, 531
operating system inspection, 132
operating system, log provider choice, 385
operation, 442
operational applications, auditing, 396

Operation property (File System task), 161, 167
operations
  and execution monitoring, 465–466
  SSISDB reports on, 506
  types of, 441–442
operation status values, 465–466
Operator event property, 386
operators
  batch mode processing, 63
  in expressions, 255
optimizing queries, 60–61
Order Details table, 28
outliers, finding, 688
outputs
  adding column to script component, 713
  error, 317
  for script components, 708–709
  Lookup transformation
    merging with Union All component, 220–221
  SCD transformation, 289–290
overall recurrence frequency, SQL Server Agent jobs, 463
overlapping permissions, 619
OverwriteDestination property (File System task), 161, 167

P

Package Configuration Organizer, possible tasks, 368
package configurations, 367–375
  creating, 369–371
  elements configured, 368
  in Execute Package task, 269
  modifying, 375
  ordering, 375
  parameters, 371
  sharing, 373, 375
  specifying location for, 371–372
  using, 375–377
Package Configurations Organizer, 368–369
Package Configuration Wizard, 369, 372, 373
package deployment model
  package configuration and, 368
  passing package variables in, 371
Package deployment model, 269
package execution
  monitoring, 521–522
  observing, 520–521
  performance tuning, 511–520
  troubleshooting, 498–509
Package ID variable, 400
package-level audit, table schema, 398
package-level connection managers, converting to project-level, 355
package-level parameter, filtering source data with, 364–365
Package name variable, 400
package parameters, 303
package performance
  benchmarking, 518–519
  monitoring, 519
package properties, in package configurations, 368
packages
  CDC (change data capture)
    LSN (log sequence number) ranges, 305
  design-time troubleshooting, 498–499
  ETL (extract-transform-load)
    error flows, 317–321
  executing, 456–470
  execution and monitoring, 473–474
  incremental loads, 312–314
  in deployment model, 440
  Integration Services Logging, 382
  logging, 383–388
  master package concept, 265–266
  parameters, 303
  restartability, 762
  securing, 480–484
  SSIS (SQL Server Integration Services)
    Control Flow Designer, 105–107
    creating, 91–98
    developing in SSDT, 101–108
    editing, 99, 114–120
    ETL (extract-transform-load), 218
    importing into SSIS projects, 112–123
    parameterization, 137
    parameters, 122
    preparing, 162–163
    preparing for incremental loads, 299–316
    processing files, 157–159
    running, 96
    saving, 96
    verifying, 162–163
    viewing files, 98

starting and monitoring, 470–479
validation in SSMS, 470–472
validation of, 466–467
package-scoped connection managers, 136–137
package-scoped errors, log providers, 385
package-scoped variables, 271
as global, 248, 249
default setting, 250
package settings, updating across packages, 367
package status, viewing in Locals window, 502
package templates
preparing, 406–407, 408–409
using, 408, 409–410
package transactions
creating restartable, 336–339, 347
defining settings, 328–330
implementing, 333–335
manual handling, 332–333
transaction isolation levels, 331–332
package validations, 466–467
package variable properties, in package configurations, 368
package variables, interacting with, 702
page compression, 62
parallel execution, 517
parallel queries, 58
parameterization
applicable properties, 241
connection manager
case scenario, 125
connection managers, 137
OLE DB connection manager, 122
of properties, 251
reusability and, 240
SSIS (SQL Server Integration Services), 111, 123
Parameterize dialog box, 358
Parameterize option, 358
parameters
adding in build configurations, 359–360
and running packages, 363
connection managers, 354
defining, 356–358
editing, 357
for connection strings, 363–364
implementing, 363–366
in Execute Package task, 269
mapping to SSIS variables, 301
package, 303
packages, 303
project, 303
project and package, 441
read-only, 242
SSIS (SQL Server Integration Services) packages, 122
uses for, 356
validating, 466–467
values available, 356
vs. variables, 242
Parameters tab, 364, 471, 473
parameter values, for data taps, 507
parameter window, creating parameters in, 357
parent events, 344
parent level, 20
parent object, log settings, 387
parent packages, 269, 328, 330
Parent Package Variable, 370–371
partial-blocking transformations, 199, 513
Partial Cache mode (Lookup transformations), 219
partially denormalized dimensions, 11
partition elimination, 72
Partition function, 72
partition indexes, 72
partitioning, 44–45
fact tables, 74–76
tables
testing, 80
partitioning technique, 739
Partition Processing destination, 184
partition schemes, 72
partitions, loading fact tables, 71–73
partition switching, 72, 76–78
PATH environment variable
in 64-bit installation, 427
SQL Server Agent, 428
pattern distribution, 532
Pending operation status, 466
people, 567
Percentage Sampling transformation, 202, 505, 690
performance
DW
batch processing, 62–64
columnstore indexes, 62–64
data compression, 61–62
indexed views, 58–61
indexing dimensions, 56–59
indexing fact tables, 56–59
queries, 60–61

performance counters, 519–520
Performance logging level, 468
Performance Monitor, 519–520
Performance property value, 518
performance tuning, of package execution, 511–520
Periodically Remove Old Versions property, 439
permission assignments, testing, 489
permission inheritance, 483
permission-related problems, operator ID, 386
permissions
  assigning, 480, 481, 482–484
  Bulk Insert task, 150
  implicit, 619
  testing assignment of, 622–623
  testing security settings, 633
Permissions page, 488
Person.Person table (TK463DW), creating data flow tasks, 190–192
PipelineComponentTime event, 386
PipelineExecutionTrees event, 513
Pivot editor, 203
pivot graphs, 17
pivoting
  attributes, 17
  dynamic, 203
pivoting (data), 146
pivot tables, 17
Pivot transformation, 202
places, 567
Play button, in Data Viewer window, 505
POC projects
  automatic discretization, 18
  shared dimensions, 9–10
point of failure, restarting packages from, 336, 347
policies, data governance, 570
populating entities, 596–599
population completeness, 531
pop-up message window, in debugging, 505
PostExecute method, customization, 724
Precedence Constraint Editor, 118, 119
precedence constraints, 119, 164–169
  Boolean expressions in, 256
  completion, 165
  creating, 118
  expressions and, 259–260
  failure, 165, 167–169
  and multiple tasks, 517
  success, 165
precedence, controlling w/ variables, 502
predefined reports
  benchmarking performance with, 519
  monitoring execution with, 506
predictability, 240
predictions
  and training, 690
  in data mining, 668
predictive models
  efficiency of, 689–690
  in SSIS, 667
PreExecute method, customization, 723
prefix compression, 62
PrepareForExecute method, customization, 722
presentation quality, measurement of, 533
primary data
  overgrowth, 401
  storage, 396
Primary filegroup, creating partition schemes, 74
primary keys, joins, 57
primary table schema, 397
PrimeOutput method, customization, 723
principals, implementing, 481
privacy laws, and completeness measuement, 531
private dimensions, 9–10
problem detection, execution monitoring, 465
problems, reporting w/ normalized schemas, 5–7
procedures, T-SQL data lineage, 73
processes, external, 131, 171
processing files, SSIS packages, 157–159
processing mode options (CDC), 306
ProcessInput method, customization, 723–724
Produce Error task, 345
production data, copying to development, 125
production environments
  SSIS (SQL Server Integration Services) projects, 111
  troubleshooting, 498, 506–508
production server
  SSIS installation choices, 424
  SSIS package deployment, 437
production systems, defining master data, 574
Products dimension
  columns, 66
  creating, 51
product taxonomies, 566
Profiler trace, log providers, 385
profiling data, methods for, 654
Profiling tab, exclamation point tooltip in, 552

programmatic execution, 458–462
Progress tab, execution details, 499–500, 509
project configurations, 358
Project Deployment model, 269, 437, 440
  build configurations in, 358
  moving solutions with, 371
project-level connection managers, 136–137, 354–355
project parameters, 303, 363–364
Project.params window, 356
Project Properties dialog box, 488
projects
  checking for problems, 466
  SSIS (SQL Server Integration Services)
    creating, 103–105
    deploying, 110–124
    development environments, 111
    importing SSIS packages into, 112–123
    production environments, 111
    properties, 139
  validation in SSMS, 470–472
Projects object, 482
proof-of-concept (POC) projects. *See* POC projects
properties
  Execute SQL Task, 116
  Expressions, 302
  File System task, 161, 167, 168
  Foreach Loop Editor, 157, 159
  member, 18, 18–19
  setting with expressions, 303
  SqlCommand
    modifying for data flow task, 303
    setting expressions for, 303–304
  SSIS (SQL Server Integration Services) projects, 139
  updating, 363
  ValidateExternalMetadata, 183
Properties property, 374
Properties To Export, 373
Properties window, editing with, 357
Property Expression Editor, 160, 361
property expressions, 259
  applying, 361–362
  evaluation of, 362
Property Expressions Editor, 302
Property Overrides grid, 473
Property Pages dialog box, 359, 360–361
property parameterization, by explicit assignment, 251
property paths
  assigning at run time, 468
  vs. parameters, 242
ProvideComponentProperties, customization, 719
pruning method, 739
Publish And Validate group, 626
Publish button, 626
purposes, separation of, 267

Q

queries, 3
  aggregating columns in AdventureWorksDW2012
    database tables, 59–62
  batch mode processing, 63
  clustered indexes, 56
  covered, 56
  creating, 626
  data mining, 132
  data quality measurement, 531
  in data mining predictions, 672
  indexed views, 58–61
  multiple lookups, 517
  non-equi self joins, 60–61
  optimizing, 60–61
  for package execution, 506
  parallel, 58
  profiling data with, 654–656
  Snowflake schema, 11
  specifying
    compared to copying data, 94
  Star schema, 11
  testing indexing methods, 79
Query Optimizer
  columnstore indexes, 62–64
  hash joins, 57
  joins, 57
  partition elimination, 72

R

RaiseChangeEvent property, 244
random samples
  achieving, 690
  performing, 690–693
Ratcliff/Obershelp algorithm, 738
Raw File destination, 184
Raw File source, 180

RDBMS (relational database management system), data integrity, 569–570
ReadCommitted property, IsolationLevel, 331
READ_OBJECTS permission, 482
Read-Only permission, 618
Read-only property, 159, 244
ReadOnlyVariables, 701, 708
READ permission, 482, 484
ReadUncommitted property, IsolationLevel, 331
ReadWriteVariables, 701, 708
real-world objects, correct representation of, 534
Rebuild Index task, 152
REBUILD option, 56
reconstruction, rows, 62
records
  inserting into fact tables, 285
  updating, 226
Recordset destination, 184
recovery models
  Bulk Logged, 43
  of DQS databases, 555
  Full, 43
  Simple, 43
  transaction logs, 42–43
recursive hierarchies, 590–591
Redirect Rows option, 318
red squiggly line, in spell checker, 640
reference data providers, 541
Reference Data Services (RDS), 638
  configuring, 549–550
  logging setting, 552
reference dictionary, for term lookup, 682
reference relation, in measuring completeness, 531–532
references
  adding, 702
  columns
    errors, 208
    resolving, 207–208
  in custom component development, 718
Registry Entry, package configuration type, 370
Regular Expressions, data extraction with, 731
regular users group, 617–618
ReinitializeMetaData, customization, 721
relational database management system (RDBMS). *See* RDBMS (relational database management system)
relational databases
  master data, 567
  nulls in, 654
relation completeness, 532, 654
relations, 589
ReleaseConnections method, customization, 724
Reliability And Performance Monitor, 384, 386
Remarks column, 50
Reorganize Index task, 152
REORGANIZE option, 56
RepeatableRead property, IsolationLevel, 331
reports
  aggregate functons, 29–30
  reporting problems with normalized relational schemas, 5–7
  slow DW reports, 79
  for troubleshooting packages, 506
Required property, 329, 330, 332, 339, 357
Resolve References editor, 207–208, 208
resource planning, execution monitoring, 465
RetainSameConnection property, 332
retention period, 402, 470
Retention Period (days) property, 439
Retention window operation, 442
Retrieve file name property (Foreach Loop Editor), 157
return on investment (ROI). *See* ROI (return on investment)
reusability
  defined, 240
  improving, 480, 492
  master data, 568
  master package concept and, 266
  maximizing, 242
  of custom components, 711
Reuse Existing button, 373
ROI (return on investment), MDM, 569
roles
  data stewards, 570
  Windows Web Server
    MDS (Master Data Services), 577
root causes, finding, 536
row-based (synchronous) transformations, 513
row compression, 61
Row Count data flow, 398
row counts, described in debugging, 499
Row Count transformation, 200, 399, 505
row-level auditing, 396–397, 398, 402
row-level security, 483
rows
  buffers, 178
  capture count of, 505
  capturing with data taps, 508
  data flow task, 177

displaying in data viewer, 503–504
error paths, 319–321
filtering, 626
filtering with package-level parameters, 364–365
identifying problems, 503
nulls, 400
reconstruction, 62
shrinking size of, 515
standardizing data in, 758
storage in buffers, 512
in transformations, 513
unique identifiers, 401
row sampling, testing, 694
Row Sampling transformation, 202, 505, 690
row sets
retaining multiple, 248
reusing, 248
storing, 243, 247
variables and, 243
Run64BitRuntime setting, 137
Running operation status, 466
run time engine, parallel execution in, 517
run-time methods
customization, 719, 721–725
of script component, 710

S

Sales Fact table, 28, 29
same-table row-level audits, 396
samples
in data mining, 687
selecting appropriate, 689–692
testing row sampling, 694
SaveCheckpoints property, 337, 340
SByte data type, 246
scalar values, variables and, 243
SCD (Slowly Changing Dimension), 21–24, 284, 285–290
SCD transformation, 285–290
implementing SCD logic, 293–298
modifying to include set-based updates, 296–298
outputs, 289–290
updating dimensions, 290
SCD Wizard, 287
schedules, in SQL Server Agent, 463
SCHEMABINDING option, 59
schemas
database
AdventureWorksDW2012 sample database, 14–15
Dimensional Models, 6
documentation of, 534
measurement of quality, 534
measuring completeness, 534
measuring correctness, 534
normalized relational
reporting problems, 5–7
Snowflake, 3, 4, 9–11
case scenarios, 34–38
hierarchies, 21
queries, 11
Star, 3, 4, 7–10
case scenarios, 34–38
dimensions tables, 8–9
fact table, 7–8
hierarchies, 21
queries, 11
scope
ambiguity resolution, 250
changing, 250
function of, 248
overriding variables with, 250
Scope property, 244
score, in term extraction, 681
Script Component, 205, 707–711
coding, 709–711
configuring, 708–709
custom components and, 716
debugging, 503, 505
reusing row sets, 248
using, 712–715
Script Component Editor, 708
Script Component transformation, 205, 291
scripting
debugging, 702
uses for, 699
ScriptMain class, 701, 702
ScriptObjectModel class, 702
Script page, 701
Script properties, 705
script task, 154, 700–707
coding, 702–703
configuring, 701
custom components and, 716
debugging, 505
parameterization with, 254
reading profiling results with, 704–706
reusing row sets, 248

Script Task Editor, 701
Script transformation, 517
search space
  reducing, 747
  reduction techniques, 739
secondary keys. *See* primary and secondary keys
securables, 481, 482
security
  and storing connection information, 371, 375
  complex settings, 617
  governance policies, 570
  MDS security, 633
  of master data, 616–624
  of SSIS packages, 480–484
  overlapping permissions, 619
  SSIS component installation, 426, 431
  SSISDB catalog, 423, 437
  SSIS solutions, 455
  SSIS (SQL Server Integration Services) deployment, 111
security model, Integration Services, 422
segments, 63
Select Destination wizard page, 447
Select New Scope pane, 250
Select Properties To Export page, 374
Select Source wizard page, 448
semi-structured data, 566
Send Mail task, 149
Sensitive property, 357
Sequence container, 156, 329
sequences, 45
  CACHE option, 46
  compared to identity columns, 45–46
  creating
    syntax, 46
  SQL Server, 45
Serializable property, IsolationLevel, 331
series, sequences, 45
Server default parameter value, 356
server variables, 441
Server-wide Default Logging Level property, 439, 506
service accounts, 426–427
Service Accounts Step-by-Step Guide (Microsoft), 426
set-based updates, 229
  updating Customer Dimension table, 229–231
  updating dimensions, 292–293
Set Breakpoints dialog box, 500–501
Set Parameter Value dialog box, 488
Set Query Parameters dialog box, 300–301, 364
shared dimensions, 8–9
  DimProduct, 8
  POC projects, 9–10
  SSAS (SQL Server Analysis Services), 9
Show Explorer button, 626
Show System Variables option, 245
Show Variables Of All Scopes option, 249
Shrink Database task, 152
shrinking databases, 153
side-by-side upgrade, 428
Silverlight 5
  installing, 586
  Master Data Manager, 577
Simil algorithm, 738
similarity scores, 759
similarity threshold, 747, 758
similarity threshold parameter, 740
similar strings, finding, 738
simple data movements, 88
  creating, 91–98
  planning, 89–99
Simple recovery model, 43, 555
simple sequential integers, surrogate keys, 45
Simple Table Quick Profiling Form dialog box, 658
Single data type, 246
64-bit environments, 427
  migration, 429
  tools available, 431
Slowly Changing Dimension. *See* SCD
Slowly Changing Dimension transformation, 204
SMO connection manager, 135
SMTP connection manager, 135
Snapshot property, IsolationLevel, 331
Snippets.txt file, 706
Snowflake schema, 3, 4, 9–11
  case scenarios, 34–38
  hierarchies, 21
  queries, 11
soft dimensions
  defined, 531
  measurement of, 533, 535, 537
solution configurations, 358
solution deployment, 437. *See also* SSIS project deployment
Solution Explorer, 103–104, 355, 356
  assigning project properties in, 360
  build configurations in, 359

sorted input, data flow tips for, 516
sorting neighborhood technique, 739
sort tables, staging area, 227–228
Sort transformation, 202, 224–225, 513, 516
source adapters
  debugging, 505
  execution trees, 513–514
  optimization of, 517
Source Assistant
  adding data flow source adapters, 181–182
  creating data flow source adapters, 181
source columns, mapping of, 556
SourceConnectionOLEDB connection manager, 121
source control, SSIS templates, 407
source data
  filtering with package-level parameter, 364
  incremental loads, 299
SourceID event property, 386
SourceName event property, 386
Source Order Details table, 28
Source Orders Header table, 28
source speed, 518
source systems, data quality solutions, 537
source tables, selecting, 95
source threads, default value of, 517
SourceVariable property (File System task), 161, 167
space
  databases
    calculating requirements, 43
  data files, 43
specific duration, job schedule, 463
Speller property, in domains, 639
spelling checker, 640
Split Transformations, 694
sp_sequence_get_range system procedure, 45
SQLCAT (SQL Server Customer Advisory Team), 44
SqlCommand property
  modifying for data flow task, 303
  setting expressions for, 303–304
SQL, reading data, 299–304
SQL Server
  autogrowing, 43
  autoshrinking, 43
  batch mode processing, 63
    operators, 63
  creating DW databases, 42–44
  data compression, 61–62
  indexed views, 58–61
  joins, 58–59
  log provider, 384
  maintenance, 132
  native auditing, 399
  sequences, 45
  SSIS option, 423
  SSIS solution deployment, 421
  transactions in, 328
SQL Server 2005/2008, upgrade paths, 429
SQL Server 2012
  data viewer display, 503
  package configuration type, 370
  solution deployment, 437
  SSIS server, 437
  SSIS Toolbox, 179
SQL Server 2012 Database Engine Services, DQS installation, 542
SQL Server 2012 Master Data Services (MDS). *See* MDS (Master Data Services)
SQL Server 2012 Upgrade Advisor, 429
SQL Server Agent, 462–464
  cleanup operation, 439
  development vs. production environment, 425
  PATH environment variable, 428
  schedules, 463
  SSIS execution automation, 455
  SSIS package creation, 475–476
  SSIS package scheduling, 477–479
  SSIS process execution, 455
  vs. master package concept, 464
SQL Server Analysis Services. *See* SSAS
SQL Server Business Intelligence Development Studio (BIDS), 102
SQL Server Business Intelligence suite, in data mining, 669
SQL Server Compact destination, 184
SQL Server Compact Edition connection manager, 135
SQL Server Configuration Manager
  service accounts, 427
  shortcut location, 434
SQL Server configuration, options available, 372
SQL Server Customer Advisory Team (SQLCAT), 44
SQL Server Database Engine
  accessing SSISDB catalog, 506
  development vs. production environmnet, 425
  SSIS functionality, 425
SQL Server databases, creating, 48–49
SQL Server Data Tools. *See* SSDT (SQL Server Data Tools)

SQL Server destination, 184
SQL Server Import and Export Wizard, 88, 89–99, 430, 431
  64-bit installation, 427
  creating simple data movements, 91–98
  editing SSIS packages created in, 114–116
  transformations, 89
  T-SQL scripts, 116
  viewing SSIS package files, 98–99
SQL Server Installation Center, 428–429, 432
SQL Server instance, CLR integration, 445
SQL Server Integration Services 11.0 properties, 435
SQL Server Integration Services Deployment Wizard, 430, 442
SQL Server Integration Services Execute Package Utility, 427, 430, 431
SQL Server Integration Services Package Installation Utility, 430
SQL Server Integration Services Package Upgrade Wizard, 430
SQL Server Integration Services Package Utility, 427, 430, 431
SQL Server Integration Services Project Conversion Wizard, 430
SQL Server Integration Services (SSIS). *See* SSIS (SQL Server Integration Services)
SQL Server Management Studio (SSMS)
  administering DQS with, 553–555
  management capabilities, 423
  on-demand execution, 457–458
  package deployment, 448–449
  project deployment, 442
  SSISDB settings, 438–439
  SSIS validation in, 469–471
  vs. Deployment Wizard, 442–443
SQL Server Management Tools, DQS database administration, 542
SQL Server permissions, 426, 429
SQL Server Profile, log provider, 384
SQL Server Profiler trace, log providers, 384
SQL Server service accounts, 426, 435
SQL Server services, account recommendations, 426
SQL Server Setup, installing MDS, 578
SQL Server table, storing package configurations, 372
SQL statements, set-based update logic, 292–293
SSAS (SQL Server Analysis Services), 9, 132
  additivity of measures, 30–31
  data mining engine, 668
  data mining installation, 670
  data mining support, 670
  discretizing attributes, 17
  for frequency of distributions, 656
  multidemensional auditing, 398
  storage types, 30
SSDT (SQL Server Data Tools), 88
  32- vs. 64-bit environments, 427
  Complete installation option, 427
  connection manager scope, 136–137
  creating SSIS (SQL Server Integration Services) projects, 103–105
  developing SSIS packages, 101–108
  development vs. production environment, 425
  IDE (integrated development environment), 110
  implementing event handlers, 344
  Log Events window, 513
  vs. Deployment Wizard, 442–443
ssis_admin role, 481
  default permission, 483
  folder creation, 484
SSIS component installation
  64-bit vs. 32-bit environments, 427
  development vs. production, 424–425, 433
  hardware/software requirements, 425
  preparing, 424
  security considerations, 426–427
  SQL Server Installation Center, 428–429, 432
  SSIS server functionalities, 423
  SSIS tools, 430
  upgrading SSIS, 429–430
SSIS Data Flow Engine, 512–514
SSIS data flow mining model, 675
SSIS Data Profiling task, 656–657
  using, 657–659
  vs. cleansing, 657
SSISDB catalog, 459
  as deployment target, 480
  benchmarking performance with, 518–519
  configuration, 438–439
  execution data, 465–466
  functions of, 423
  native auditing capabilities, 399
  package storage in, 467
  parameter access, 441
  parameter value control, 441, 507
  SSIS package deployment, 437, 438, 444
  timeout property, 467

troubleshooting with, 506
upgrades and, 430
SSISDB configuration, project deployment, 444–446
SSISDB objects, project deployment, 440–442
SSISDB permissions, 481, 484–489
SSISDB principals, 481
SSISDB securables, 481, 482
SSISDB security, 455, 480–484
SSIS Designer
Control Flow tab, 336
Execution Results tab, 499
log providers, 384
Progress tab, 499–500
viewing execution information, 500
SSIS development
automated execution of, 239
package templates, 406–408
SSIS DQS Cleansing component, 646
SSIS DQS Cleansing transformation, 541
SSIS fuzzy transformations
algorithm for, 756–757
built-in filtering, 740
finding similar strings with, 738
implementing, 756–763
versions of, 758–759
SSIS installation paths, 428
SSIS Legacy Service
availability of, 423
earlier SSIS packages, 443
manual configuration, 429
SSIS logs, correlating w/ audit data, 401–402
SSIS managed API, programmatic execution with, 458, 460–461
SSIS management assembly, 460
SSIS packages
creating dynamic, 378
custom code for, 699–730
from previous SQL versions, 443
in 64-bit environments, 428
mining model for, 675
storage, 421, 422
uses in data mining, 671
SSIS processes, automatic vs. manual, 455
SSIS project deployment
Deployment Wizard vs. SSDT/SSMS, 442–443, 446, 448–449
development vs. production environments, 421, 437
SSISDB catalog, 438, 444
SSISDB configuration, 438–439, 444–446
SSISDB objects, 440–442
SSIS runtime
for SSIS execution, 456
SSIS processes, 421
SSIS server
development vs. production environment, 425
functionalities, 423
SSIS solutions, 421–423. *See also* SSIS project deployment
advantages of separate packages, 266
storage, 423
SSIS (SQL Server Integration Services), 87
administration operations, 132
CDC implementation, 304–307, 308–316
connection managers, 133–144
containers, 155–163
Foreach Loop, 156
For Loop, 156
Sequence, 156
data flow task. *See* data flow task
data management operations compared to LOB, 87
data mining queries, 132
data profiling, 132
data types, 187–189
development, 110
email, 132
event handling, 342–344
external processes, 131
extract-transform-load. *See* ETL (extract-transform-load)
file systems, 131
FTP access, 131
logging, 13
logging/auditing, 382
measuring performance, 386
operating system inspection, 132
package execution monitoring, 506–508
packages
Control Flow Designer, 105–107
creating, 91–98
developing in SSDT, 101–108
editing, 99, 114–120
ETL (extract-transform-load), 218
importing into SSIS projects, 112–123
parameterization, 137
parameters, 122
preparing, 162–163

preparing for incremental loads, 299–316
processing files, 157–159
running, 96
saving, 96
verifying, 162–163
viewing files, 98
parallel execution in, 517
parameterization, 111, 123
predictive models in, 667
projects
creating, 103–105
deploying, 110–124
development environments, 111
production environments, 111
properties, 139
set-based updates, 225–227
SQL Server Import and Export Wizard, 88
creating simple data movements, 89–99
viewing SSIS package files, 98–99
SSAS (SQL Server Analysis Services), 132
SSDT (SQL Server Data Tools), 88
creating SSIS projects, 103–105
developing packages, 101–108
IDE (integrated development environment), 110
SSIS Toolbox, 105–107
tasks. *See* tasks (control flow)
transaction isolation levels, 331–332
transaction support, 328–329
updating dimensions, 290–293
SSIS (SQL Server Integration Services) Toolbox
data flow task components, 178–179
event handlers, 342, 344, 345
script task, 701
SQL Server 2012, 179
SSIS tools, stand-alone tools, 430
SSIS transformations, in test mining, 679
stage maintenance processes, 267
stage preparation, 265
staging area
join tables, 227–228
sort tables, 227–228
Sort transformation, 225
Staging Batch Interval setting, 608
staging process, 607–609, 613
staging tables, 633
data set storage in, 243
implementing, 44
populating, 607
stale data, 533
Standard Reports, 506
Star schema, 3, 4, 7–10
case scenarios, 34–38
dimensions tables, 8–9
fact table, 7–8
hierarchies, 21
measurement of quality, 534
queries, 11
StartTime event property, 387
statements
INSERT, 73
MERGE, 293
Execute SQL task, 226
set-based update logic, 292–293
UPDATE
Execute SQL task, 226
StateProvince entity, 597
statistical sampling, 689, 690
stg.entityname_Consolidated, 607
stg.entityname_Leaf, 607
stg.entityname_Leaf table, 608
stg.entityname_Relationship, 607
Stopping operation status, 466
storage
disk storage
filegroups, 44
of SSIS solutions, 423
of rows in buffers, 512
SSAS (SQL Server Analysis Services), 30
SSIS packages, 421
String data type, 246, 247
string functions, 257
string length distribution, 532
string similarity algorithms, 738
subroutines, implementing, 291–292
subscription view
creating, 613–615
exporting master data with, 609
Subscription views metadata, 606
subsets, 567
Succeeded operation status, 466
success constraints, 165
Suggested status, in cleansing projects, 648
SUM aggregate function, 29, 399
Supported property, TransactionOption, 330
surrogate keys, 23–24
fact tables, 28
simple sequential integers, 45
Type 2 SCD, 284

suspicious values, discovering, 655
swapping to disk storage, 516, 520
switching data, partition switching, 72, 76–78
SynchronousInputID property, 709
synchronous (non-blocking) transformations, 513
synchronous output, 709
synonyms, creating, 640
Syntax algorithms property, 639
syntax, creating sequencies, 46
sysadmin fixed server role, 543
System Administration
  access to, 618
  changing administrator, 617
System namespace, 251
system variables, 243, 251
  adding to data flow, 382
  to specify file location, 371
System.Windows.Forms namespace, 505

T

Table Lock check box, 185
Table Or View - Fast Load option, 185
tables
  aggregated
    indexed views, 58–61
  aggregating columns in AdventureWorksDW2012 sample database, 59–62
  audit data, 396–397, 397
  audting, 13
  cdc.captured_columns system table, 305
  cdc.change_tables system table, 305
  cdc.ddl_history system table, 305
  cdc.index_columns system table, 305
  cdc.lsn_time_mapping system table, 305
  cdc.stg_CDCSalesOrderHeader_CT, 305
  cleansing errors, 344
  columns
    updating, 229–231
  complete auditing, 395
  cross join of, 737
  Customer dimension
    loading data with SCD tranformation, 293–296
    set-based update logic, 292–293
  Customer Dimension
    updating, 229–231
  Customers Dimension
    loading data to DW, 209–212
    loading into DW, 212–214
  defining location of, 373
  destination
    truncating, 117
  dimension
    history, 287
    selecting, 287
  dimensions, 3, 8–9
    conformed, 8
    private, 9–10
    shared, 8–9
  entities, 589
  extracting data from views and loading into, 91–98
  fact, 3, 567
    activity of measures, 29–30
    AdventureWorksDW2012 sample database, 32–33
    business key, 28–29
    columns, 28–29, 47–48
    columnstore indexes, 64
    creating, 52–54
    design, 27–33
    early-arriving facts, 285, 288
    foreign keys, 28
    granularity, 12
    implementing, 47–48
    incremental loads, 299–316
    inferred members, 47
    inserting unresolved records, 285
    loading, 71–73
    many-to-many relationships, 30–32
    partitioning, 44–45, 74–76
    partition switching, 76–78
    Star schema, 7–8
    surrogate keys, 28
  FactInternetSales
    adding partitioning column to, 74
  InternetSales
    adding foreign keys, 75
    creating columnstore index, 75
    loading data, 75
  InternetSales fact
    columnstore indexes, 68–69
    data compression, 68–69
  InternetSales Fact
    columns, 67

- join
  - staging area, 227–228
- joined
  - indexed views, 58–61
- lineage, 13
- LOB applications, 5
- mdm.tblUser table, 617
- multiple fact
  - dimensions, 8
- normalization, 5
- Order Details, 28
- package templates, 407
- partitioning, 44–45
  - testing, 80
- Person.Person (TK463DW)
  - creating data flow tasks, 190–192
- pivot, 17
- records
  - updating, 226
- rows
  - reconstruction, 62
- Sale Facts, 29
- Sales Fact, 28
- sort
  - staging area, 227–228
- source
  - selecting, 95
- Source Order Details, 28
- Source Orders Header, 28
- staging
  - implementing, 44
- staging tables, 243, 633
- stg.CDCSalesOrderHeader
  - enabling CDC, 304–305
- stg.entityname_Leaf table, 608

table schema, package-level audit, 398
tabular mode, data mining and, 670
tabular storage, in SSAS, 30
target variables, in data minig, 668
task editors, property expressions, 362
Task ID variable, 400
Task name variable, 400
TaskResult property, 703
tasks
- breakpoint icons, 501
- parameterization of, 254
- variable access and, 248

tasks (control flow), 145, 147–155
- administration, 151
- analysis services, 153
- Analysis Services Execute DDL, 153
- Analysis Services Processing, 153
- Back Up Database, 152
- Bulk Insert, 150
- CDC Control, 149, 305
- Check Database Integrity, 152
- custom, 154–155
- data flow, 177
  - creating, 178–180, 190–192
  - defining data destination adapters, 184–187
  - defining data source adapters, 180–183
  - rows, 177, 178
  - SSIS (SQL Server Integration Services) components, 178–179
- Data Flow, 150
- Data Mining Query, 153
- data movement, 150
- data preparation, 148
- Data Profiling, 148
- Execute Package, 149
- Execute Process, 149
- Execute SQL, 150, 226
  - set-based update logic, 292, 292–293
- Execute SQL Server Agent Job, 152
- Execute T-SQL Statement, 152
- Expression, 149
- File System, 148, 161
  - general settings, 161, 167, 168
- FTP, 148
- grouping, 155
- History Cleanup, 152
- maintenance, 151–152
- Maintenance Cleanup, 152
- Message Queue, 149
- Notify Operator, 152
- precedence constraints, 164–169
- Rebuild Index, 152
- Reorganize Index, 152
- Script, 154
- Send Mail, 149
- Shrink Database, 152
- Transfer Database, 151
- Transfer Error Messages, 151
- Transfer Jobs, 151
- Transfer Logins, 151

- Transfer Master Stored Procedures, 151
- Update Statistics, 152
- Web Service, 148
- WMI Data Reader, 149
- WMI Event Watcher, 149
- workflow, 148–149
- XML, 148

task-scoped errors, 385
task transactions, defining settings, 328–330
taxonomies, product, 566
templates, SSIS package, 407–408
term-based relations, creating, 640
Term-Based Relations tab, 644
term columns
- in term extraction, 681
- in term lookup, 682

Term Extraction Transformation, 205, 667, 679–681, 682
Term Frequency/Inverse Document Frequency (TFIDF), 681
term frequency (TF), 681
Term Lookup Transformation, 205, 667, 679, 681, 684
test sets, 690
text columns
- in term lookup, 681
- retrieving key terms from, 680

text file log provider, 384
text mining
- defined, 679
- performing, 682–686
- transformations in SSIS, 670
- with transformations, 667

theoretical models, measuring compliance with, 534
things, 567
third normal form, non-key columns, 9
third-party data providers, 660
32-bit environments, 427
- migration, 429
- tools available, 431

threads, allocation w/ execution trees, 513
timeliness, measuring, 533
tokenizer, 756
token transformations., 757
token weights, 757
top-level audits, 396
Toyoda, Sakichi, 536
Toyota Motor Corporation, 536
trace files, log providers, 384
training
- of data in data mining, 668
- of predictive models, 689

training sets, 690
transactional data, 566, 567
transaction inheritance, 328
transaction isolation levels, 331
transaction logs, recovery models, 42–43
TransactionOption property, 328, 330, 339
transactions
- defined, 328, 626
- defining settings, 328–330
- implementing, 333–335
- manually handling, 332–333
- rollback, 333, 347

Transact-SQL
- administering SQL with, 553–554
- data quality measurement, 531
- executing data taps, 507
- profiling data with, 654
- programmatic execution with, 458–459
- SSISDB settings, 438
- SSIS management, 423

Transfer Database task, 151
Transfer Error Messages task, 151
Transfer Jobs task, 151
Transfer Logins task, 151
Transfer Master Stored Procedures task, 151
Transformation Editor, 212
transformation processes, 267
transformations, 177
- advanced data preparation, 204
- backpressure mechanism for, 514
- blocking, 199, 203
- buffer architecture and, 512–513
- Cache Transform
  - Lookup transformation, 223–224
- Conditional Split, 214, 291
- custom data flow, 226
- data flow tips, 516
- Derived Column, 206, 212
- design-time debugging, 505
- loading data, 209–210
- logical-row, 199–200
- Lookup, 200–201
  - cache modes, 218–219
  - Cache Transform transformation, 223–224
  - case sensitivity, 220
  - composite joins, 220
  - connection managers, 218
  - ETL (extract-transform-load), 218–224

merging outputs with Union All component, 220–221
replacing Union All transformation, 228–229
rows with no matching entities, 220
Merge Join, 211
multi-input, 200–201
multi-output, 200–201
multi-row, 202–203
non-blocking, 199
OLE DB Command
updating dimensions, 290
partial-blocking, 199
resolving column references, 207–208
SCD (Slowly Changing Dimension), 285–290
implementing slowly changing dimension logic, 293–298
modifying to include set-based updates, 296–298
outputs, 289–290
updating dimensions, 290
Script Component, 291
script component as, 708
selecting, 198–204
Sort, 224–225
SQL Server Import and Export Wizard, 89
types of, 512
Union All
replacing multiple Lookup transformation outputs, 228–229
transformation speed, 518
transform step, in data mining, 669, 670
Traverse subfolders property (Foreach Loop Editor), 157
troubleshooting
automation of, 239
execution monitoring, 465
of package execution, 498–508
TRUNCATE TABLE statement, 71
Truncate Update Table task, 339
truncating destination tables, 117
truncation error, repairing, 508–509
trust, measuring, 533
T-SQL commands, data lineage, 73
T-SQL HASHBYTES function, 291
T-SQL INNER JOIN operator, 739
T-SQL procedures, data lineage, 73
T-SQL queries
exporting master data with, 609
in data mining, 688
vs. SSAS, 656
T-SQL scripts, SQL Import and Export Wizard, 116
T-SQL statements, 332. *See also* statements
tuples, 531, 532, 654
Type 1 SCD, 22, 292–293
Type 1 SCD attribute, 284, 290–293
Type 2 SCD, 22
set-based updates, 292–293
surrogate key, 284
Type 2 SCD attribute, 284, 290–293
Type 3 SCD, 23

U

UDM cubes, as measurement tools, 669
UInt16 data type, 246
UInt32 data type, 246
UInt64 data type, 246
Unchanged output, 290
undirected algorithms, in data mining, 668
Unicode compression, 62
Unicode strings
in term lookup, 681
retrieving key terms from, 680
unintended usage, 533
Union All component, merging Lookup transformation outputs, 220–221
Union All transformation, 201, 228–229, 513
unique identification, lack of, 735
units, atomic, 328
unknown values (NULLs)
in measuring compliance, 534
in measuring completeness, 531–532
unmanaged connections, 702
unpivoting (data), 146
Unpivot transformation, 202
Unspecified property, IsolationLevel, 331
unstructured data, 566
unstructured sources, integration of, 533
unsupported data types, storage of, 247
Update permission, 618
UPDATE statement, Execute SQL task, 226
Update Statistics task, 152
updating
columns, 229–231
dimensions, 290–293
checking attribute changes, 291–292
set-based updates, 292–293

set-based updates, 225–227, 229
updating Customer Dimension table, 229–231
upgrades, SSIS, 428–430, 451
Usage type property (File System task), 161, 168
US Counties, default KB, 639
Use A Single Column To Show Current And Expired Records option, 287
Use case-sensitive term extraction, 681
Use leading values property, 639
UseParentSetting property, 387
user accounts, installing MDS, 577
User and Group Permissions, access to, 618
user-defined variables, 243, 251–252
User namespace, 251
User name variable, 400
user perception, 533
Use Start And End Dates To Identify Current And Expired Records option, 287
US Last Name domain, 639
US Places domain, 639
US State domain, 639

V

ValidateExternalMetadata property, 183
Validate method, customization, 720, 722
Validate Project dialog box, 471
validation, 146
following data import, 608
in SSMS, 470–472
limits of, 472
validation errors, identifying, 382
Validation Report prompt, 472
validation results, 720
validations
and execution monitoring, 466–467
SSISDB reports, 506
validation step, described in debugging, 499
Validation Timeout property, 439, 467
Valid From column, 284
Valid To column, 284
value comparisons, 256
value completeness, 532
Value Inclusion, profiling of, 657
Value property, 244, 357, 502
Value property (Foreach Loop Editor), 159
values
determining with expressions, 255
direct assignment of, 467
Value type property (Foreach Loop Editor), 159
VARCHAR(MAX) columns, 516
variableA variable, 249
variableB variable, 249
Variable Grid Options dialog box, 244–245, 249
variables
creation of, 244
data types, 245–248
in data mining, 668–669
learning how to use, 702
mapping parameters to, 301
measurement in data mining, 688
namespaces for, 251
naming, 250
vs. parameters, 242
properties of, 244
scope of, 248–250
types of, 243
uses for, 241
Variables collection class, 711
variable settings, Foreach Loop Editor, 159
Variables page, 486
Variables pane, 244, 250
Variables property, 374, 701, 702
variable values
changing, 502
debugging script components, 503
viewing in Locals window, 502
verbose logging, 469
Verbose logging level, 468, 506
verbs, 567
verifying
SSIS (SQL Server Integration Services) packages, 162–163
Version flags metadata, 606
Version ID variable, 400
versioning, master data, 568, 569
Version Management, access to, 618
viewing
SSIS (SQL Server Integration Services) package files, 98
views
catalog, 63–64
extracting data from, 94
extracting data from and loading into tables, 91–98
selecting, 95

virtual accounts, SSIS installation, 426
virtual memory allocation, buffer size and, 515
Visual Studio 2010
  build configurations in, 358
  building custom components in, 718
Visual Studio Tools for Applications (VSTA). *See* VSTA
Visual Studio Tools for Applications (VSTA) IDE, 503
volatility, of master data, 568
VSTA Solution Explorer, 702, 709
VSTA (Visual Studio Tools for Applications), 154

W

warnings, in debugging, 499
Warn logging level, 552
watch windows, adding variables to, 502
web service source, creating, 731
web services, scripting for, 708
Web Service task, 148
windows
  Add SSIS Connection Manager, 192
  Column Mappings, 95
  Create Website, 584
Windows 7, account recommendations, 426
Windows Application Log, logging errors, 411
Windows Azure MarketPlace DataMarket, 541, 551
Windows Control Panel, component changes, 428
Windows Event Log
  configuring logging, 390
  log provider, 384
Windows Event Viewer, 384
Windows operating system, Performance Monitor, 519
Windows PowerShell, programmatic execution with, 458, 459–460, 474
Windows Server 2008/2008 R2, account recommendations, 426
Windows user
  checking a name, 621
  creating, 620
Windows user groups
  in administration, 618
  permissions, 633
Windows Vista, account recommendations, 426
Windows Web Server roles, MDS, 577
wizards
  Slowly Changing Dimension
    selecting dimension tables, 287
  SQL Server Import and Export Wizard, 88, 89–99
    creating simple data movements, 91–98
    editing SSIS packages created in, 114–116
    transformations, 89
    T-SQL scripts, 116
    viewing SSIS package files, 98–99
WMI connection manager, 135
WMI Data Reader task, 149
WMI Event Watcher task, 149
worker threads, required by execution trees, 513
workflow
  debugging, 503
  designing in advance, 268–269
workflow tasks, 148–149
workloads, separation of, 267
work threads, parallel execution of, 517

X

XML Configuration File
  package configuration type, 370
  sharing, 372
XML data, 566–567
XML data type columns, measuring completeness, 532
XML documents, for multiple result sets, 248
XML File Configuration, creating, 371–372, 376–377
XML file, log provider, 384
XML source, 180
XML task, 148, 330