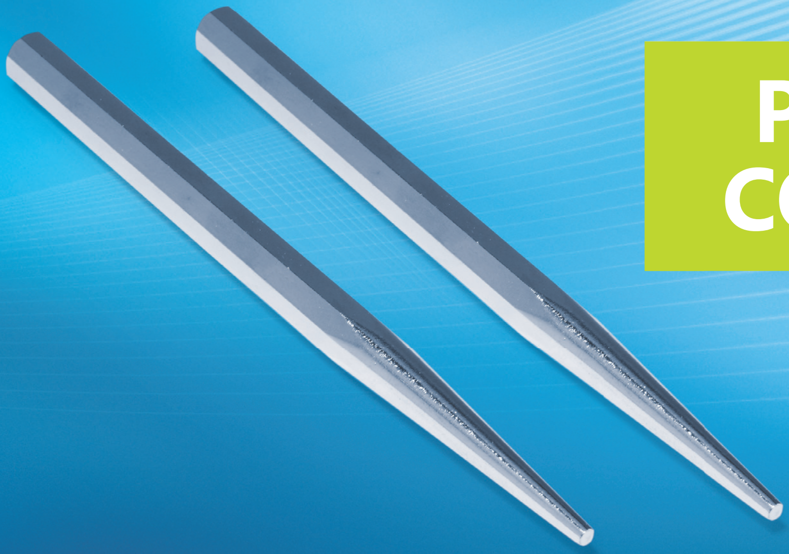
Microsoft

SPECIAL EXCERPT II
Complete book available Spring 2012

Introducing Microsoft® SQL Server® 2012



PREVIEW CONTENT

Ross Mistry and Stacia Misner

About the Authors

Ross Mistry is a Principal Enterprise Architect with Microsoft, working out of the Microsoft Technology Center in Silicon Valley. He designs solutions for Microsoft largest customers and specializes in SQL Server high availability, appliances, consolidation, virtualization, and private cloud.

Ross is also an author, community champion, and seasoned architect. He has a tremendous amount of experience designing and deploying technology solutions for Internet startups and fortune 100 organizations located in the Silicon Valley. He is known in the world-wide community for his expertise in SQL Server, Windows, Exchange, Virtualization, and Private Cloud. Ross frequently speaks at technology conferences around the world and has published many books, whitepapers, and magazine articles. His recent books include *Introducing SQL Server 2008 R2* (Microsoft Press), *Windows Server 2008 R2 Unleashed* (SAMS) and *SQL Server 2008 Management and Administration* (SAMS).

You can follow him on Twitter @RossMistry.

Stacia Misner is a consultant, educator, mentor, and author specializing in Business Intelligence solutions since 1999. During that time, she has authored or co-authored multiple books about BI. Stacia provides consulting and custom education services through Data Inspirations and speaks frequently at conferences serving the SQL Server community. She writes about her experiences with BI at *blog.datainspirations.com*, and tweets as @StaciaMisner.

Contents

Chapter 2

High Availability and Disaster Recovery Enhancements 5

SQL Server AlwaysOn: An Integrated Solution 5

AlwaysOn Availability Groups 7

Understanding Concepts and Terminology 8

Configuring Availability Groups 13

Monitoring Availability Groups with the Dashboard 15

Active Secondaries 16

Read-Only Access to Secondary Replicas 17

Backups on Secondary 17

AlwaysOn Failover Cluster Instances (FCI) 18

Support for Deploying SQL Server 2012 on Windows Server Core 20

SQL Server 2012 Prerequisites for Server Core 21

SQL Server Features Supported on Server Core 22

SQL Server on Server Core Installation Alternatives 22

Additional High Availability and Disaster Recovery Enhancements 23

Support for Server Message Block 23

Database Recovery Advisory 23

Online Operations 24

Rolling Upgrade and Patch Management 24

Chapter 3

Scalability and Performance 25

Columnstore Index Overview 25

Columnstore Index Fundamentals and Architecture 26

How is Data Stored When using a Columnstore Index? 26

How do Columnstore Indexes Significantly Improve the Speed of Queries? 28

Columnstore Index Storage Organization 30

Columnstore Index Support and SQL Server 2012 30

Columnstore Index Restrictions 31

Columnstore Index Design Considerations and Loading Data 32

When to build a columnstore index 32

What tables should have columnstore indexes 33

When not to build a columnstore index 33

Loading new Data 33

Creating a Columnstore Index 34

Creating a columnstore index by using the SQL Server Management Studio 35

Creating a columnstore index by using Transact-SQL 36

Using Columnstore Indexes 37

Using Hints with a Columnstore Index 39
Columnstore Index Observations and Best Practices 39
Summary 40
Chapter 4
Security Enhancements 41
Security Manageability Improvements 42
Default Schema for Groups 42
User Defined Server Roles 43
Audit Enhancements 46
Audit Supported on all SKUs 46
Improved Resilience 46
User-Defined Audit Event 50
Record Filtering 50
T-SQL Stack Information 51
Database Authentication Enhancements 51
Enabling Contained Databases 52
Creating Users 53
Contained Database Authentication Security Concerns 54
Additional Security Enhancements 54
Cryptography Changes 54
SharePoint Active Directory 55
Provisioning Enhancements 55
New Permissions 56
Chapter 6
Integration Services 57
Developer Experience 57
Add New Project Dialog Box 57
General Interface Changes 58
Getting Started Window 60
SSIS Toolbox 61
Shared Connection Managers 62
Scripting Engine 63
Expression Indicators 63
Undo and Redo 64
Package Sort By Name 64
Status Indicators 64
Control Flow 65
Expression Task 65
Execute Package Task 66

Data Flow ... 67
- Sources and Destinations ... 67
- Transformations ... 70
- Column References ... 71
- Collapsible Grouping ... 73
- Data Viewer ... 74

Change Data Capture Support ... 75
- CDC Control Flow ... 76
- CDC Data Flow ... 77

Flexible Package Design ... 78
- Variables ... 79
- Expressions ... 79

Deployment Models ... 80
- Supported Deployment Models ... 80
- Project Deployment Model Features ... 82
- Project Deployment Workflow ... 83

Parameters ... 86
- Project Parameters ... 86
- Package Parameters ... 87
- Parameter Usage ... 87
- Post-Deployment Parameter Values ... 89

Integration Services Catalog ... 91
- Catalog Creation ... 92
- Catalog Properties ... 93
- Environment Objects ... 95

Administration ... 98
- Validation ... 98
- Package Execution ... 98
- Logging and Troubleshooting Tools ... 100
- Security ... 101

Package File Format ... 102

Chapter 7

Data Quality Services ... 103

Data Quality Services Architecture ... 103
- Data Quality Services Server ... 103
- Data Quality Client ... 104

Knowledge Base Management ... 105
- Domain Management ... 105
- Knowledge Discovery ... 117
- Matching Policy ... 120

Data Quality Projects ... 124
  Cleansing Projects ... 125
  Matching Projects ... 127
Administration ... 129
  Activity Monitoring ... 129
  Configuration ... 131
Integration ... 134
  Integration Services ... 134
  Master Data Services ... 137

Chapter 8

Master Data Services ... 138

Getting Started ... 138
  Upgrade Considerations ... 138
  Configuration ... 139
Master Data Manager ... 140
  Explorer ... 140
  Integration Management ... 144
  User and Group Permissions ... 146
Model Deployment ... 147
MDS Add-In for Excel ... 149
  Installation of the MDS Add-In ... 149
  Master Data Management ... 149
  Model Building Tasks ... 154
  Shortcut Query Files ... 156
  Data Quality Matching ... 157
Miscellaneous Changes ... 159
  SharePoint Integration ... 159
  Metadata ... 159
  Bulk Updates and Export ... 159
  Transactions ... 160
  PowerShell ... 160

Chapter 2
High Availability and Disaster Recovery Enhancements

Microsoft SQL Server 2012 delivers significant enhancements to well-known, critical capabilities like high availability and disaster recovery. These enhancements promise to assist organizations in achieving the highest mission-critical confidence to date. Server Core support along with breakthrough features like AlwaysOn Availability Groups, active secondaries, and key improvements to features such as failover clustering now offer organizations a range of accommodating options to achieve maximum application availability and data protection for SQL Server instances and databases within a datacenter and across datacenters.

With SQL Server's heavy investment in AlwaysOn, this chapter's goal is to bring readers up-to-date with the high availability and disaster recovery capabilities that are fully integrated into SQL Server 2012.

SQL Server AlwaysOn: An Integrated Solution

Every organization's success and service reputation is built on ensuring that its data is always accessible and protected. In the IT world, this means delivering a product that achieves the highest level of availability and disaster recovery while minimizing data loss and downtime. With the previous versions of SQL Server, organizations achieved high availability and disaster recovery by using technologies such as failover clustering, database mirroring, log shipping and peer-to-peer replication. Although organizations achieved great success with these solutions, they were tasked with combining these native SQL Server technologies to achieve their business requirements affiliated with their Recovery Point Objective (RPO) and Recovery Time Objective (RTO).

Figure 2-1 illustrates a very common high availability and disaster recovery strategy used by organizations with the previous versions of SQL Server. This strategy includes failover clustering to protect SQL Server Instances within each datacenter combined with asynchronous database mirroring to provide disaster recovery capabilities for mission critical databases.

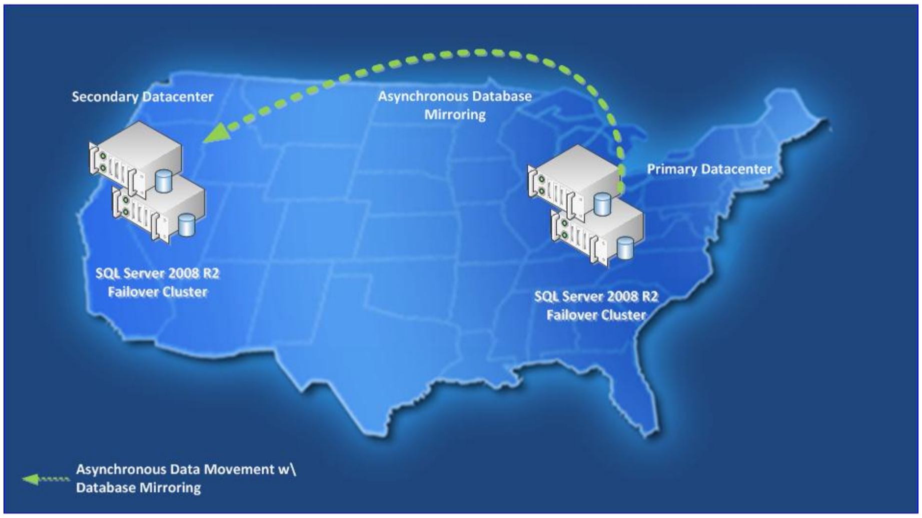

Secondary Datacenter
Asynchronous Database Mirroring
Primary Datacenter
SQL Server 2008 R2 Failover Cluster
SQL Server 2008 R2 Failover Cluster
Asynchronous Data Movement w\ Database Mirroring

FIGURE 2-1 Achieving high availability and disaster recovery with failover clustering combined with database mirroring in SQL Server 2008 R2.

Likewise, the high availability and disaster recovery deployment for organizations that either required more than one secondary datacenter or who did not have shared storage incorporated synchronous database mirroring with a witness within the primary datacenter combined with Log Shipping for moving data to multiple locations. This deployment strategy is illustrated in Figure 2-2.

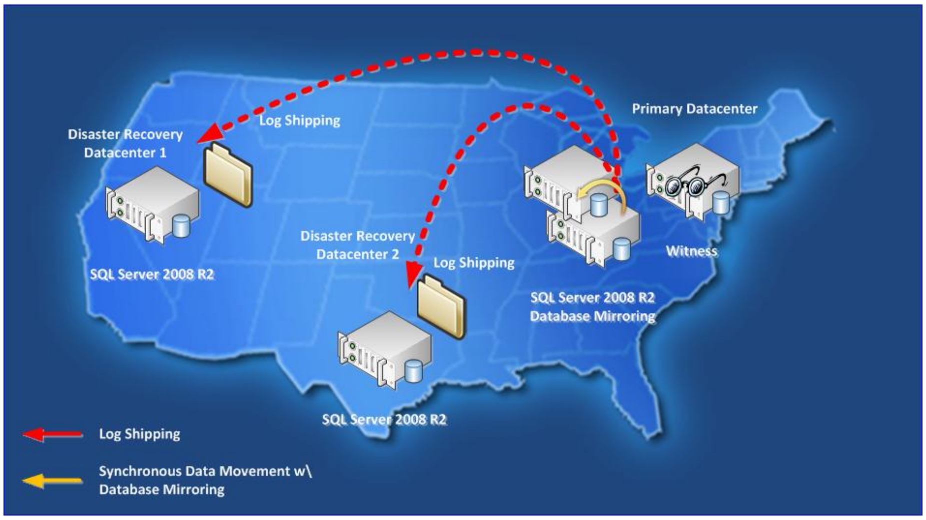

Disaster Recovery Datacenter 1
Log Shipping
Primary Datacenter
SQL Server 2008 R2
Disaster Recovery Datacenter 2
Log Shipping
Witness
SQL Server 2008 R2 Database Mirroring
SQL Server 2008 R2
Log Shipping
Synchronous Data Movement w\ Database Mirroring

FIGURE 2-2 Achieving high availability and disaster recovery with database mirroring combined with Log Shipping in SQL Server 2008 R2.

Both Figures 2-1 and 2-2 reveal successful solutions for achieving high availability and disaster _recovery. However, the approach to these solutions was fragmented instead of seamless which warranted changes. In addition, with organizations constantly evolving, it was only a matter of time until they voiced their own concerns and sent out a request for more options and changes.

One concern for many organizations was directed at database mirroring. Database mirroring is a great way to protect databases; however, the solution is a one-to-one mapping, making multiple secondaries unattainable. When confronted with this situation, many organizations reverted to Log Shipping as a replacement for database mirroring because it supports multiple secondaries. Unfortunately, organizations encountered limitations with Log Shipping because it did not provide zero data loss or automatic failover capability. Concerns were also experienced by organizations working with failover clustering because they felt that their shared storage devices, such as a SAN, could be a single point of failure. Similarly, many organizations thought that from a cost perspective their investments were not being used to their fullest potential. For example, the passive servers in many of these solutions were running idle. Finally, many organizations wanted to offload reporting and maintenance tasks from the primary database servers, which was not an easy task to achieve.

SQL has evolved to answer many of these concerns and this includes an integrated solution called *AlwaysOn*. AlwaysOn *Availability Groups* and *AlwaysOn Failover Cluster Instances* are new features, introduced in SQL Server 2012, that are rich with options and promise the highest level of availability and disaster recovery to its customers. At a high level, *AlwaysOn Availability Groups* are used for database protection and offer multi-database failover, multiple secondaries, active secondaries, and integrated HA management. On-the-other-hand, *AlwaysOn Failover Cluster Instances* are tailored towards instance-level protection, multi-site clustering and consolidation while consistently providing flexible failover polices and improved diagnostics.

AlwaysOn Availability Groups

The AlwaysOn Availability Groups provide an enterprise-level alternative to database mirroring and give organizations the ability to automatically or manually failover a group of databases as a single unit with support for up to four secondaries. The solution provides zero data loss protection and is flexible. It can be deployed on local storage or shared storage, and it supports both synchronous and asynchronous data movement. The application failover is very fast, supports automatic page repair, and the secondary replicas can be leveraged to offload reporting and a number of maintenance tasks such as backups.

Take a look at Figure 2-3 which simply illustrates an AlwaysOn Availability Group deployment strategy that includes one primary replica and three secondary replicas.

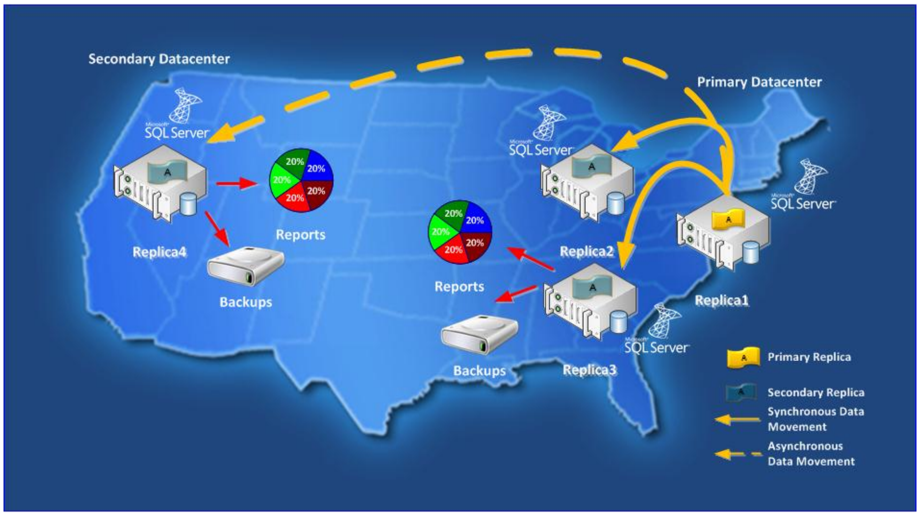


FIGURE 2-3 Achieving high availability and disaster recovery with AlwaysOn Availability Groups.

In this figure, synchronous data movement is used to provide high availability within the primary datacenter and asynchronous data movement is used to provide disaster recovery. Moreover, secondary replica 3 and replica 4 are employed to offload reports and backups from the primary replica.

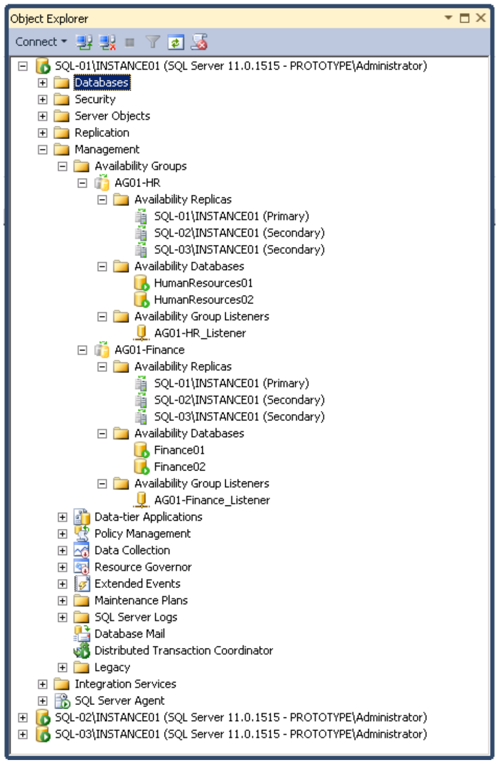
It is now time to carry out a deeper dive on AlwaysOn Availability Groups through a review of the new concepts and terminology associated with this breakthrough capability.

Understanding Concepts and Terminology

Availability groups are built on top of Windows Failover Clustering and support both shared and non-shared storage. Depending on an organization's Recovery Point Objective (RPO) and Recovery Time Objective (RTO) requirements, availability groups can use either an asynchronous-commit availability mode or a synchronous-commit availability mode to move data between primary and secondary replicas. Availability groups include built in compression and encryption as well as a support for file-stream replication and auto page repair. Failover between replicas are either automatic or manual.

When deploying AlwaysOn Availability Groups, the first step is to deploy a Windows Failover Cluster. This is completed by using the Failover Cluster Manager Snap-In within Windows Server 2008 R2. Once the Windows Failover Cluster is formed, the remainder of the Availability group configurations is completed in SQL Server Management Studio. When using the Availability Group Wizards to configure availability groups, SQL Server Management Studio automatically creates the appropriate services, applications, and resources in Failover Cluster Manager, hence the deployment is much easier for database administrators who are not familiar with failover clustering.

Now that the fundamentals of the AlwaysOn Availability Group has been laid down, the most natural question that follows is how is an organization's operations are enhanced with this feature. Unlike database mirroring which supports only one secondary, AlwaysOn Availability Groups support one primary replica and up to four secondary replicas. Availability groups can also contain more than one availability database. Equally appealing, it is possible to host more than one availability group within an implementation. As a result, it is possible to group databases with application dependencies together within an availability group and have all the availability databases seamlessly failover as a single cohesive unit as depicted in Figure 2-4.



```
SQL-01\INSTANCE01 (SQL Server 11.0.1515 - PROTOTYPE\Administrator)
    Databases
    Security
    Server Objects
    Replication
    Management
        Availability Groups
            AG01-HR
                Availability Replicas
                    SQL-01\INSTANCE01 (Primary)
                    SQL-02\INSTANCE01 (Secondary)
                    SQL-03\INSTANCE01 (Secondary)
                Availability Databases
                    HumanResources01
                    HumanResources02
                Availability Group Listeners
                    AG01-HR_Listener
            AG01-Finance
                Availability Replicas
                    SQL-01\INSTANCE01 (Primary)
                    SQL-02\INSTANCE01 (Secondary)
                    SQL-03\INSTANCE01 (Secondary)
                Availability Databases
                    Finance01
                    Finance02
                Availability Group Listeners
                    AG01-Finance_Listener
        Data-tier Applications
        Policy Management
        Data Collection
        Resource Governor
        Extended Events
        Maintenance Plans
        SQL Server Logs
        Database Mail
        Distributed Transaction Coordinator
        Legacy
    Integration Services
    SQL Server Agent
SQL-02\INSTANCE01 (SQL Server 11.0.1515 - PROTOTYPE\Administrator)
SQL-03\INSTANCE01 (SQL Server 11.0.1515 - PROTOTYPE\Administrator)
```

FIGURE 2-4 Dedicated availability groups for Finance and HR availability databases.

In addition, as shown in Figure 2-4, there is one primary replica and two secondary replicas with two availability groups. One of these availability groups is called Finance and it includes all the Finance databases; the other availability group is called HR and it includes all the Human Resources databases. The Finance Availability Group can failover independently of the HR availability group and unlike database mirroring, all availability databases within an availability group failover as a single unit. Moreover, organizations can improve their IT efficiency, increase performance, and reduce total cost of ownership with better resource utilization of secondary/passive hardware because these secondary replicas can be leveraged for backups and read-only operations such as reporting and maintenance. This is covered in the "Active Secondaries" section later in this chapter.

Now that you have been introduced to some of the benefits that the AlwaysOn Availability Groups offer for an organization, take the time to get a stronger understanding of the AlwaysOn Availability Group concepts and how the new capability operates. The concepts covered include:

- Availability Replica Roles
- Failover and Synchronization Modes
- Data Synchronization Options
- Connection Mode in Secondaries
- Availability Group Listener

Availability Replica Roles

Each AlwaysOn Availability Group is comprised of a set of two or more failover partners that are referred to as availability replicas. The availability replicas can consist of either a primary role or a secondary role. It is worth noting that there can be a maximum of four secondaries and of these four secondaries only a maximum of two secondaries can be configured to use the synchronous-commit availability mode synchronous-commit availability mode availability mode.

The roles affiliated with the availability replicas in AlwaysOn Availability Groups follow the same principals as the legendary Sith rule of two doctrines in the Star Wars[1] saga. In Star Wars, there can only be two Sith at one time, a master and an apprentice. Similarly, a SQL Server instance in an availability group can only be a primary replica or a secondary replica. At no time can it be both because the role swapping is controlled by Windows Server Failover Cluster (WSFC).

Each of the SQL Server instances in the availability group is hosted on either a SQL Server Failover Cluster Instance (FCI) or a stand-alone instance of SQL Server 2012. Each of these instances resides on different nodes of a WSFC. WSFC is typically used for providing high availability and disaster recovery for well-known Microsoft products. As such, availability groups use WSFC as the underlying mechanism to provide inter-node health detection, failover coordination, primary health detection, and distributed change notifications for the solution.

Each availability replica hosts a copy of the availability databases in the availability group. Since there are multiple copies of the databases being hosted on each availability replica, there isn't a prerequisite for utilizing shared storage like there was in the past when deploying traditional SQL Server Failover Clusters. On the flip side, when using non-shared storage, an organization must keep in mind that storage requirements increase depending on the number of replicas it plans on hosting.

Data Synchronization Modes

To move data from the primary replica to the secondary replica, each mode uses either synchronous-commit availability mode or asynchronous-commit availability mode. Give consideration to the

[1] Trademark of Lucasfilm Ltd.

following items when selecting either option:

- When using the *synchronous-commit* mode, a transaction is committed on both replicas to guarantee transactional consistency. This, however, means increased latency. As such, this option might not be inappropriate for partners who don't share a high-speed network or reside in different geographical locations.
- The *asynchronous-commit* mode commits transactions between partners without waiting for the partner to write the log to disk. This maximizes performance and is well suited for disaster-recovery solutions.

Availability Groups Failover Modes

When configuring AlwaysOn Availability Groups, database administrators can choose from two failover modes when swapping roles from the primary to the secondary replicas. For those of you who are familiar with Database Mirroring, the failover modes in order to obtain high availability and disaster recovery are very similar to the modes in Database Mirroring. The two AlwaysOn failover modes modes available when using the New Availability Group wizard include:

- **Automatic Failover** This Replica uses synchronous-commit availability mode and supports both automatic failover and manual failover between the replica partners. A maximum of up to two failover replica partners are supported when choosing automatic failover.
- **Manual Failover** This Replica uses synchronous or asynchronous commit availability mode and supports only manual failovers between the replica partners.

Connection Mode in Secondaries

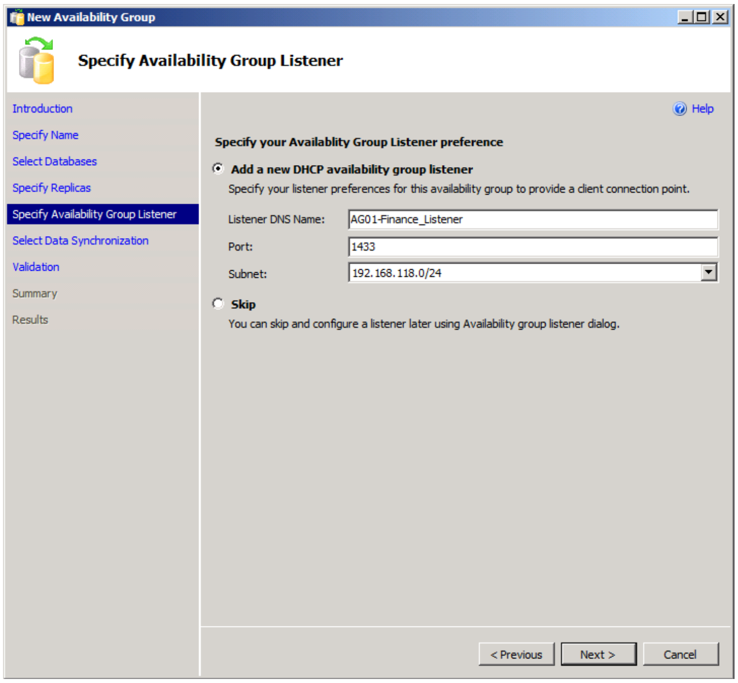
As indicated earlier, each of the secondaries can be configured to support read-only access for reporting or other maintenance tasks such as backups. During the final configuration stage of the AlwaysOn Availability Groups, database administrators decide on the connection mode for the secondary replicas. There are three connections modes available:

- **Disallow connections** In the secondary role, this availability replica does not allow any connections.
- **Allow only read-intent connections** In the secondary role, this availability replica allows only read-intent connections.
- **Allow all connections** In the secondary role, this availability replica allows all connections for read access, including connections running with older clients.

Availability Group Listeners

The Availability Group Listener provides a way of connecting to an availability group via a virtual network name that is bound to the primary replica. Applications can specify the network name affiliated with the Availability Group Listener in connection strings. After the availability group fails over from the primary replica to a secondary replica, the network name directs connections to the new primary replica. The Availability Group Listener concept is very similar to a Virtual SQL Server Name when using failover clustering; however, with an Availability Group Listener, there is a virtual network name for each availability group, whereas with SQL Server failover clustering, there is one virtual network name for the instance.

You can specify your Availability Group Listener preferences when using the Create a New Availability Group Wizard in SQL Server Management Studio or you can manually create or modify an Availability Group Listener after the availability group is created. Alternatively, you can use Transact-SQL to create or modify the listener too. Notice in Figure 2-5 that each Availability Group Listener requires a DNS name, an IP Address, and a Port such as 1433. Once the Availability Group Listener is created, a server name and an IP address cluster resource are automatically created within Failover Cluster Manager. This is certainly a testimony to the availability group's flexibility and tight integration with SQL Server as the majority of the configurations are done within SQL Server.



Specify your Availablity Group Listener preference

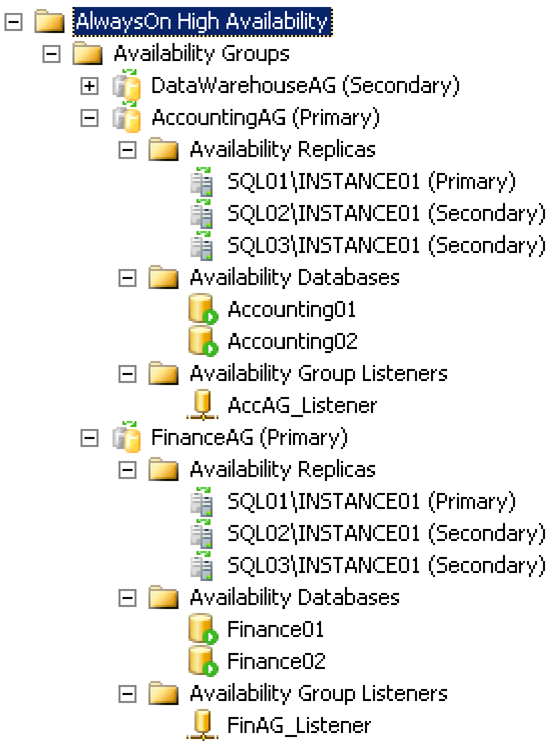
(●) Add a new DHCP availability group listener
Specify your listener preferences for this availability group to provide a client connection point.

Listener DNS Name: AG01-Finance_Listener
Port: 1433
Subnet: 192.168.118.0/24

( ) Skip
You can skip and configure a listener later using Availability group listener dialog.

FIGURE 2-5 Specifying Availability Group Listener Properties.

It is worth stating that there is a one-to-one mapping between Availability Group Listeners and availability groups. This means you can create one Availability Group Listener for each availability group. However, if more than one availability group exists within a replica, it is possible to have more than one Availability Group Listener. For example, there are two availability groups shown in Figure 2-6; one is for the Finance availability databases and the other is for the HR availability databases. Each availability group has its own Availability Group Listener that clients and applications connect to.



```
⊟ AlwaysOn High Availability
    ⊟ Availability Groups
        ⊞ DataWarehouseAG (Secondary)
        ⊟ AccountingAG (Primary)
            ⊟ Availability Replicas
                SQL01\INSTANCE01 (Primary)
                SQL02\INSTANCE01 (Secondary)
                SQL03\INSTANCE01 (Secondary)
            ⊟ Availability Databases
                Accounting01
                Accounting02
            ⊟ Availability Group Listeners
                AccAG_Listener
        ⊟ FinanceAG (Primary)
            ⊟ Availability Replicas
                SQL01\INSTANCE01 (Primary)
                SQL02\INSTANCE01 (Secondary)
                SQL03\INSTANCE01 (Secondary)
            ⊟ Availability Databases
                Finance01
                Finance02
            ⊟ Availability Group Listeners
                FinAG_Listener
```

FIGURE 2-6 Illustrating two Availability Group Listeners within a replica.

Configuring Availability Groups

When creating a new availability group, a database administrator needs to specify an availability group name such as AvailablityGroupFinance, and then select one or more databases to partake in the availability group. The next step involves first specifying one or more instances of SQL Server to host secondary availability replicas and then specifying your Availability Group Listener preference. The final step is selecting the data synchronization preference and connection mode for the secondary replicas. These configurations are conducted with the New Availability Group Wizard or with Transact-SQL PowerShell scripts.

Prerequisites

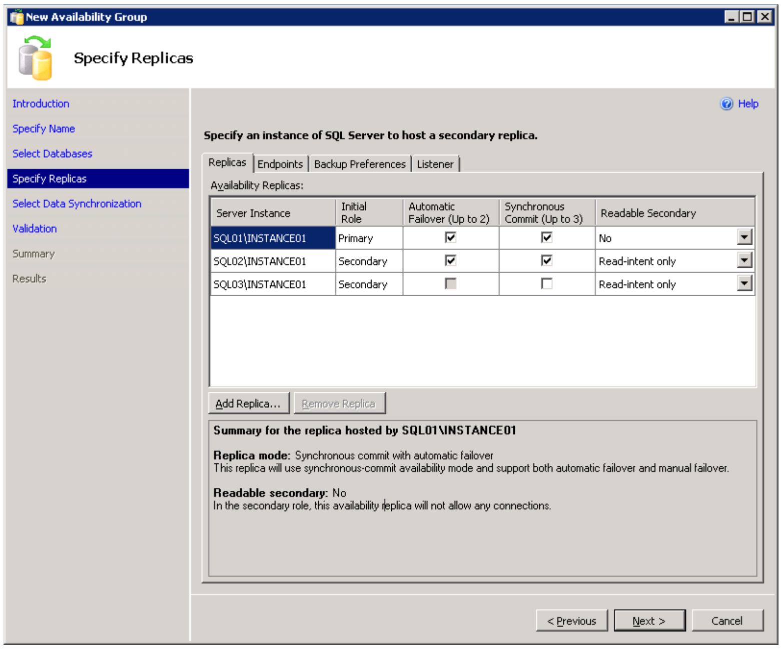
To deploy AlwaysOn Availability Groups, the following prerequisites must be met:

- All computers running SQL Server, including the servers that will reside in the disaster recovery site, must reside in the same Windows-based domain.
- All SQL Server computers must partake in a single Windows Server failover cluster even if the servers reside in multiple sites.

- A Windows Server failover cluster must be formed.
- AlwaysOn Availability Groups must be enabled on each server.
- All the databases must be in full recovery mode.
- A full backup must be conducted on all databases before deployment.

Deployment Examples

Figure 2-7 illustrates the Specify Replicas page when using the New Availability Group Wizard. In this example, there are three SQL Server instances in the availability group called Finance: SQL01\Instance01, SQL02\Instance01, and SQL03\Instance01. SQL01\Instance01 is configured as the Primary Replica whereas SQL02\Instance01 and SQL03\Instance01 are configured as secondaries. SQL01\Instance01 and SQL02\Instance01 support Automatic Failover with Synchronous data movement, whereas SQL-03\Instance01 uses Asynchronous-commit availability mode and only supports a forced failover. Finally, SQL01\Instance01 does not allow read-only connections to the secondary, whereas, and SQL02\Instance01 and SQL03\Instance01 allow Read-Intent connections to the secondary. SQL01\Instance01 and SQL02\Instance01 reside in a primary datacenter for high availability within a site and SQL03\Instance01 resides in the disaster recovery datacenter and will be brought online manually in the event the primary datacenter becomes unavailable.



Specify an instance of SQL Server to host a secondary replica.

Availability Replicas:

Server Instance	Initial Role	Automatic Failover (Up to 2)	Synchronous Commit (Up to 3)	Readable Secondary
SQL01\INSTANCE01	Primary	☑	☑	No
SQL02\INSTANCE01	Secondary	☑	☑	Read-intent only
SQL03\INSTANCE01	Secondary	☐	☐	Read-intent only

Summary for the replica hosted by SQL01\INSTANCE01

Replica mode: Synchronous commit with automatic failover
This replica will use synchronous-commit availability mode and support both automatic failover and manual failover.

Readable secondary: No
In the secondary role, this availability replica will not allow any connections.

FIGURE 2-7 Specifying the SQL Server instances in the availability group.

One thing becomes vividly clear from Figure 2-7 and the preceding example: there are many different deployment configurations available to satisfy any organization's high availability and disaster recovery requirements. See Figure 2-8 for additional deployment alternatives such as:

- Non Shared Storage Local, Regional and Geo Target
- Multi-site Cluster with another Cluster as DR
- 3 Node Cluster with similar DR target
- Secondary Targets for Backup, Reporting and DR

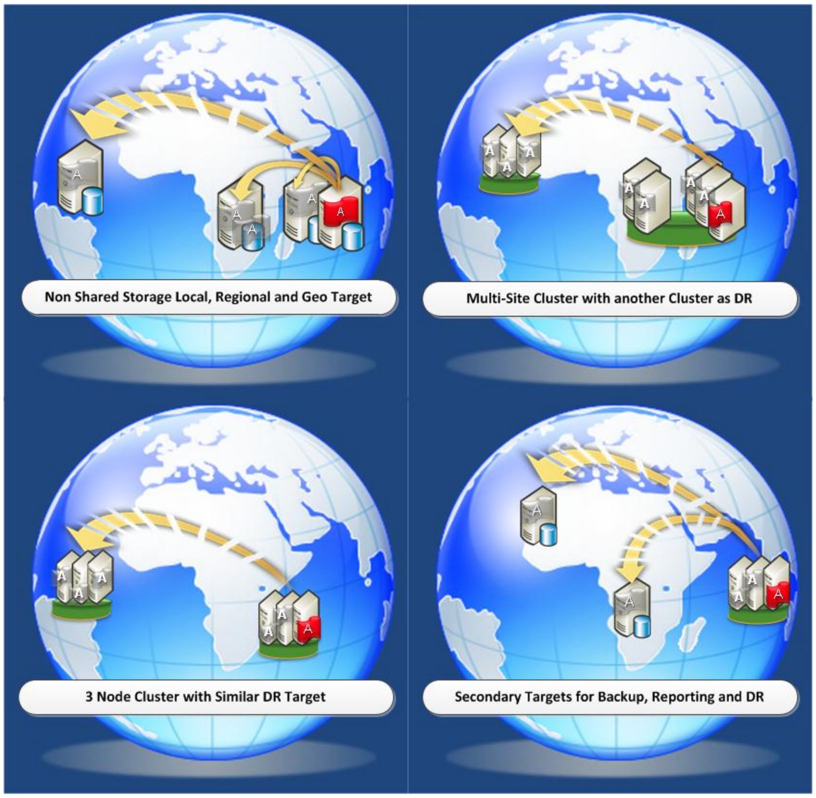
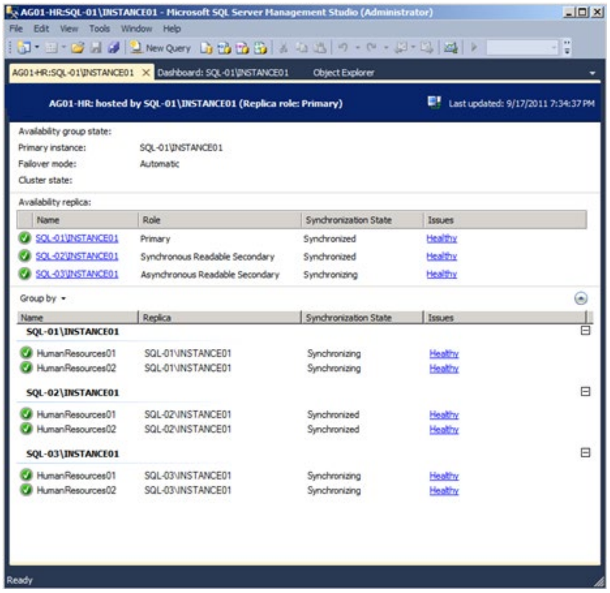

Non Shared Storage Local, Regional and Geo Target
Multi-Site Cluster with another Cluster as DR
3 Node Cluster with Similar DR Target
Secondary Targets for Backup, Reporting and DR

FIGURE 2-8 Additional AlwaysOn Deployment Alternatives.

Monitoring Availability Groups with the Dashboard

Administrators have an opportunity to leverage a new and remarkably intuitive manageability dashboard in in SQL Server 2012 to monitor availability groups. The dashboard, as shown in Figure 2-9 reports the health and status associated with each instance and availability database in the availability group. Moreover, the dashboard displays the specific replica role of each instance and provides synchronization status. If there is an issue or more information on a specific event is required, a database administrator can click the Availability Group State, Server Instance name, or health status hyperlinks for additional information. The dashboard is launched by right-clicking the Availability Groups Folder and selecting Show Dashboard.


AG01-HR:SQL-01\INSTANCE01 - Microsoft SQL Server Management Studio (Administrator)
File Edit View Tools Window Help
New Query
AG01-HR:SQL-01\INSTANCE01
Dashboard: SQL-01\INSTANCE01
Object Explorer
Group by
Ready

AG01-HR: hosted by SQL-01\INSTANCE01 (Replica role: Primary)

Last updated: 9/17/2011 7:34:37 PM

Availability group state:
Primary instance: SQL-01\INSTANCE01
Failover mode: Automatic
Cluster state:

Availability replica:

Name	Role	Synchronization State	Issues
✓ SQL-01\INSTANCE01	Primary	Synchronized	Healthy
✓ SQL-02\INSTANCE01	Synchronous Readable Secondary	Synchronized	Healthy
✓ SQL-03\INSTANCE01	Asynchronous Readable Secondary	Synchronizing	Healthy

Name	Replica	Synchronization State	Issues
SQL-01\INSTANCE01			
✓ HumanResources01	SQL-01\INSTANCE01	Synchronizing	Healthy
✓ HumanResources02	SQL-01\INSTANCE01	Synchronizing	Healthy
SQL-02\INSTANCE01			
✓ HumanResources01	SQL-02\INSTANCE01	Synchronized	Healthy
✓ HumanResources02	SQL-02\INSTANCE01	Synchronized	Healthy
SQL-03\INSTANCE01			
✓ HumanResources01	SQL-03\INSTANCE01	Synchronizing	Healthy
✓ HumanResources02	SQL-03\INSTANCE01	Synchronizing	Healthy

FIGURE 2-9 Monitoring availability groups with the new Availability Group dashboard.

Active Secondaries

As indicated earlier, many organizations communicated to the SQL Server team their need to improve IT efficiency by optimizing their existing hardware investments. Specifically, organizations hoped their production systems for passive workloads could be used in some other capacity instead of remaining in an idle state. These same organizations also wanted reporting and maintenance tasks offloaded from production servers because these tasks negatively impacted production workloads. With SQL Server 2012, organizations can leverage the AlwaysOn Availability Group capability to configure a secondary replica, also referred to as Active Secondaries, to provide read-only access to databases affiliated with an availability group.

All read-only operations on the secondary replicas are supported by row versioning and are automatically mapped to snapshot isolation transaction level, which eliminates reader/writer contention. In addition, the data in the secondary replicas is near real time. In many circumstances, data latency between the primary and secondary databases should be within seconds. It is worth noting that the latency of log synchronization impacts data freshness.

For organizations, active secondaries are synonymous with performance optimization on a primary replica and increases to overall IT efficiency and hardware utilization. 2012

Read-Only Access to Secondary Replicas

Recall that when configuring the connection mode for secondary replicas, you can Disallow Connections, Allow Only Read-Intent Connections, and Allow All Connections. Allow Only Read-Intent Connections and Allow All Connections both provide read-only access to secondary replicas. The Disallow Connections alternative does not allow read-only access as implied by its name.

Now let's look at the major differences between Allow Only Read-Intent Connections and Allow All Connections. The Allow Only Read-Intent Connections option allows connections to the databases in the secondary replica when the Application Intent connection property is set to Read-only in the SQL Server Native Client. When using the Allow All Connection settings, all client connections are allowed independent of the Application Intent property. What is the Application Intent property in the connection string? The Application Intent property declares the application workload type when connecting to a server. The possible values are Read-only and Read Write. Commands that try to create or modify data on the secondary replica will fail.

Backups on Secondary

Backups of availability databases participating in availability groups can be conducted on any of the replicas. Although backups are still supported on the primary replica, log backups can be conducted on any of the secondaries. It is worth noting that this is independent of the replication commit mode being used—synchronous-commit or asynchronous-commit. Log backups completed on all replicas form a single log chain, as shown in Figure 2-10.

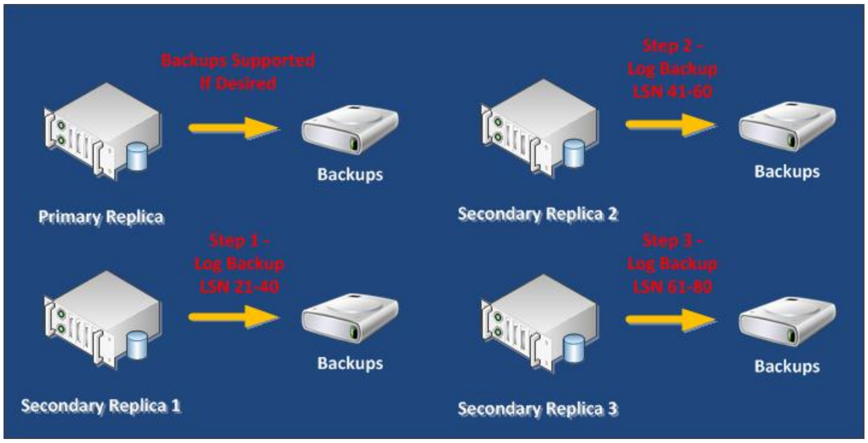

Backups Supported If Desired
Backups
Primary Replica
Step 2 - Log Backup LSN 41-60
Backups
Secondary Replica 2
Step 1 - Log Backup LSN 21-40
Backups
Secondary Replica 1
Step 3 - Log Backup LSN 61-80
Backups
Secondary Replica 3

FIGURE 2-10 Forming a single log chain by backing up the transaction logs on multiple secondary replicas.

As a result, the transaction log backups do not all have to be performed on the same replica. This in no way means that serious thought should not be given to the location of your backups. It is recommended to store all backups in a central location because all transaction log backups are required to perform a restore in the event of a disaster. Therefore, if a server is no longer availability and it contained the backups, you will be negatively affected. In the event of a failure, use the new Database Recovery Advisor Wizard; it provides many benefits when conducting restores. For example, if performing backups on different secondaries, the wizard generates a visual image of a chronological timeline by stitching together all of the log files based on the Log Sequence Number (LSN).

AlwaysOn Failover Cluster Instances (FCI)

You've seen the results of the development efforts in engineering the new AlwaysOn Availability Groups capability for high availability and disaster recovery, and the creation of active secondaries. Now you'll explore the significant enhancements to traditional capabilities like SQL Server failover clustering that leverages shared storage. The following list itemizes some of the improvements that will appeal to database administrators looking to gain high availability for their SQL Server instances. Specifically, this section discusses the following features:

- **Multi-Subnet Clustering** This feature provides a disaster recovery solution in addition to high availability with new support for multi-subnet failover clustering.
- **Support for TempDB on Local Disk** Another storage level enhancement with failover clustering is associated with TempDB. TempDB no longer has to reside on shared storage as it did in previous versions of SQL Server. It is now supported on local disks, which results in many practical benefits for organizations. For example, it is now possible to offload TempDB I/O from share storage devices like a SAN and leverage fast SSD storage locally within the server nodes to optimize TempDB workloads, which are typically random I/O.
- **Flexible Failover Policy** 2012 introduces improved failure detection for the SQL Server Failover Cluster Instance (FCI) by adding failure condition-level properties which allow you to configure a more flexible failover policy.

Note It is also worth mentioning that AlwaysOn Failover Clustering Instances can be combined with Availability Groups to offer maximum SQL Server instance and database protection.

With the release of Windows Server 2008, new functionality enabled cluster nodes to be connected over different subnets without the need for a stretch VLAN across networks. The nodes could reside on different subnets within a datacenter or in another geographical location such as a disaster recovery site. This concept is commonly referred to as *multi-site clustering*, *multi-subnet clustering*, or *stretch-clustering*. Unfortunately, the previous versions of SQL Server could not take advantage of this Windows failover clustering feature. Organizations that wanted to create either a multi-site or multi-subnet SQL Server failover cluster still had to create a stretch VLAN to expose a single IP address for failover across sites. This was a complex and challenging task for many organizations. This is no longer the case because SQL Server 2012 supports multi-subnet and multi-site clustering out-of-the-box; therefore, the need for implementing stretch VLAN technology no longer exists.

Figure 2-11 illustrates an example of a SQL Server multi-subnet failover cluster between two subnets spanning two sites. Notice how each node affiliated with the multi-subnet failover cluster resides on a different subnet. Node 1 is located in Site 1 and resides on the 192.168.115.0/24 subnet whereas; Node 2 is located in Site 2 and resides on the 192.168.116.0/24 subnet.

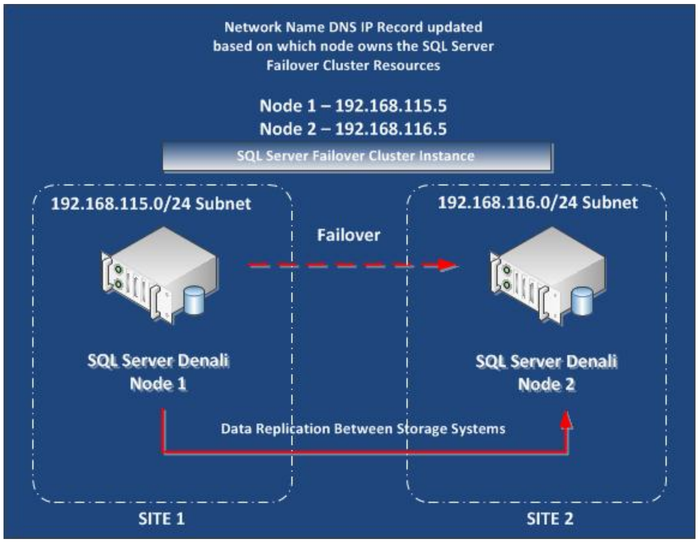
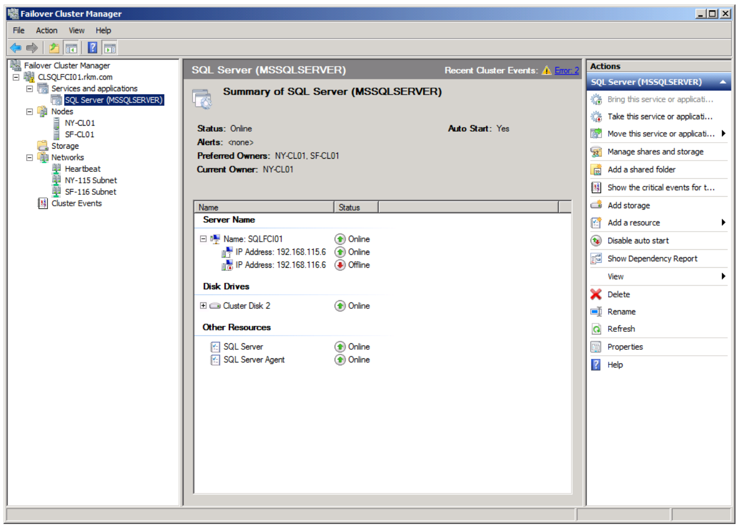

Network Name DNS IP Record updated based on which node owns the SQL Server Failover Cluster Resources
Node 1 – 192.168.115.5
Node 2 – 192.168.116.5
SQL Server Failover Cluster Instance
192.168.115.0/24 Subnet
Failover
192.168.116.0/24 Subnet
SQL Server Denali Node 1
SQL Server Denali Node 2
Data Replication Between Storage Systems
SITE 1
SITE 2

FIGURE 2-11 A multi-subnet failover cluster instance example.

For clients and applications to connect to the SQL Server failover cluster, they need two IP addresses registered to the SQL Server failover cluster resource name in WSFC. For example, imagine your server name is SQLFCI01 and the IP addresses are 192.168.115.5 and 192.168.116.5. WSFC automatically controls the failover and brings the appropriate IP address online depending on the node that currently owns the SQL Server resource. Again, if Node 1 is affiliated with the 192.168.115.0/24 subnet and owns the SQL Server failover cluster then the IP address resource 192.168.115.6 is brought online as shown in Figure 2-12. Similarly, if a failover occurs and Node 2 owns the SQL Server resource, then IP address resource 192.165.115.6 is taken offline and the IP address resource 192.168.116.6 is brought online.



SQL Server (MSSQLSERVER) Recent Cluster Events: Error: 2

Summary of SQL Server (MSSQLSERVER)

Status: Online Auto Start: Yes
Alerts: <none>
Preferred Owners: NY-CL01, SF-CL01
Current Owner: NY-CL01

Name	Status
Server Name	
Name: SQLFCI01	Online
IP Address: 192.168.115.6	Online
IP Address: 192.168.116.6	Offline
Disk Drives	
Cluster Disk 2	Online
Other Resources	
SQL Server	Online
SQL Server Agent	Online

FIGURE 2-12 Multiple IP addresses affiliated with a multi-subnet failover cluster instance screenshot.

Since there are multiple IP addresses affiliated with the SQL Server failover cluster instance virtual name, the online address will change automatically when there is a failover. In addition, Windows failover cluster will issue a DNS update immediately after the network name resource name comes online. The IP address change in DNS might not take effect on clients due to cache settings, therefore, it is recommended to minimize the client downtime by configuring the HostRecordTTL in DNS to 60 seconds.

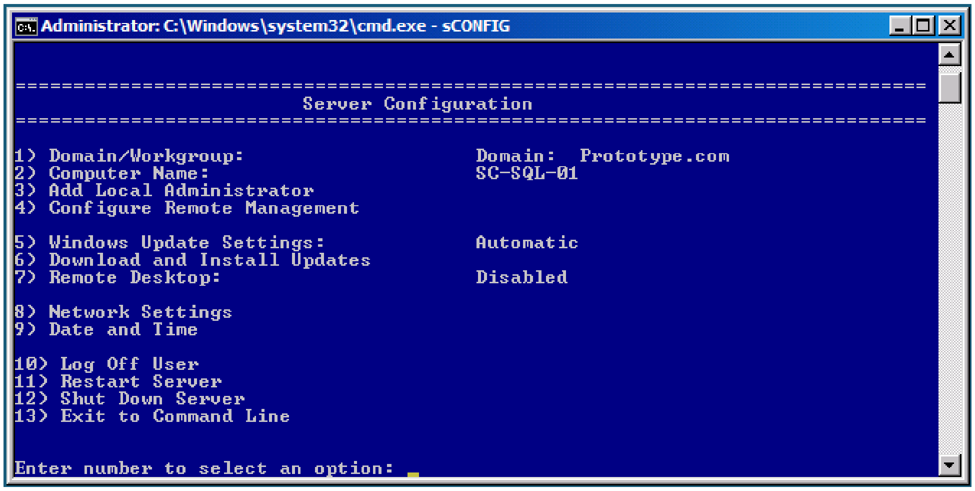
Support for Deploying SQL Server 2012 on Windows Server Core

Windows Server Core was originally introduced with Windows Server 2008 and saw significant enhancements with the release of Windows Server 2008 R2. For those who are unfamiliar with Server Core, it is an installation option for the Windows Server 2008 and Windows Server 2008 R2 operating systems. Since Server Core is a minimal deployment of Windows, it is much more secure because its attack surface is greatly reduced. Server Core does not include a traditional Windows graphical interface and, therefore, is managed via a command prompt or by remote administration tools.

Unfortunately, previous versions of SQL Server did not support the Server Core operating system, but that has all changed. For the first time, Microsoft SQL Server “2012” supports Server Core installations for organizations running Server Core based on Windows Server 2008 R2 with Service Pack 1 or later.

Why is Server Core so important to SQL Server and how does it positively impact availability? When running SQL Server 2012 on Server Core, operating system patching is drastically reduced—by up to 60 percent. This translates to higher availability and a reduction in planned downtime for any organization's mission-critical databases and workloads. In addition, surface area attacks are greatly reduced and overall security of the database platform is strengthened, which again translates to maximum availability and data protection.

When first introduced, Server Core required the use and knowledge of command line syntax to manage it. Most IT professionals at this time were accustomed to using a graphical user interface (GUI) to manage and configure Windows so they had a difficult time embracing Server Core. This impacted its popularity and, ultimately, its implementation. To ease these challenges, Microsoft introduced SCONFIG. SCONFIG is an out-of-the-box utility that was introduced with the release of Windows Server 2008 R2 to dramatically ease server configurations. To navigate through the SCONFIG options, you only need to type one or more numbers to configure server properties as displayed in Figure 2-13.



```
===============================================================================
                          Server Configuration
===============================================================================

1) Domain/Workgroup:                    Domain:  Prototype.com
2) Computer Name:                       SC-SQL-01
3) Add Local Administrator
4) Configure Remote Management

5) Windows Update Settings:             Automatic
6) Download and Install Updates
7) Remote Desktop:                      Disabled

8) Network Settings
9) Date and Time

10) Log Off User
11) Restart Server
12) Shut Down Server
13) Exit to Command Line


Enter number to select an option: _
```

FIGURE 2-13 SCONFIG utility for configuring server properties in Server Core.

The following sections articulate the SQL Server 2012 prerequisites for Server Core, SQL Server features supported on Server Core, and the installation alternatives.

SQL Server 2012 Prerequisites for Server Core

Organizations installing SQL Server 2012 on Windows Server 2008 R2 Server Core must meet the following operating system, features, and components prerequisites:

Operating system:

- Windows Server 2008 R2 SP1 64-bit x64 Data Center Server Core
- Windows Server 2008 R2 SP1 64-bit x64 Enterprise Server Core
- Windows Server 2008 R2 SP1 64-bit x64 Standard Server Core

- Windows Server 2008 R2 SP1 64-bit x64 Web Server Core

Features and components:

- .NET Framework 2.0 SP2
- .NET Framework 3.5 SP1 Full Profile
- .NET Framework 4 Server Core Profile
- Windows Installer 4.5
- Windows PowerShell 2.0

Once the prerequisites are fulfilled, it important to become familiar with the SQL Server components supported on Server Core.

SQL Server Features Supported on Server Core

There are numerous SQL Server features that are fully supported on Server Core. They include Database Engine Services, SQL Server Replication, Full Text Search, Analysis Services, Client Tools Connectivity and Integration Services. Likewise, Sever Core does not support the many other features including Reporting Services, Business Intelligence Development Studio, Client Tools Backward Compatibility, Client Tools SDK, SQL Server Books Online, Distributed Replay Controller, SQL Client Connectivity SDK, Master Data Services and Data Quality Services. Some features like Management Tools – Basic, Management Tools – Complete, Distributed Replay Client and Microsoft Sync Framework are only supported remotely. Therefore, these features can be installed on editions of the Windows operating system that are not Server Core, and then used to remotely connect to a SQL Server instance running on Server Core.

Note To leverage Server Core, it is important to plan your SQL Server installation ahead of time. Give yourself the opportunity to fully understanding which SQL Server features that are required to support your mission critical workloads.

SQL Server on Server Core Installation Alternatives

The typical SQL Server Installation Setup Wizard is not supported when installing SQL Server 2012 on Server Core. As a result, there is a need to automate the installation process by either using a command-line installation, a configuration file, or leveraging the DefaultSetup.ini methodology. Details and examples for each of these methods can be found in Books Online.

Note When installing SQL Server 2012 on Server Core, ensure you use Full Quiet mode by using the /Q parameter or the Quiet Simple mode by using the /QS parameter.

Additional High Availability and Disaster Recovery Enhancements

This section summarizes some of the new features and enhancements you can expect to see.

Support for Server Message Block

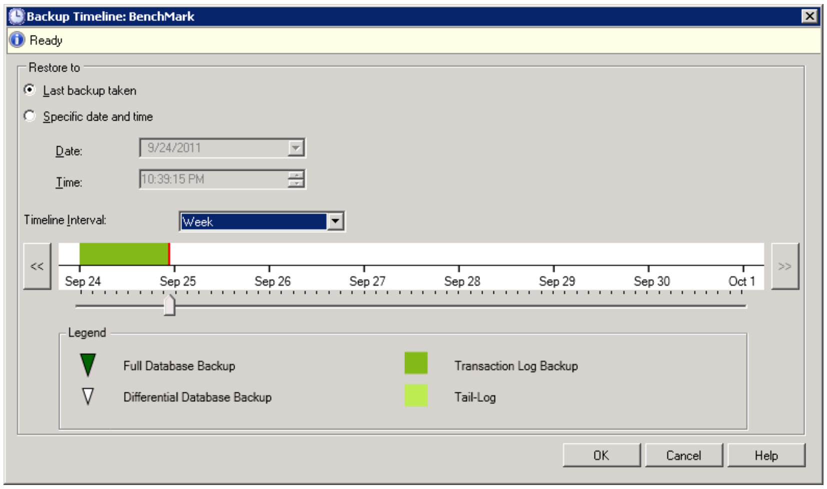
A common movement for organizations in recent years has been towards consolidating databases and applications onto few servers—specifically, hosting many instances of SQL Server running on a failover cluster. When using failover clustering for consolidation, the previous versions of SQL Server required a single drive letter for each SQL Server failover cluster instance. Because there are only 23 drive letters available, without taking into account reservations, the maximum amount of SQL Server instances supported on a single failover cluster was 23. Twenty-three instances sounds like an ample amount; however, the drive letter limitation negatively impacts organizations running powerful servers that have the compute and memory resources to host more than 23 instances on a single server. Going forward, SQL Server 2012 and failover clustering introduces support for Server Message Block (SMB).

Note Some of you must be thinking you can use mount points to alleviate the drive letter pain point. When working with previous versions of SQL Server, even with mount points, you require at least one drive letter for each SQL Server Failover Cluster Instance.

Some of the SQL Server 2012 benefits brought about by SMB are, of course, database storage consolidation and the potential to support more than 23 clustering instances in a single WSFC. To take advantage of these features, the file servers must be running Windows Server 2008 or later versions of the operating system.

Database Recovery Advisory

The Database Recovery Advisor is a new feature aimed at optimizing the restore experience for database administrators conducting database recovery tasks. This tool includes a new timeline feature that provides a visualization of the backup history, as shown in Figure 2-14.


Backup Timeline: BenchMark

Ready

Restore to

(●) Last backup taken

( ) Specific date and time

Date: 9/24/2011

Time: 10:39:15 PM

Timeline Interval: Week

Legend

▼ Full Database Backup

▽ Differential Database Backup

🟩 Transaction Log Backup

🟩 Tail-Log

FIGURE 2-14 Database Recovery Advisory backup and restore visual timeline.

Online Operations

SQL Server 2012 also includes a few enhancements for online operation that reduce downtime during planned maintenance operations. Line of business (LOB) re-indexing and adding columns with defaults are now supported.

Rolling Upgrade and Patch Management

All of the new AlwaysOn capabilities reduce application downtime to only a single manual failover by supporting rolling upgrades and patching of SQL Server. This means a database administrator can apply a service pack or critical fix to the passive nodes if using a failover cluster or secondary replicas if using availability groups. Once the installation is complete on all passive nodes or secondaries, a database administrator can conduct a manual failover and then apply the service pack or critical fix to the node in a FCI or replica. This rolling strategy also applies when upgrading the database platform.

Chapter 3
Scalability and Performance

SQL Server 2012 introduces a new index type called Columnstore. The columnstore index feature was originally referred to as project "Apollo" during the development phases of SQL Server 2012 and during the distribution of the Community Technology Preview (CTP) releases of the product. Together, this new index combined with the new advanced query processing enhancements offer blazing fast performance optimizations for data warehousing workloads and other queries which are similar in nature. In many cases, data warehouse query performance has improved by tens to hundreds of times.

This chapter focuses on answering specific questions that aim to teach, enlighten, and even dispel flawed beliefs so that Database Administrators can obtain blazing fast query performance for their data warehouse workloads. The questions this chapter focuses on appear below:

- What is a columnstore index?
- How does a columnstore index drastically increase the speed of data warehouse queries?
- When should a Database Administrator build a columnstore index?
- Are there any well-established best practices for a columnstore index deployment?

Let's dive under the covers to see how organizations will benefit from astonishing data warehouse performance gains with the new in-memory columnstore index technology, which also helps with managing increasing data volumes.

Columnstore Index Overview

Due to an explosion of data being captured across devices, applications and services, organizations today are tasked with storing massive amounts of data to successfully operate their businesses. Using traditional tools to capture, manage, and process data within an acceptable time is becoming increasingly challenging as data continues to grow. For example, the volume of data is overwhelming the ability of data warehouses to execute queries in timely manner and considerable time is spent tuning queries and designing and maintaining indexes to try to get acceptable query performance. In addition, in many cases so much time may have elapsed that organizations have difficulty recalling what the original request was about or equally unproductive, the business opportunity is lost.

With the many issues organizations were facing, the Query Processing and Storage teams from the SQL Server Product Group set to work on new technologies that would allow very large data sets to be read quickly and accurately while simultaneously transforming the data into useful information and knowledge for organizations in a timely manner. The Query Processing team reviewed academic research in column store data representations and analyzed new improved query execution capabilities for data warehousing. In addition, they collaborated the SQL Server Product Group Analysis Services Team to gain a stronger understanding about the other team's work with their columnstore implementation known as PowerPivot for SQL Server 2008 R2. Their research and analysis led the Query Processing team in creating the new columnstore index and query optimizations based on vector-based execution capability which significantly improves data warehouse query performance by tens to hundreds of times in many cases.

When developing the new columnstore index, the Query Processing team committed to a number of goals. They aimed to ensure that an interactive experience existed with all data sets, whether large or small, which meant the response time on data must be swift. These strategies also apply to AD hoc and reporting queries. Moreover, Database Administrators may even be able to reduce their needs for manually tuning queries, summary tables, indexed views, and in some cases OLAP cubes. All these goals naturally impact Total Cost of Ownership because hardware costs are lowered and fewer people are required to get a task accomplished.

Columnstore Index Fundamentals and Architecture

Before designing, implementing or managing a Columnstore index it is beneficial to understand how columnstore indexes work, how data is stored and what type of queries benefit from a columnstore index.

How is Data Stored When using a Columnstore Index?

With traditional tables (heaps) and indexes (B-trees), SQL Server stores data in pages in a row-based fashion. This storage model is typically referred to as a row store. With column stores, it is like turning the traditional storage model 90 degrees where all the values from a single column are stored contiguously in a compressed form. The columnstore index stores each column in a separate set of disk pages rather than storing multiple rows per page, which has been the traditional storage format. The following examples will illustrate the differences.

Let's review a common table populated with employee data as illustrated in Table 3-1 and then evaluate the different ways data can be stored. This Employee Table includes typical data such as the EmployeeID, Name, City and State. Depending on the type of index chosen, traditional or columnstore, Database Administrators can organize their data by row as shown in Table 3-2 or by column as shown in Table 3-3.

TABLE 3-1 A traditional table containing employee data.

EmployeeID	Name	City	State
1	Ross	San Francisco	CA
2	Sherry	New York	NY
3	Gus	Seattle	WA
4	Stan	San Jose	CA
5	Lijon	Sacramento	CA

As you can see, the major difference between the columnstore format, Table 3-3 and the row store method, Table 3-2 is that a columnstore index groups and stores data for each column and then joins all the columns to complete the whole index, whereas a traditional index groups and stores data for each row and then joins all the rows to complete the whole index.

TABLE 3-2 Employee data stored in a traditional "Row Store" format.

Row Store
1 Ross San Francisco CA
2 Sherry New York NY
3 Gus Seattle WA
4 Stan San Jose CA
5 Lijon Sacramento CA

TABLE 3-3 Employee data stored in the new “Columnstore” format.

Columnstore
1 2 3 4 5
Ross Sherry Gus Stan Lijon
San Francisco New York Seattle San Jose Sacramento
CA NY WA CA CA

Now that we understand how data is stored when using Columnstore indexes compared to traditional b-tree indexes, let’s understand how this new storage model and advanced query optimizations significantly speed up retrieval of data. The next section will include three ways that SQL Server columnstore indexes significantly improve the speed of queries.

How do Columnstore Indexes Significantly Improve the Speed of Queries?

The new columnstore storage model significantly improves data warehouse query speeds for many reasons. First, data organized in a column share many more similar characteristics than data organized across rows. As a result, a much higher level of compression can be achieved compared to data organized across rows. Moreover, the columnstore index within SQL Server uses the VertiPaq compression algorithm technology, which in SQL Server 2008 R2 was found only in Analysis Server for PowerPivot. VertiPaq compression is far superior to traditional row and page compression used in the Database Engine and compression rates of up to 15-to-1 have been achieved. When data is compressed, queries require less IO because the amount of data transferred from disk to memory is significantly reduced. Reducing IO when processing queries equates to faster performance response times. The advantages of columnstore do not come to an end here. With less data transferred to memory, less space is required in memory to hold the working set affiliated with the query.

Second, when a user runs a query using the columnstore index, SQL Server only fetches data for the columns that are required for the query, as illustrated in Figure 3-1. In this example, there are 15 columns in the table, however, since the data required for the query resides in Column 7, Column 8 and Column 9, only these columns are being retrieved.

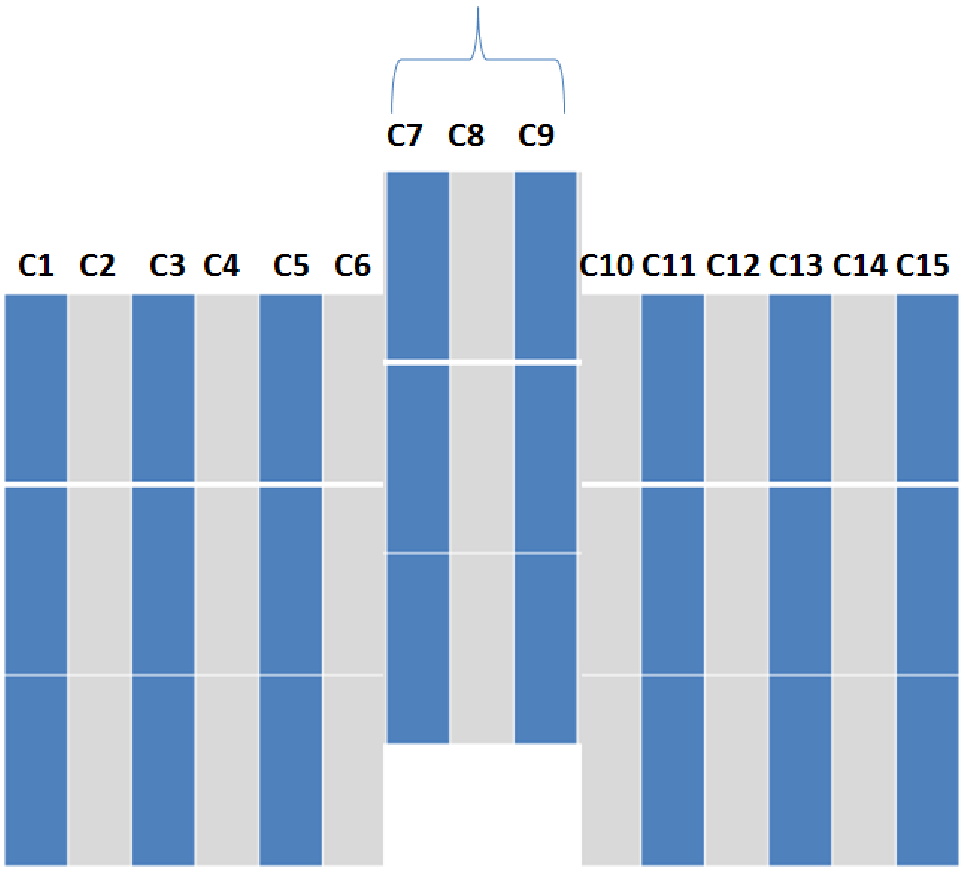


FIGURE 3-1 Improving Performance and Reducing IO by only fetching columns required for the query.

Because data warehouse queries typically touch only ten to fifteen percent of the columns in large Fact tables, fetching only selected columns translates into additional savings of approximately eighty-five to ninety percent of an organization's IO, which again increases performance speeds.

Batch Mode Processing

Finally, an advanced technology for processing queries that use columnstore indexes speeds up queries yet another way. Speaking about processes, it is a good time to look a little deeper into how queries are processed. First, the data in the columns are processed in batches using a new highly efficient vector technology that works with columnstore indexes. Database Administrators should take a moment to review the query plan and notice the groups of operators that execute in batch mode. Note that not all operators execute in batch mode; however, the most important ones affiliated with data warehousing do, such as Hash Join and Hash Aggregation. All of the algorithms have been significantly optimized to take advantage of modern hardware architecture such as increased core counts and additional RAM, accordingly improving parallelism. All of these improvements affiliated with the columnstore index contribute to better batch mode processing compared to traditional row mode processing.

Columnstore Index Storage Organization

Let's examine how the storage associated with a columnstore index is organized. First let's define new storage concepts affiliated with columnstore indexes such as a Segment and a Row Group, and then I will elucidate how all of this correlates to one another. As illustrated in Figure 3-2, data in the columnstore index is broken up into segments. A segment contains data from one column for a set of about 1 million rows. Segments for the same set of rows comprise a row group. Instead of storing the data page-by-page, SQL Server stores the row group as a unit. Each segment is internally stored in a separate Large Object (LOB). Therefore, when SQL Server reads the data, the unit reading from disk consists of a segment and the segment is a unit of transfer between the disk and memory.

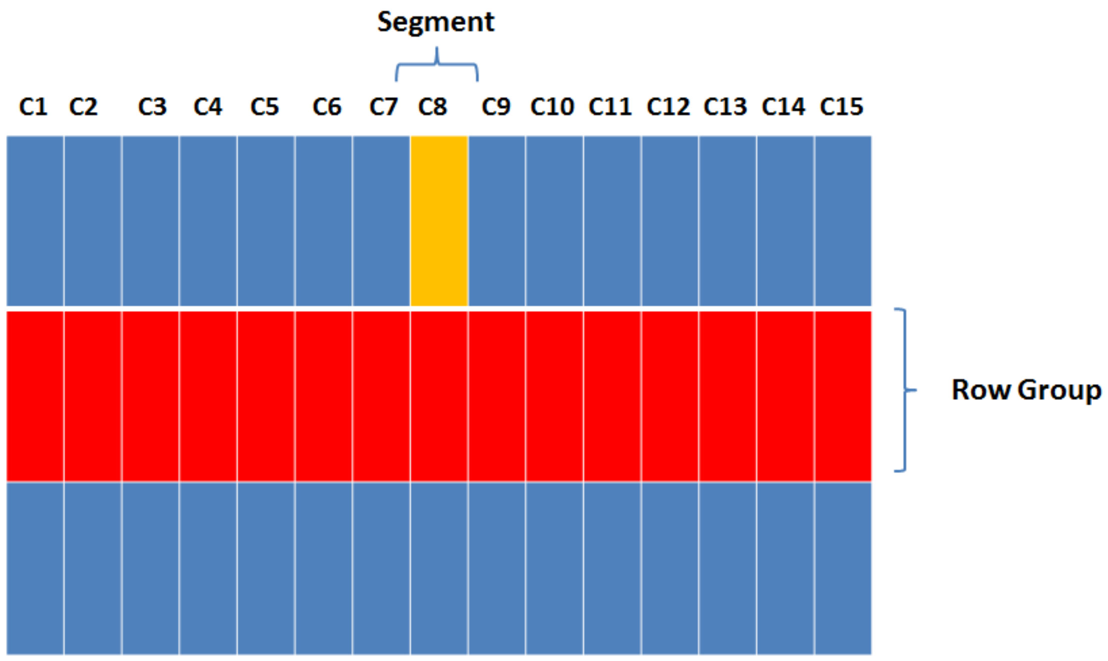


FIGURE 3-2 How a Columnstore Index Stores Data.

Columnstore Index Support and SQL Server 2012

Columnstore indexes and batch query execution mode are deeply integrated into SQL Server 2012 and work in conjunction with many of the Database Engine features found in SQL Server 2012. For example, Database Administrators can implement a columnstore index on a table and still successfully use

AlwaysOn Availability Groups (AG), AlwaysOn Failover Cluster Instances (FCI), Database Mirroring, Log Shipping, and SQL Server Management Studio Administration Tools. With respect to data types, here are the common business data types supported:

- char and varchar
- All Integer types (int, bigint, smallint, and tinyint)
- real and float
- string
- money and small money
- All date, time and DateTime types with one exception (datetimeoffset with precision > 2)
- Decimal and numeric with precision <= 18 (i.e. <= 18 Digits)

Columnstore Index Restrictions

Although columnstore indexes work with the majority of the data types, components and features found in SQL Server 2012, there are some restrictions and situations where columnstore indexes cannot be leveraged. These restrictions and situations are outlined below:

- You can enable PAGE or ROW compression on the base table but you cannot enable PAGE or ROW compression on the columnstore index.
- Tables and Columns cannot participate in a Replication topology.
- Tables and Columns using Change Data Capture are unable to participate in a columnstore index.
- Create Index - You cannot create a columnstore index on the following data types:
    - decimal > 18 Digits
    - binary
    - BLOB
    - CLR
    - (n)varchar(max)
    - uniqueidentifier
    - datetimeoffset with precision > 2
- Table Maintenance – If a columnstore index exists you can read the table but, you cannot directly update it. This is because Columnstore indexes are designed for data warehouse workloads that are typically read based. Rest assured that there is no need to agonize. The upcoming

section – Loading Data - articulates strategies on how to load new data when using columnstore indexes.

- Process Queries - You can process all read only T-SQL queries using the columnstore index, but because batch processing only works with certain operators, you will witness that some queries are accelerated more than others.
- A column that contains filestream data cannot participate in a columnstore index.
- INSERT, UPDATE, DELETE, and MERGE statements are not allowed on tables using Columnstore indexes.
- More than 1024 columns are not supported when creating a Columnstore index.
- Only non-clustered columnstore indexes are allowed Filtered columnstore indexes are not allowed.
- Computed and sparse columns cannot be part of a columnstore index.
- A columnstore index cannot be created on an indexed view.

Columnstore Index Design Considerations and Loading Data

When working with Columnstore indexes, some queries are accelerated much more than others. Therefore it is important to understand when to build a columnstore index and when not to build a columnstore index in order to optimize query performance to its maximum potential. The next sections address the learning's required and will cover columnstore index design considerations and how to load data when using a columnstore index.

When to build a columnstore index

The following bullets exemplify when Database Administrators should use a columnstore index to optimize query performance.

- When workloads are mostly Read based – specifically data warehouse.
- Your workflow permits partitioning (or drop rebuild index strategy) to handle new data. Most commonly this will be associated with periodic maintenance windows when indexes can be rebuilt or when staging tables are switching into empty partitions of existing tables.
- If most queries fit a star join pattern or entail scanning and aggregating large amounts of data
- If update occurs, most of these updates append new data, which can be loaded using staging tables and partition switching.

What tables should have columnstore indexes

- Your large fact tables
- Consider building a columnstore index if you have a very large (millions of rows) dimension table

When not to build a columnstore index

There may be situations where Database Administrators may encounter situations from time to time when the performance benefits achieved by using traditional B-tree indexes on their tables are greater than the benefits of using a columnstore index. The bullets below point out some of these situations:

- Data in your table constantly require updating.
- Partition switching or rebuilding an index does not meet the workflow requirements of your business.
- You encounter frequent small look up queries. Note, however, that a columnstore index may still benefit you in this situation As such; you can implement a columnstore index without any repercussions because the query optimizer should be able to determine when to use the traditional B-Tree index vs. the Columnstore index. This strategy assumes you have updated statistics.
- You test columnstore indexes on your workload and do not see a benefit.

Loading new Data

As mentioned in earlier sections, tables with a columnstore index cannot be updated directly. However, there are three alternatives for loading data into a table with a columnstore index. Let’s take a look at these strategies:

Disable the columnstore index

The procedure consists of:

- Start by first disabling the columnstore index.
- Update the data.
- Rebuild the index when the updates are complete.
- Database Administrators should ensure a maintenance window exists when leveraging this strategy.

Leverage Partitioning and Partition Switching

Partitioning data enables Database Administrators to manage and access subsets of their data quickly and efficiently while maintaining the integrity of the entire data collection. Partition switching also allows Database Administrators to quickly and efficiently transfer subsets of their data by assigning a table as a partition to an already existing partitioned table, switching a partition from one partitioned table to another, or reassigning a partition to form a single table. Partition switching is fully supported with a columnstore index and is a practical way for updating data.

To use partitioning to load data:

- Ensure you have an empty partition to accept the new data
- Load data into an empty staging table
- Switch the staging table (containing the newly loaded data) into the empty partition

To use partitioning to update existing data:

- Determine which partition contains the data to be modified
- Switch the partition into an empty staging table
- Disable the columnstore index on the staging table
- Update the data
- Rebuild the columnstore index on the staging table
- Switch the staging table back into the original partition (which was left empty when the partition was switched into the staging table)

Union All

Database Administrators can load data by storing their main data in a fact table that has a columnstore. Next, create a secondary table to add or update data. Finally, leverage a UNION ALL query so it returns all of the data between the large fact table with columnstore and smaller updateable tables. Periodically load the table from the secondary table into the main table by using partition switching or by disabling and rebuilding the columnstore index. Note that some queries using the UNION ALL strategy may not be as fast as if all the data were in a single table.

Creating a Columnstore Index

Creating a columnstore index is very similar to creating any other traditional SQL Server indexes. Create it using the graphical user interface in SQL Server Management Studio or by using Transact-SQL. Many individuals prefer to use the graphical user interface as they want to avoid typing all of the column names when creating the index with Transact-SQL. In addition, a few frequent questions arise when creating a columnstore index. Database Administrators want to know (1) which columns should be included in the columnstore index and (2) is it possible to create a clustered columnstore index. When creating a columnstore index, a Database Administrator should typically include all of the columns associated with the table. It is worth noting that it is not necessary to include all of the columns, and the limit on the number of columns is 1024, which is similar to any other index. For the second question, no, all columnstore indexes must be non-clustered; therefore, a cluster columnstore index is not allowed. The next sections explain both strategies for creating a columnstore index – SQL Server Management Studio and with Transact-SQL.

Creating a columnstore index by using the SQL Server Management Studio

In SQL Server Management Studio, use Object Explorer to connect to an instance of the SQL Server Database Engine.

In Object Explorer, expand the instance of SQL Server, expand Databases, expand a database and expand a table in which you would like to create a new columnstore index.

Expand the Table, Right-click the Index folder, choose New Index, and then click Non-Clustered Columnstore Index.

On the General tab, in the Index name box, type a name for the new index, and then click Add.

In the Select Columns dialog box, select the columns to participate in the columnstore index and then click OK.

Alternatively, configure the settings on the Options, Storage and Extended Properties pages. If you will maintain the defaults, click OK to create the index, as illustrated in Figure 3-3.

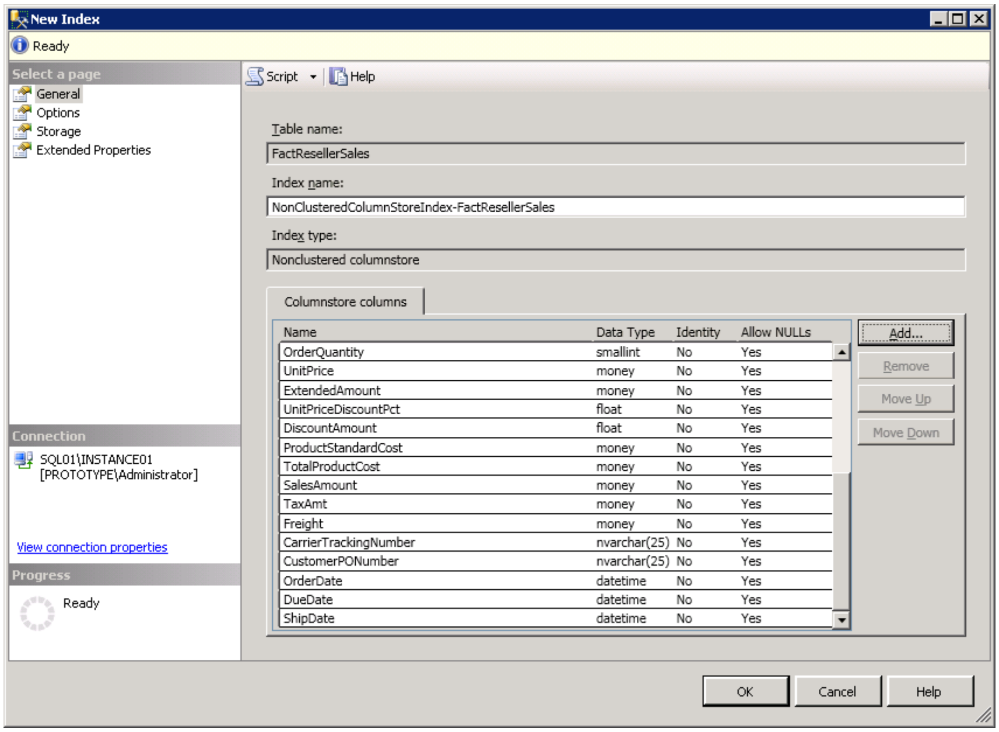


Table name: FactResellerSales

Index name: NonClusteredColumnStoreIndex-FactResellerSales

Index type: Nonclustered columnstore

Columnstore columns

Name	Data Type	Identity	Allow NULLs
OrderQuantity	smallint	No	Yes
UnitPrice	money	No	Yes
ExtendedAmount	money	No	Yes
UnitPriceDiscountPct	float	No	Yes
DiscountAmount	float	No	Yes
ProductStandardCost	money	No	Yes
TotalProductCost	money	No	Yes
SalesAmount	money	No	Yes
TaxAmt	money	No	Yes
Freight	money	No	Yes
CarrierTrackingNumber	nvarchar(25)	No	Yes
CustomerPONumber	nvarchar(25)	No	Yes
OrderDate	datetime	No	Yes
DueDate	datetime	No	Yes
ShipDate	datetime	No	Yes

SQL01\INSTANCE01
[PROTOTYPE\Administrator]

FIGURE 3-3 Creating a new Non-Clustered Columnstore Index with SSMS

Creating a columnstore index by using Transact-SQL

As mentioned earlier, you can use Transact-SQL to create a columnstore index instead of the graphical user-interface in SQL Server Management Studio. The example below illustrates the syntax for creating a columnstore index with Transact-SQL.

```
CREATE [ NONCLUSTERED ] COLUMNSTORE INDEX index_name
    ON <object> ( column  [ ,...n ] )
    [ WITH ( <column_index_option> [ ,...n ] ) ]
    [ ON {
           { partition_scheme_name ( column_name ) }
           | filegroup_name
           | "default"
         }
    ]
[ ; ]
<object> ::=
{
    [database_name. [schema_name ] . | schema_name . ]
     table_name
{

<column_index_option> ::=
{
      DROP_EXISTING = { ON | OFF }
    | MAXDOP = max_degree_of_parallelism
 }
```

The following bullets explain the arguments affiliated with the Transact-SQL syntax to create the non-clustered columnstore index.

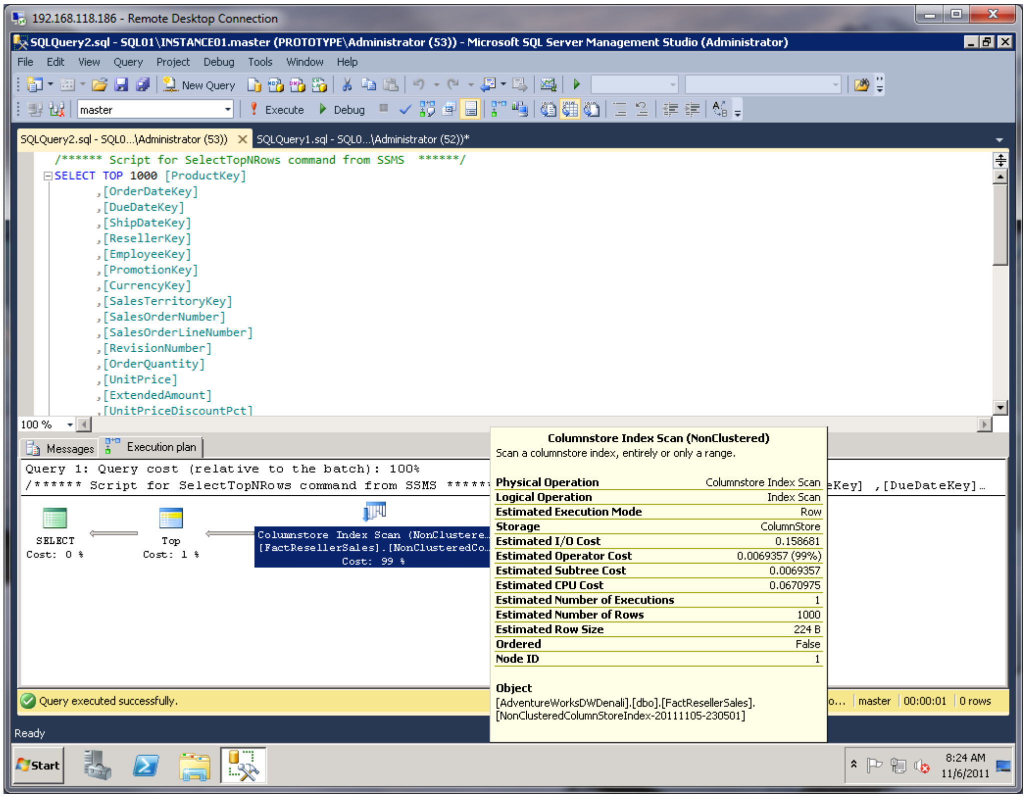
- **NONCLUSTERED** The NONCLUSTERED argument indicates that this index is a secondary representation of the data.
- **COLUMNSTORE** This argument indicates that the index which will be created is a columnstore index.
- **index_name** This is where you specify the name of the columnstore index to be created. Index names must be unique within a table or view but do not have to be unique within a database.
- **column** This refers to the column or columns to be added to the index. As a reminder, a columnstore index is limited to 1024 columns.
- **ON partition_scheme_name(column_name)** Specifies the partition scheme that defines the filegroups on which the partitions of a partitioned index are mapped. The column_name specifies the column against which a partitioned index will be partitioned. This column must match the data type, length, and precision of the argument of the partition function that the partition_scheme_name is using. partition_scheme_name or filegroup. If these are not specified and the table is partitioned, the index is placed in the same partition scheme using the same partitioning column as the underlying table.
- **ON filegroup** The filegroup name represents the name of the filegroup to create the specified index.
- **ON "default"** This argument is utilized when you want to create the specified index on the default filegroup.

Using Columnstore Indexes

Now that you have created a new columnstore index to speed up your queries, it is important to understand how you determine if the columnstore index is actually being used to accelerate your query. The first step is to examine the execution plan associated with query being fired. In the results window of the graphical execution show plan, there is a new Columnstore Index Scan Operator icon, as illustrate in Figure 3-4.The Columnstore Index Scan Operator icon indicates that a columnstore index is being used for index scans.

FIGURE 3-4 New Columnstore Index Scan Operator Icon

You can obtain additional columnstore indicators and performance cost details if you highlight the new Columnstore Index Scan Operator icon. For example, in Figure 3.5, the Physical Operation element indicates Columnstore Index Scan was used and the Storage element states Columnstore.


192.168.118.186 - Remote Desktop Connection
SQLQuery2.sql - SQL01\INSTANCE01.master (PROTOTYPE\Administrator (53)) - Microsoft SQL Server Management Studio (Administrator)
File Edit View Query Project Debug Tools Window Help
New Query
master
Execute
Debug
SQLQuery2.sql - SQL0...\Administrator (53))
SQLQuery1.sql - SQL0...\Administrator (52))*
100 %
Messages
Execution plan
Query executed successfully.
master
00:00:01
0 rows
Ready
Start
8:24 AM
11/6/2011

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [ProductKey]
      ,[OrderDateKey]
      ,[DueDateKey]
      ,[ShipDateKey]
      ,[ResellerKey]
      ,[EmployeeKey]
      ,[PromotionKey]
      ,[CurrencyKey]
      ,[SalesTerritoryKey]
      ,[SalesOrderNumber]
      ,[SalesOrderLineNumber]
      ,[RevisionNumber]
      ,[OrderQuantity]
      ,[UnitPrice]
      ,[ExtendedAmount]
      ,[UnitPriceDiscountPct]
```

```
Query 1: Query cost (relative to the batch): 100%
/****** Script for SelectTopNRows command from SSMS ******…eKey] ,[DueDateKey]…

SELECT        Top           Columnstore Index Scan (NonClustere…
Cost: 0 %     Cost: 1 %     [FactResellerSales].[NonClusteredCo…
                            Cost: 99 %
```

Columnstore Index Scan (NonClustered)
Scan a columnstore index, entirely or only a range.

Physical Operation	Columnstore Index Scan
Logical Operation	Index Scan
Estimated Execution Mode	Row
Storage	ColumnStore
Estimated I/O Cost	0.158681
Estimated Operator Cost	0.0069357 (99%)
Estimated Subtree Cost	0.0069357
Estimated CPU Cost	0.0670975
Estimated Number of Executions	1
Estimated Number of Rows	1000
Estimated Row Size	224 B
Ordered	False
Node ID	1

Object
[AdventureWorksDWDenali].[dbo].[FactResellerSales].
[NonClusteredColumnStoreIndex-20111105-230501]

FIGURE 3-5 Reviewing the Columnstore Index Scan results.

Using Hints with a Columnstore Index

Finally, if you believe the query can benefit from a columnstore index and the query execution plan is not leveraging it, then it is possible to force the query to use a columnstore index. This is achieved by using the WITH (INDEX(<indexname>)) hint where the <indexname> argument would be the name of the columnstore index you wish to force.

The following example illustrates a query with an index hint forcing the use of a columnstore index.

```
SELECT DISTINCT (SalesTerritoryKey)
FROM dbo.FactResellerSales with (index (Non-ClusteredColumnStoreIndexSalesTerritory)
GO
```

The next example illustrates a query with an index hint forcing the use of a different index such as a traditional clustered b-tree index over a columnstore index. For example, let's say there are two indexes on this table called SalesTerritoryKey, a clustered index called ClusteredIndexSalesTerritory and a Non-Clustered Columnstore index called Non-ClusteredColumnStoreIndexSalesTerritory. Instead of using the columnstore index, the hint will force the query to use the clustered index known as ClusteredIndexSalesTerritory.

```
SELECT DISTINCT (SalesTerritoryKey)
FROM dbo.FactResellerSales with (index (ClusteredIndexSalesTerritory)
GO
```

The final example illustrates a query with an index hint option forcing the query to ignore the columnstore index.

```
SELECT DISTINCT (SalesTerritoryKey)
FROM dbo.FactResellerSales
Option (ignore_nonclustered_columnstore_index)
GO
```

Columnstore Index Observations and Best Practices

It is without a doubt that the SQL Server Product Group has made major investments in the creation of columnstore indexes to optimize query processing times affiliated with data warehouse workloads. The Query Optimization, Query Execution, and Storage Engine teams, in collaboration with the SQL Server Performance Team, SQL Server Customer Advisory Team (SQLCAT) and Microsoft Technology Centers (MTCs) have been testing columnstore indexes with numerous customers since the new technology came into existence. Based on results from these tests, customers have indicated their queries are currently "Ridiculously Fast" and the results are "Mind Boggling". Kudos goes out to everyone who was an active participant in the creation of the columnstore index. Now, turn your attention the columnstore index best practices that are based on initial testing results:

- Try to write your queries to match the "sweet spot" for columnstore indexes. Star join queries, for example, can really fly.

- Whenever possible, avoid constructs that may reduce the benefit of columnstore indexes, such as Outer Joins, Unions, and Union All.
- Include all columns in the columnstore index whenever possible.
- If possible convert decimal/numeric to precision <= 18
- Creating indexes requires a considerable amount of memory, therefore, size the memory on the SQL Server system accordingly.
- Ensure your query leverages batch mode processing whenever possible. It is very important and provides considerable benefits to speed up query.
- Use integer types whenever possible since they provide more compact representation and more opportunity for early filtering.
- Consider table partitioning to facilitate updates.
- Even when your query cannot use batch mode, you can still experience performance benefits using columnstore indexes by reducing I/O.

Summary

By delivering a columnstore index directly in the relational engine, SQL Server 2012 offers organizations the ability to gain breakthrough & predictable performance for database warehouse type queries at astonishing speeds. A columnstore index can be created with the graphical user interface in SQL Server Management Studio or with Transact-SQL. As a reminder, depending on the type of query including the processing characteristics, some queries will be optimized more than others.

Chapter 4
Security Enhancements

The general consensus on the previous version of Microsoft SQL Server was that it delivered exceptional data protection, access control, and compliance. SQL Server 2008 R2 came equipped with numerous capabilities for organizations to leverage including, but not limited to Transparent Data Encryption to protect data at rest, Extensive Key Management to fulfill data and key separation, Kerberos Authentication to achieve the strongest authentication and, SQL Server Audit, Policy Based Management and Change Data Capture for fulfilling compliance requirements. While SQL Server 2008 R2 was unquestionably robust from a security perspective and a fervent leader in the database platform industry with the least amount of vulnerabilities and least amount of security patches required to maintain the system, SQL Server 2012 increases SQL Server's popularity as it delivers several security enhancements to help organizations improve their control of data access while maintaining the highest level of data protection and compliance.

The following list is a few of the improvements made to SQL Server 2012 that will appeal to organizations looking to gain maximum security and control of their database platform:

- Security Manageability Improvements
- Default Schema for Groups
- User Defined Server Roles
- Audit Enhancements
- Audit Supported on all SKUs
- Improved Resilience
- User-Defined Audit Event
- Record Filtering
- T-SQL Stack Information
- Database Authentication Enhancements
- Contained Databases Authentication
- Crypto Changes
- Hashing Algorithms
- Certificate Key Length

- Service Master Key and Database Master Key Encryption changes from 3DES to AES
- Miscellaneous Security Enhancements
- SharePoint Active Directory
- Provisioning Enhancements
- New Permissions

This chapter communicates each of these new security enhancements introduced in SQL Server 2012, starting with Security Manageability Improvements.

Security Manageability Improvements

Two small but very significant changes are introduced to improve security manageability in SQL Server 2012. The first improvement is Default Schema for Groups and the second improvement is User Defined Server Roles. It is worth mentioning that the Default Schema for Groups is the number one security feature request from the SQL Server community on the Microsoft Connect site.

Default Schema for Groups

In the previous versions of SQL Server, it was possible to define a default schema for SQL Server Users; this action improved security and simplified administration. The default schema was the first schema searched when resolving the names of objects it referenced. When a default schema for an account did not exist, SQL Server assumed dbo was the default schema. One not so minor setback was the inability to define default schemas for Windows Groups. This as anyone can guess, caused administrative challenges; if a user was authenticated by SQL Server as a member of a Windows Group, a default schema was not associated with the user. If a user created an object such as a Table, a new schema was generated and the default name of the schema was the same as the user. As you can imagine, this was a managing nightmare for database administrators. Just think, if there is 500 users associated with a Windows Group and all of the users within the Windows Group created objects, and then a database administrator would need to manage 500 different schemas.

Fortunately with SQL Server 2012, the security management associated with schemas for groups is not only simplified, but now default schemas can be created for Windows Groups. By assigning default schemas to Windows Groups, organizations can experience a simplification in database schema administration and database schema management through individual Windows users. Equally important, the possibility of delegating schema to the wrong users is thwarted when users change groups. On the flip side, if an incorrect schema is used, query errors are prevented with Default Schema for Groups. Lastly, unnecessary implicit schema creation is prevented.

A frequent question is raised when discussing schemas. What happens if users are affiliated with more than one Windows Group? If no default schema is defined for a user account, reviews the sys.principal table and chooses the group with the lowest principal ID as the default schema.

> **Note** With SQL Server 2012, the default schema for a group can be defined by using the DEFAULT_SCHEMA option of CREATE USER or ALTER USER. If no default schema is defined for a group, SQL Server will assume dbo is the default schema.

The following Transact-SQL script demonstrates the creation of a Default Schema for a new Windows Group local group.

```
-- Create User based on Local Group [SQL01\CDBUsers]
CREATE USER [SQL01\CDBUsers]
GO
--Allocate Database Role Membership
ALTER ROLE DB_DATAREADER ADD MEMBER [SQL01\CDBUsers]
ALTER ROLE DB_DATAWRITER ADD MEMBER [SQL01\CDBUsers];
GO
--Create Default Schema for Group
CREATE SCHEMA Users AUTHORIZATION [SQL01\CDBUsers];
GO
-- Set the default schema for Group
ALTER USER [SQL01\CDBUsers] WITH DEFAULT_SCHEMA = Users
GO
--Create Table with Group and Default Schema
CREATE TABLE Users.t1(c1 int)
GO
--Insert Value
INSERT INTO Users.t1 VALUES (1)
```

The example creates the local group called [SQL01\CDBUsers], allocates the appropriate database role membership, creates and then allocates the default schema for the group. The final piece creates a table with the default schema for the group and then inserts 1 record.

User Defined Server Roles

Role based security and separation of duties are strategies that organizations must adhere to in order to achieve compliance with their systems. The main goal of these strategies is to limit system access to authorized users to reduce security threats, compromises, and operational mistakes while improving manageability of users and their privileges. These strategies are typically achieved by creating a role, applying permissions to the role, and then assigning members to the roles instead of just applying the same level of permissions to every user or administrator who requires access or administrative rights.

In previous versions of SQL Server, the User Defined Role ensured role based security in order to achieve separation of duties. However, the User Defined Role provided separation of duties as the database level and not at the server level. This is because at the server level, administrators had access to Fixed Roles. As the name implies, unlike the User Defined Role, Server Roles were fixed and Database administrators could not customize the securable from a granular perspective. This typically led to database administrators providing members with elevated access, such as the sysadmin role because you couldn't find a fixed role which closely met the business and security requirements. With SQL Server 2012, User Defined Roles have been introduced at the server level to increase flexibility, manageability, and facilitates compliance towards better separation of duties when administering the server.

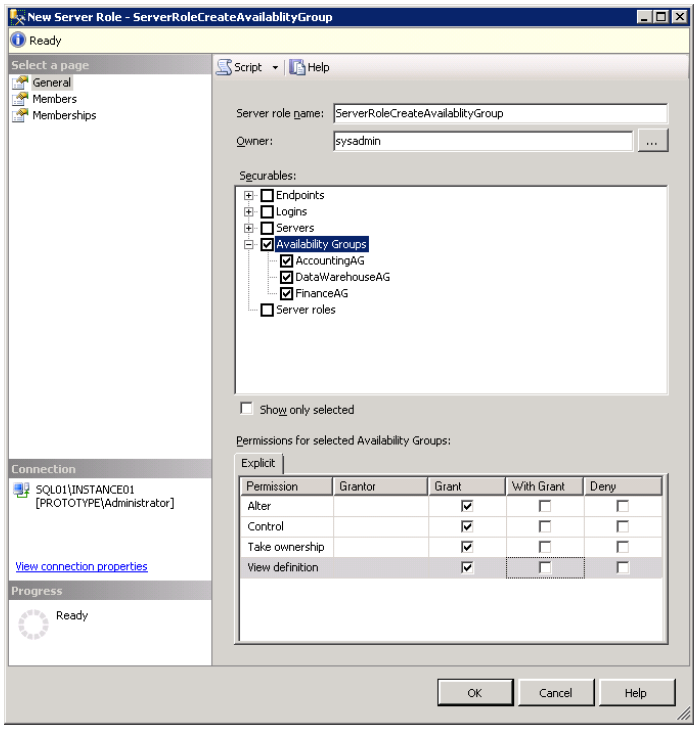
Creating and Managing Server Roles

When it's time to create Server Roles use SQL Server Management Studio (SSMS) by expanding the Security folder in Object Explorer and then right-clicking Server Roles and choosing New Server role. Alternatively, create and manage Server roles with the following Transact-SQL statements: CREATE SERVER ROLE, ALTER SERVER ROLE, and DROP SERVER ROLE.

Creating Server Roles with SQL Server Management Studio

The following example demonstrates the ability to create a server role with SSMS.

1. In SQL Server Management Studio, use Object Explorer to connect to an instance of the SQL Server Database Engine.In SQL Server Management Studio, use Object Explorer to connect to an instance of the SQL Server Database Engine.
2. In Object Explorer, expand the instance of SQL Server, expand the Security folder.
3. Right-click the Server Roles folder and select New Server Role.
4. On the General Page of the New Server Role wizard:
    a. Specify the name of the new Server Role.
    b. Select the Owner for the new Server Role.
    c. Choose the appropriate Securables as they pertain to the new Server Role.
5. When a Securable is selected, apply explicit permission by selecting the checkbox to Grant, Grant with Grant or Deny permissions, as illustrated in Figure 4-1.


New Server Role - ServerRoleCreateAvailablityGroup

Ready

Select a page: General, Members, Memberships

Server role name: ServerRoleCreateAvailablityGroup

Owner: sysadmin

Securables:
- [ ] Endpoints
- [ ] Logins
- [ ] Servers
- [x] Availability Groups
  - [x] AccountingAG
  - [x] DataWarehouseAG
  - [x] FinanceAG
- [ ] Server roles

[ ] Show only selected

Permissions for selected Availability Groups:

Explicit

Permission	Grantor	Grant	With Grant	Deny
Alter		✓		
Control		✓		
Take ownership		✓		
View definition		✓		

Connection: SQL01\INSTANCE01 [PROTOTYPE\Administrator]

View connection properties

Progress: Ready

FIGURE 4-1 Applying Securables for a new Server Role tailored towards Availability Groups.

6. On the Members page, add logins that represent individuals or groups to be added to one or more server roles.

7. Finally, on the Memberships page, choose the appropriate Server Roles that the new Server Role will be a member of.

Creating Server Roles with Transact-SQL

Alternatively use Transact-SQL to create a server role and apply the appropriate permissions, members and server-role membership. The following Transact-SQL Statement creates a new Server role called DBAControlServer, adds the Windows Finance group membership and allows the group to create databases and availability groups, however, this group cannot alter logins or server audits.

```
USE [master]
CREATE SERVER ROLE [DBAControlServer] AUTHORIZATION [sysadmin]
ALTER SERVER ROLE [DBAControlServer] ADD
 MEMBER [PROTOTYPE\Finance]
GRANT CONTROL SERVER TO [DBAControlServer]
GO
GRANT CREATE ANY DATABASE TO [DBAControlServer]
GRANT CREATE AVAILABILITY GROUP TO [DBAControlServer]
DENY ALTER ANY LOGIN TO [DBAControlServer]
DENY ALTER ANY SERVER AUDIT TO [DBAControlServer]
GO
```

Audit Enhancements

With more and more organizations governed by some form of regulatory compliance the SQL Server Product Group responsible for security, decided to invest in and enhance the existing audit capabilities affiliated with servers and databases. Enhancements are seen in the following areas:

- Audit Supported on all SKUs
- Improved Resilience
- User-Defined Audit Event
- Record Filtering
- T-SQL Stack Information

Audit Supported on all SKUs

The Server and Database Audit Specifications were common features embraced by many organizations to achieve their audit and compliance needs. However, customers were not very satisfied as these features were only available in the premium SKUs of SQL Server As such, customers were forced to revert back to using SQL Trace to capture auditing information when using the Standard edition of SQL Server. As anyone can imagine, this brought about many challenges because SQL Trace has limited audit functionality compared to Server and Database Audit Specifications, at times could negatively impact performance and organizations did not have a single holistic audit solution to achieve their business goals – specifically, around collecting audit data from trace and security logs. Since, SQL Trace will eventually be retired; basic audit functionality will be available on all SQL Server 2012 SKUs.

Improved Resilience

One of the challenges organizations faced with previous versions of SQL Server was the possibility of losing audit data in the event of a failure. For example, if audit logs are being written to a network share and it suddenly becomes unavailable then audit information would no longer be captured. This would negatively impact an organization during a forensic investigation as audit data would be missing. Moreover, if the setting is configured to shut down the server if a failure takes place (ON_FAILURE = SHUTDOWN), then a SQL Server system wide outage can take place. This would not only impact audit data, but also system availability until the problem was resolved.

With SQL Server 2012, the Product Group responsible for security addressed these concerns by improving resilience associated with audit log failures by introducing new alternatives for Audit Log Failure. These enhancements include:

- **On Audit Log Failure: Continue** This new option allows SQL Server to continue operations if data cannot be written to the audit log. During the failure the system continues to attempt to write events to the audit logs, however, it is worth noting that Audit records are not retained during the failure. Only use this option if it is more important to ensure that SQL Server is operational during an audit failure and the organizations corporate policy is not in violation.
- **On Audit Log Failure: Fail Operation** When this option selected, SQL Server will fail transactions if it cannot write audit events to the audit log. However, transitions which are not governed by audits will continue to process. For example, if you have two tables such as Customer and Sales and you only audit the Customer table as it contains sensitive data. A failure will occur to transactions associated with the Customer table, however, since auditing is not enabled on the Sales table, sales data will continue to work during a Log failure.

There are additional auditing enhancements which improve resilience. These enhancements are above and beyond the improvements associated with handling audit log failures which was discussed in the previous section. These additional audit improvements include:

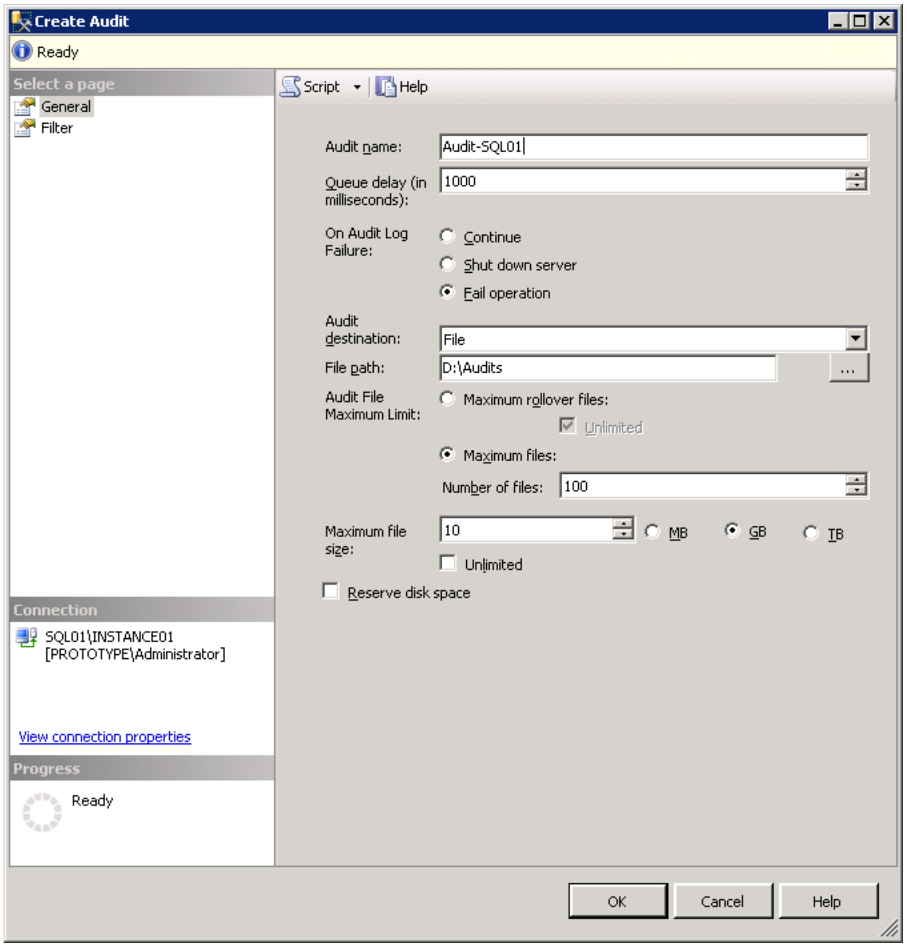
- A new option, Maximum Files or Max_Files, has been introduced when using a file as the Audit destination. Compared to the previous version of SQL Server which allowed an indeterminate number of log files to be retained, this option caps the amount of audit files to be used without rolling them over.
- With the previous version of the SQL Server it was challenging to determine whether a query was issued through a stored procedure or an application. The audit log now provides additional Transact-SQL stack frame information, therefore, auditors can make the differentiation.
- The new sp_audit_write (Transact-SQL) procedure allows applications to write customized information to the audit log as the SQL Server audit specifications now supports a user-defined audit group. A common request was to be able to capture additional information such as the application user who in many cases connected with a common login.
- New columns are added to sys.server_file_audits, sys.server_audits, and sys.fn_get_audit_file to track user-defined audit events.
- SQL Server Audit now supports the ability to filter audit events before they are written to the audit log. For more information, see the WHERE clause in CREATE SERVER AUDIT and ALTER SERVER AUDIT.
- New audit groups support the monitoring of contained database users.
- The new audit options have been added to the audit dialog boxes in Management Studio.

Create a new Audit with SSMS

The following steps illustrate some of the new Audit functionality such as Audit Log Failure and Audit File Maximum Files options when creating an audit with SQL Server Monument Studio.

1. In SQL Server Management Studio, use Object Explorer to connect to an instance of the SQL Server Database Engine.
2. In Object Explorer, expand the instance of SQL Server, expand the Security folder.
3. Right-click the Audits folder and select New Audit.
4. On the General Page of the Create Audit wizard:
    a. Specify the name of the new audit.
    b. Specify the Queue delay in milliseconds.
5. Choose the appropriate Audit Log Failure option:
    - Continue
    - Shut Down Server
    - Fail Operation
6. Select the Audit Destination:
    - Security Log
    - Application Log
    - File
7. If File was selected, then specify the following additional items:
    - File Path
    - Audit File Maximum Limit
    - Unlimited
    - Maximum Number of Files
    - Maximum File Size
8. Enable Reserve Disk Space option if you want to pre-allocate space on the disk to accommodate for the size of the audit files.

    Alternatively, use the Audit Properties Filter page to add a predicate (Where clause) to a server audit.
9. Click OK, as illustrated in Figure 4-3, to finalize the creation of the new Audit.


Create Audit

Ready

Audit name: Audit-SQL01

Queue delay (in milliseconds): 1000

On Audit Log Failure: ( ) Continue ( ) Shut down server (● Fail operation)

Audit destination: File

File path: D:\Audits

Audit File Maximum Limit: ( ) Maximum rollover files: [✓ Unlimited] (● Maximum files:)

Number of files: 100

Maximum file size: 10 ( ) MB (● GB) ( ) TB

[ ] Unlimited

[ ] Reserve disk space

Connection: SQL01\INSTANCE01 [PROTOTYPE\Administrator]

Progress: Ready

FIGURE 4-2 Creating a new SQL Server Audit with SSMS.

As you can see from the illustration, a new Audit was created with the option to Fail Operations in cases where the SQL Server Audit cannot write to the audit log. In addition, when the maximum number of files is reached such as 10, any action that causes additional audit events to be generated will fail with an error.

The same Create Audit example can be generated with Transact-SQL.

```
USE [master]

GO
CREATE SERVER AUDIT [Audit-SQL01]
TO FILE
(  FILEPATH = N'D:\Audits'
   ,MAXSIZE = 10 GB
   ,MAX_FILES = 100
   ,RESERVE_DISK_SPACE = OFF
)
WITH
(  QUEUE_DELAY = 1000
   ,ON_FAILURE = FAIL_OPERATION
)
GO
```

User-Defined Audit Event

The new User-Defined Audit Event allows applications to write custom events into the audit log to allow more flexibility in storing audit information. For example, (example to come).

Record Filtering

It worth noting that SQL Server auditing is built on top of SQL Server Extended Events (Extended Events). For those who are not familiar, Extended Events is a general event-handling system for server systems. The Extended Events infrastructure supports the correlation of data from SQL Server, and under certain conditions, the correlation of data from the operating system and database applications. Extended Events provides SQL Server Audit the framework for fast performance and throughput. SQL Server 2012 will leverage the Extended Event filtering capability; therefore, it is possible to filter unwanted events before they are written into an audit log. For example, let's say you have an application which accesses a table by using an application account. You may not want to audit this type of activity if the table is accessed by the application account; however, if a user accesses the same table from outside the application then you would want to audit this event. Therefore, you would set up a filter to exclude the application account from being audited when accessing the table.

The following example creates a database, schema, and two tables for the example. The table named DataSchema.SensitiveData will contain confidential data and access to the table must be recorded in the audit. The table named DataSchema.GeneralData does not contain confidential data. The database audit specification audits access to all objects in the DataSchema schema. The server audit is created with a WHERE clause that limits the server audit to only the SensitiveData table. The server audit presumes a audit folder exists at C:\SQLAudit.

```
CREATE DATABASE TestDB;
GO
USE TestDB;
GO
CREATE SCHEMA DataSchema;
GO
CREATE TABLE DataSchema.GeneralData (ID int PRIMARY KEY, DataField varchar(50) NOT NULL);
GO CREATE TABLE DataSchema.SensitiveData (ID int PRIMARY KEY, DataField varchar(50) NOT NULL);
GO
-- Create the server audit in the master database USE master;
GO
CREATE SERVER AUDIT AuditDataAccess TO FILE ( FILEPATH ='C:\SQLAudit\' ) WHERE object_name =
'SensitiveData' ;
```

```
GO
ALTER SERVER AUDIT AuditDataAccess WITH (STATE = ON);
GO
-- Create the database audit specification in the TestDB database USE TestDB;
GO
CREATE DATABASE AUDIT SPECIFICATION [FilterForSensitiveData] FOR SERVER AUDIT [AuditDataAccess] ADD
(SELECT ON SCHEMA::[DataSchema] BY [public]) WITH (STATE = ON);
GO
-- Trigger the audit event by selecting from tables SELECT ID, DataField FROM DataSchema.GeneralData;
SELECT ID, DataField FROM DataSchema.SensitiveData;
GO
-- Check the audit for the filtered content SELECT * FROM
fn_get_audit_file('C:\SQLAudit\AuditDataAccess_*.sqlaudit',default,default);
GO
```

T-SQL Stack Information

(Text to come)

Database Authentication Enhancements

When working with previous versions of SQL Server, a user required a login within the Database Engine in order to authenticate to a database. The login could be a Windows User account, Windows Group or a SQL Server account. At times this dependency caused authentication issues – especially with database portability. For example, if a database was migrated or failed over from one SQL Server instance (source) to another SQL Server instance (target), a database administrator had to ensure that all logins on the source SQL Server instance existed on the target SQL Server instance. If the login did not exist on the target instance, then a user, group or application would no longer be able to authenticate to the database causing a system wide outage. Organization's often experienced this challenge during failover when using Database Mirroring.

SQL Server 2012 addresses these authentication and login dependency challenges by introducing Contained Database Authentication to enhance compliance, authorization and portability of user databases. Contained Database Authentication allows users to be authenticated directly into a user database without logins which reside in the Database Engine. This feature facilitates better portability of user databases among servers, since contained database have no external dependencies.

So how do you authenticate against a user database without a login which resides in the SQL Server database Engine? When using Contained Database Authentication, user information affiliated with a login such as a username and password, is stored directly in the user database and not the master database. Authentication is robust as authenticated users cannot perform database instance level operations and can only perform DML operations inside the user databases. Another benefit of Contained Databases is that it eliminates orphaned or unused logins in the database instance, which was a management challenge many database administrators encountered with the previous versions of SQL Server.

Note Contained Databases allow authentication without Logins for both SQL Users with passwords and Windows authentication without Login. It is a great feature to leverage when implementing AlwaysOn Availability Groups.

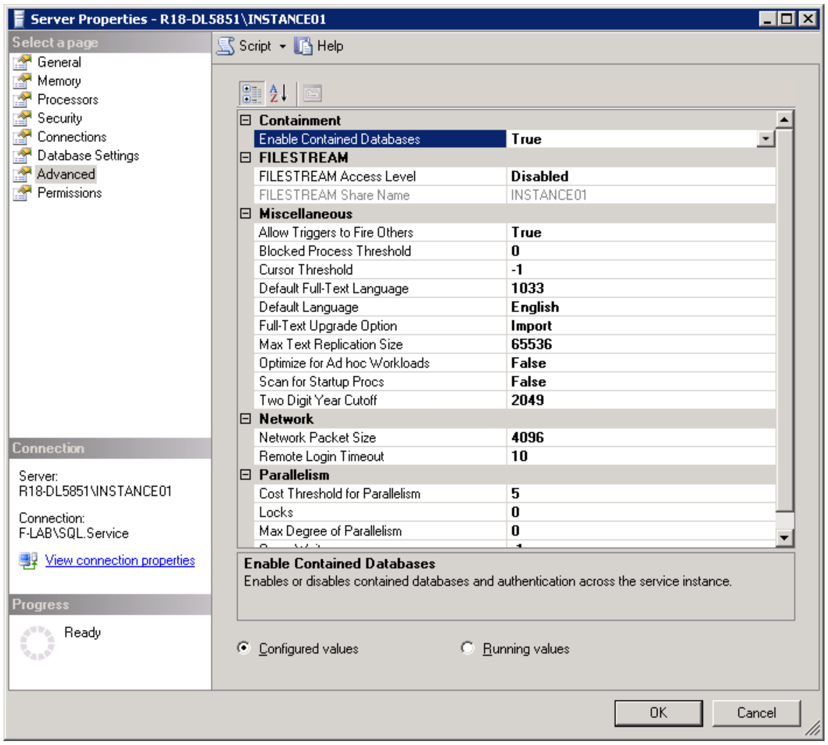
Enabling Contained Databases

Contained Database Authentication is a server wide property and very straightforward to enable. It can be enabled and disabled via the Advanced Server Properties page SQL Server Management Studio or with Transact-SQL.

Enable Contained Database Authentication with SQL Server Management Studio

Follow the steps to Enable Contained Database Authentication with SQL Server Management Studio.

1. In Object Explorer, right-click a SQL Server instance, and then click Properties.
2. Select the Advanced page and in the Containment section set the Enable Contained Databases to True, and then click Ok.



Server: R18-DL5851\INSTANCE01

Connection: F-LAB\SQL.Service

Ready

Containment	
Enable Contained Databases	True
FILESTREAM	
FILESTREAM Access Level	Disabled
FILESTREAM Share Name	INSTANCE01
Miscellaneous	
Allow Triggers to Fire Others	True
Blocked Process Threshold	0
Cursor Threshold	-1
Default Full-Text Language	1033
Default Language	English
Full-Text Upgrade Option	Import
Max Text Replication Size	65536
Optimize for Ad hoc Workloads	False
Scan for Startup Procs	False
Two Digit Year Cutoff	2049
Network	
Network Packet Size	4096
Remote Login Timeout	10
Parallelism	
Cost Threshold for Parallelism	5
Locks	0
Max Degree of Parallelism	0

Enable Contained Databases
Enables or disables contained databases and authentication across the service instance.

Enable Contained Database Authentication with Transact-SQL

The following Transact-SQL example illustrates how to use the sp_configure option to enable Contained Database Authentication for a SQL Server instance.

```
sp_configure ‘show advanced options’ 1,
GO
sp_configure 'contained database authentication', 1;
GO
RECONFIGURE;
GO
```

When using Transact-SQL, the contained database authentication option is enabled when the option is set to (1) and allows contained databases to be created, or attached to the Database Engine. On the other hand, when the option is set to (0), contained databases cannot are not supported and cannot be created, or attached to the Database Engine.

Creating Users

As mentioned earlier a Contained Database is an excellent way to decouple the user and database from the SQL Server Database Engine, therefore, an organization can achieve database portability by moving databases between instances of SQL Server. It is worth noting there are some special considerations for Contained Databases. For example, when connecting to a contained database, if the user does not have a login in the master database, the connection string must include the contained database name as the initial catalog. The initial catalog parameter is always required for a contained database user with password.

Creating a contained database user with password

The following example creates a contained database user with a password. This example can only be executed in a contained database.

```
USE AdventureWorks2012;
GO
CREATE USER SherryS
WITH PASSWORD='3e4w3cREs$mE_uk'
    , DEFAULT_LANGUAGE=[English]
    , DEFAULT_SCHEMA=[dbo]
GO
```

Creating a contained database user for a domain login

The following example creates a contained database user for a login named Kyanna in a domain named Prototype. This example can only be executed in a contained database.

```
USE AdventureWorks2012;
GO
CREATE USER [Prototype\Kyanna] ;
GO
```

In many cases you may want to change a database user which already exists within the database engine from a SQL Server authentication login to a contained database user with a password. The following Transact-SQL sp_migrate_user_to_contained procedure accomplishes this capability and is illustrated in this example.

```
sp_migrate_user_to_contained
@username = N'<User Name>',
@rename = N'keep_name',
@disablelogin = N'do_not_disable_login' ;
Go
```

Note When migrating users, be careful not to disable or delete all the administrator logins from the instance of SQL Server.

Contained Database Authentication Security Concerns

Although contained database authentication is a great way to achieve database portability, a database administrator must understand that contained databases have some security threats which need to be carefully managed. Firstly, a user can grant and create contained database users within their database without knowledge of the administrators if they have the ALTER ANY USER permission. Secondly, if a user gains access to a database via contained database authentication, they have the potential to access other databases within the database engine if these databases have the guest account enabled. It is possible to experience a denial of service attach if you create duplicate logins. For example, If a contained database user with password is created, using the same name as a SQL Server login, and if the SQL Server login connects specifying the contained database as the initial catalog, then the SQL Server login will be unable to connect. Finally, it is beneficial to leverage Windows Authentication whenever possible since Windows Authentication can take advantage Kerberos Authentication and the Windows Password Polices are far superior and much more robust.

Additional Security Enhancements

As you can see, there have been a tremendous amount of investments and enhancements for security with SQL Server 2012. The following sections online additional security enhancements which are above and beyond what has already been discussed in this chapter.

Cryptography Changes

More and more organizations are demanding the highest forms of security when it comes to using encryption to protect their data. With SQL Server 2012, the product group responsible for security has greatly enhanced SQL Server cryptography; therefore, organizations can deploy SQL Server with the highest level of confidence when achieving compliance. The major cryptography enhancements include the following:

- **Advanced Encryption Standard (AES)** AES is specification for encryption and supersedes DES as the industry standard. SQL Server 2012 uses the AES encryption algorithm to protect the service master key (SMK) and the database master key (DMK).
- **Certificate Key Length** When creating certificates, the maximum length of private keys imported from an external source is expanded from 3,456 to 4,096 bits.
- **Hashing Algorithms** In cryptography, SHA-2 is a set of cryptographic hash functions deveopled by the National Security Agency (NSA). With regards to SQL Server 2012, the HASHBYTES function now supports the SHA2_256, and SHA2_512 algorithms.
- **Binary Support** It is possible to create certificates from bytes when using the Transact-SQL CREATE CERTIFICATE procedure. The FROM BINARY option allows specifying the binary description of an ASN encoded certificate.

SharePoint Active Directory

SharePoint and SQL Server are two tightly-coupled technologies to deliver business productivity, business intelligence and reports to organizations. New SharePoint and Active Directory security modules have been introduced in order to better secure end user reports shared and published in SharePoint. Enhanced security models provide control at row and column levels and allow organizations the ability to better achieve the following:

- Enforce password policies
- Use roles and proxy accounts
- Provide security enhanced metadata access
- Enhance security features with execution context

Provisioning Enhancements

There have been three modifications to further bolster security and role separation during the provisioning process of the SQL Server Database Engine during installation.

- The BUILTIN\administrators and Local System (NT AUTHORITY\SYSTEM) are no longer automatically added in the sysadmin fixed server role. However, Local administrators can still access the Database Engine when in single user mode.
- SQL Server now supports Managed Service Accounts and Virtual Accounts when installed on Windows 7 or Windows Server 2008 R2
- The protection of operating services under a per-service SID is now extended to all operating systems

New Permissions

New permissions are available for securing and managing authorization to elements within the database. These new permissions include:

- New GRANT, REVOKE, and DENY permissions to a SEARCH PROPERTY LIST are available.
- New GRANT, REVOKE, and DENY permissions to CREATE SERVER ROLE and ALTER ANY SERVER ROLE.

Chapter 6

Integration Services

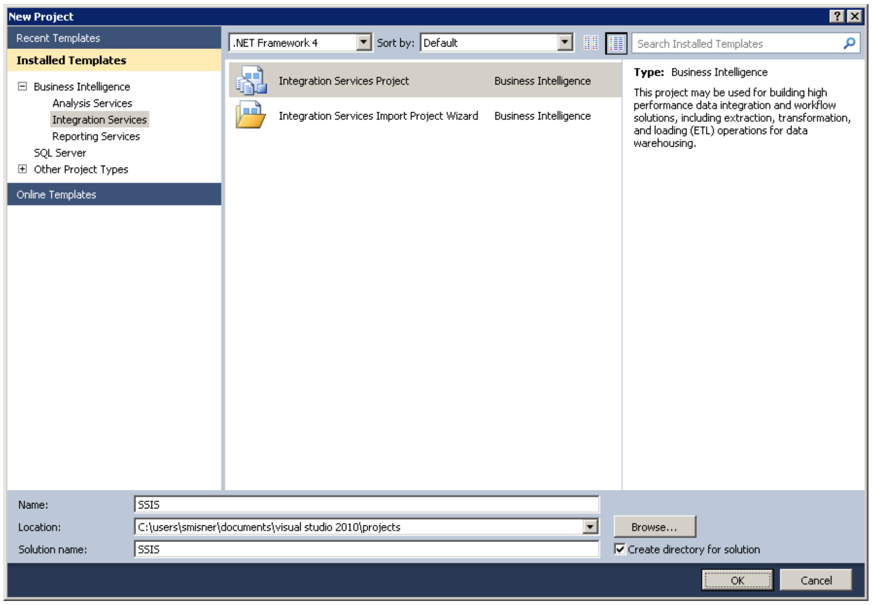
Since its initial release in Microsoft SQL Server 2005, Integration Services has had incremental changes in subsequent versions of the product. However, those changes were trivial in comparison to the number of enhancements, performance improvements, and new features introduced in SQL Server Code-Named "Denali" Integration Services. This product overhaul affects every aspect of Integration Services, from development to deployment to administration.

Developer Experience

The first change that you notice as you create a new Integration Services project is that Business Intelligence Development Studio (BIDS) is now a Visual Studio 2010 shell called SQL Server Data Tools (SSDT). The Visual Studio environment alone introduces some slight user interface changes from the previous version of BIDS. However, there are several more significant interface changes of note that are specific to Integration Services. These enhancements to the interface help you to learn about the package development process if you are new to Integration Services, and enable you to develop packages more easily if you are already have experience with Integration Services. If you are already an Integration Services veteran, you will also notice the enhanced appearance of tasks and data flow components with rounded edges and new icons.

Add New Project Dialog Box

To start working with Integration Services in BIDs, you create a new project by following the same steps that you use to perform the same task in earlier releases of Integration Services. From the File menu, point to New, and then select Project. The Add New Project dialog box displays. In the Installed Templates list, you can now select the type of Business Intelligence template you want to use and then view only the templates related to your selection, as shown in Figure 6-1. When you select a template, a description of the template displays on the right side of the dialog box.



.NET Framework 4 Sort by: Default

Integration Services Project Business Intelligence

Integration Services Import Project Wizard Business Intelligence

Type: Business Intelligence

This project may be used for building high performance data integration and workflow solutions, including extraction, transformation, and loading (ETL) operations for data warehousing.

Name: SSIS

Location: C:\users\smisner\documents\visual studio 2010\projects

Solution name: SSIS

☑ Create directory for solution

FIGURE 6-1 Add New Project dialog box displays installed templates.

There are two templates available for Integration Services projects:

- **Integration Services Project** You use this template to start development with a blank package to which you add tasks and arrange those tasks into workflows. This template type was available in previous versions of Integration Services.
- **Integration Services Import Project Wizard** You use this wizard to import a project from the Integration Services catalog or from a project deployment file. (You learn more about project deployment files in the "Deployment Models" section of this chapter.) This option is useful when you want to use an existing project as a starting point for a new project, or when you need to make changes to an existing project.

Note The Integration Services Connections Project template from previous versions is no longer available.

General Interface Changes

After creating a new package, there are several changes visible in the package designer interface, as you can see in Figure 6-2:

- **SSIS Toolbox** You now work with the SSIS Toolbox to add tasks and data flow components to a package, rather than the Visual Studio toolbox that you use in earlier versions of Integration Services. You learn more about this new toolbox in the "SSIS Toolbox" section of this chapter.

- **Parameters** The package designer includes a new tab to open the Parameters window for a package. Parameters allow you to specify run-time values for package, container, and task properties or for variables, as you learn in the ”Parameters” section of this chapter.
- **Variables button** This new button on the package designer toolbar provides quick access to the Variables window. You can also continue to open the window from the SSIS menu or by right-clicking the package designer and selecting the Variables command.
- **SSIS Toolbox button** This button is also new in the package designer interface and allows you to open the SSIS Toolbox when it is not visible. As an alternative, you can open the SSIS Toolbox from the SSIS menu or by right-clicking the package designer and selecting the SSIS Toolbox command.
- **Getting Started** This new window displays below the Solution Explorer window and provides access to links to videos and samples that you can use to learn how to work with Integration Services. This window includes the Always Show In New Project checkbox, which you can clear if you prefer not to view the window after creating a new project. You learn more about using this window in the next section, “Getting Started Window.”.
- **Zoom control** Both the control flow and data flow design surface now include a zoom control in the lower-right corner of the workspace. You can zoom in or out to a maximum size of 500 percent of the normal view or to a minimum size of 10 percent, respectively. As part of the zoom control, a button allows you to resize the view of the design surface to fit the window.

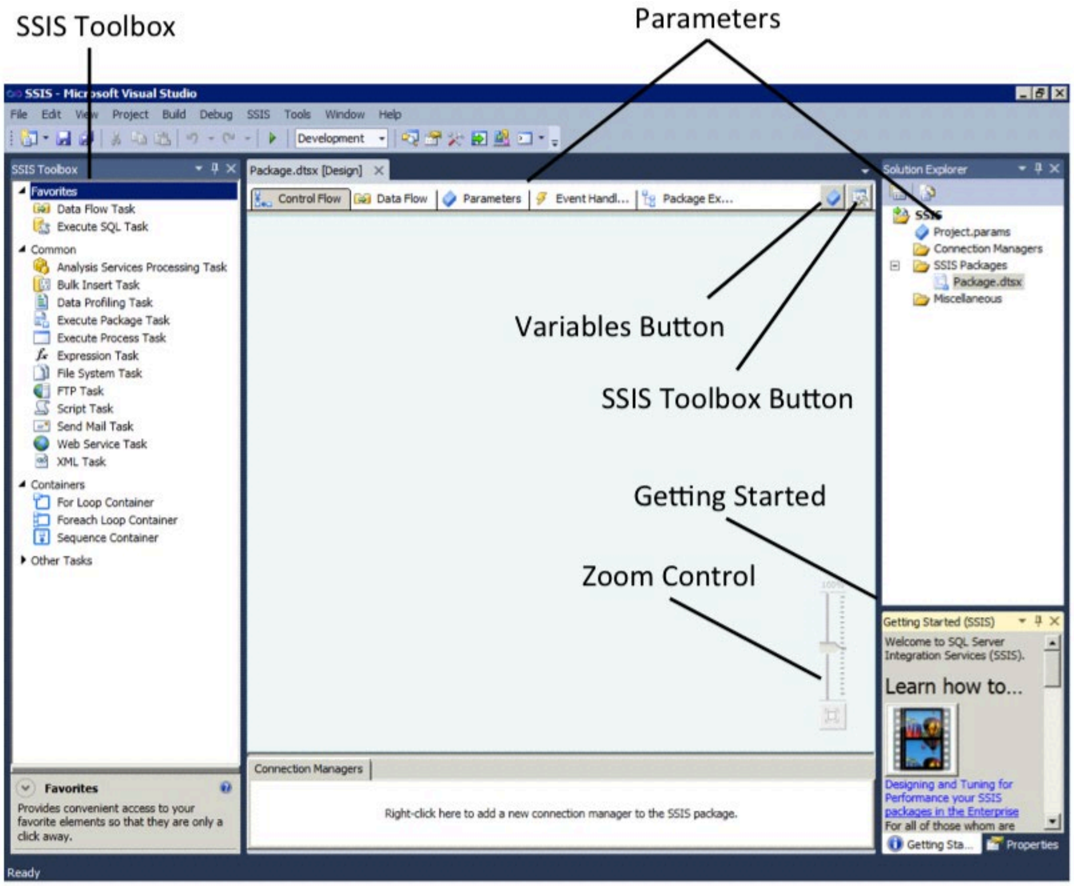


FIGURE 6-2 Package Designer Interface changes

Getting Started Window

As explained in the previous section, the Getting Started window is new to the latest version of Integration Services. Its purpose is to provide resources to new developers. It will display automatically when you create a new project unless you clear the checkbox at the bottom of the window. You must use the Close button in the upper-right corner of the window to remove it from view. Should you want to access the window later, you can choose Getting Started on the SSIS menu or right-click the design surface and select Getting Started.

In the Getting Started window, you find several links to videos and Integration Services samples. To use the links in this window, you must have Internet access. By default, the following topics are available:

- **Designing and Tuning for Performance your SSIS Packages in the Enterprise** This link provides access to a series of videos created by the SQL Server Customer Advisory Team

(SQLCAT) that explain perform how to monitor package performance and techniques to apply during package development to improve performance.

- **Parameterizing the Execute SQL Task in SSIS** This link opens a page from which you can access a brief video explaining how to work with parameterized SQL statements in Integration Services.
- **SQL Server Integration Services Product Samples** You can use this link to access the product samples available on Codeplex, Microsoft's open source project hosting site. By studying the package samples available for download, you can learn how to work with various control flow tasks or data flow components.
- **Microsoft SQL Server Community Samples: Integration Services** Another collection of sample packages is available for download from Codeplex. Many of these samples demonstrate the development and use of custom components.

Note Although the videos and samples accessible through these links were developed for previous versions of Integration Services, the principles remain applicable to the latest version. When opening a sample project in BIDS, you will be prompted to convert the project.

You can customize the Getting Started window by adding your own links to the SampleSites.xml file located in the Program Files (x86)\Microsoft SQL Server\110\DTS\Binn folder.

SSIS Toolbox

Another new window for the package designer is the SSIS Toolbox. Not only has the overall interface been improved, but you will find there is also added functionality for arranging items in the toolbox.

Interface Improvement

The first thing you notice in the SSIS Toolbox is the updated icons for most items. Furthermore, the SSIS toolbox includes a description for the item that is currently selected, allowing you to see what it does without the need to add it first to the design surface. You can continue to use drag-and-drop to place items on the design surface, or to double-click the item. However, the new behavior when you double-click is to add the item to the container that is currently selected, which is a welcome timesaver for the development process. If no container is selected, the item is added directly to the design surface.

Item Arrangement

At the top of the SSIS Toolbox, you will see two new categories, Favorites and Common, as shown in Figure 6-3. All categories are populated with items by default, but you can move items into another category at any time. To do this, right-click the item and select Move To Favorites or Move To Common. If you are working with control flow items, you have Move To Other as another choice, but if you are working with data flow items, you can choose Move To Other Sources, Move To Other Transforms, or Move To Other Destinations. You will not see the option to move an item to the category in which it already exists, nor are you able to use drag-and-drop to move items manually. If you decide to start over and return the items to their original locations, select Restore Toolbox Defaults.

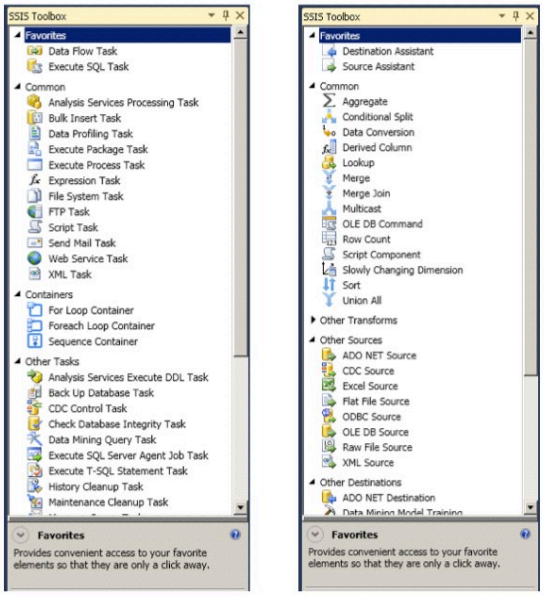


FIGURE 6-3 SSIS Toolbox for control flow and data flow.

Shared Connection Managers

If you look carefully at the Solution Explorer window, you will notice that the Data Sources and Data Source Views folders are missing, and have been replaced by a new file and a new folder. The new file is Project.params, which is used for package parameters and discussed in the "Package Parameters" section of this chapter. The Connections Manager folder is the new container for connection managers that you want to share among multiple packages.

Note If you create a Cache Connection Manager, Integration Services shares the in-memory cache with child packages using the same cache as the parent package. This feature is valuable for optimizing repeated lookups to the same source across multiple packages.

To create a shared connection manager, follow these steps:

1. Right-click the Connections Manager folder and select New Connection Manager.
2. In the Add SSIS Connection Manager dialog box, select the desired connection manager type, and then click the Add button.

3. Supply the required information in the editor for the selected connection manager type, and then click OK until all dialog boxes are closed.

A file with the CONMGR file extension displays in the Solution Explorer window within the Connections Manager folder. In addition, the file also appears in the Connections Manager tray in the package designer in each package contained in the same project. It displays with a (project) prefix to differentiate it from package connections. If you select the connection manager associated with one package and change its properties, the change affects the connection manager in all other packages.

If you change your mind about using a shared connection manager, you can convert it to a package connection. To do this, right-click the connection manager in the Connection Manager tray, and select Convert To Package Connection. The conversion removes the CONMGR file from the Connections Manager folder in Solution Explorer and from all other packages. Only the package in which you execute the conversion contains the connection. Similarly, you can convert a package connection to a shared connection manager by right-clicking the connection manager in Solution Explorer, and selecting Convert to Project Connection.

Scripting Engine

The scripting engine in SSIS is an upgrade to Visual Studio Tools for Applications (VSTA) 3.0 and includes support for the .NET Framework 4.0. When you edit a script task in the control flow or a script component in the data flow, the VSTA integrated development environment (IDE) continues to open in a separate window, but now it uses a Visual Studio 2010 shell. A significant improvement to the scripting engine is the ability to use the VSTA debug features with a Script component in the data flow.

Note As with debugging the Script task in the control flow, you must set the Run64BitRunTime project property to False when you are debugging on a 64-bit computer.

Expression Indicators

The use of expressions in Integration Services allows you as a developer to create a flexible package. Behavior can change at run-time based on the current evaluation of the expression. For example, a very common reason to use expressions with a connection manager is to dynamically change connection strings to accommodate the movement of a package from one environment to another, such as from development to production. However, earlier versions of Integration Services did not provide an easy way to determine whether a connection manager relies on an expression. In the latest version, an extra icon appears beside the connection manager icon as a visual cue that the connection manager uses expressions, as you can see in Figure 6-4.

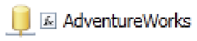


FIGURE 6-4 A visual cue that the connection manager uses an expression.

This type of expression indicator also appears with other package objects. If you add an expression to a variable or a task, the expression indicator will appear on that object.

Undo and Redo

A minor feature, but one that you will likely appreciate greatly, is the newly added ability to use Undo and Redo while developing packages in BIDS. You can now make edits in either the control flow or data flow designer surface, and use Undo to reverse a change or Redo to restore a change that you had just reversed. This capability also works in the Variables window, and on the Event Handlers and Parameters tabs. You can also use Undo and Redo when working with project parameters.

To use Undo and Redo, click the respective buttons in the standard toolbar. You can also use the Ctrl+Z and Ctrl+Y, respectively. Yet another option is to access these commands on the Edit menu.

Note The Undo and Redo actions will not work with changes that you make to the SSIS Toolbox, nor will they work with shared connection managers.

Package Sort By Name

As you add multiple packages to a project, you might find it useful to see the list of packages in Solution Explorer display in alphabetical order. In previous versions of Integration Services, the only way to resort the packages was to close the project and then reopen it. Now you can easily sort the list of packages without closing the project by right-clicking the SSIS Packages folder, and selecting Sort By Name.

Status Indicators

After executing a package, the status of each item in the control flow and the data flow displays in the package designer. In previous versions of Integration Services, the entire item was filled with green to indicate success or red to indicate failure. However, for people who are color-blind, this use of color was not helpful for assessing the outcome of package execution. Consequently, the user interface now displays icons in the upper-right corner of each item to indicate success or failure, as shown in Figure 6-5.

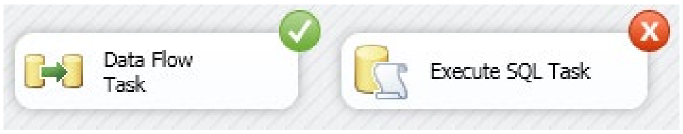


FIGURE 6-5 Item status indicators appear in the upper-right corner.

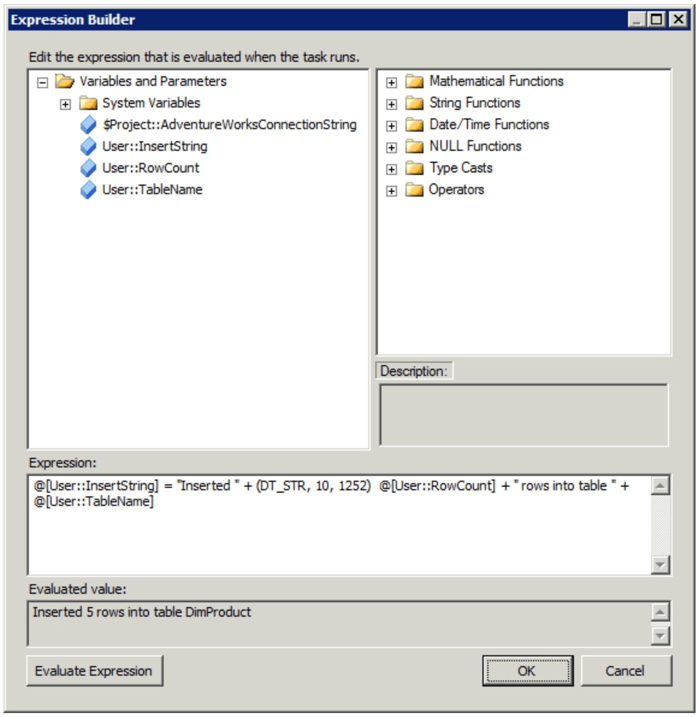
Control Flow

Apart from the general enhancements to the package designer interface, there are three notable updates for the control flow. The Expression Task is a new item available to easily evaluate an expression during the package workflow. In addition, the Execute Package Task has some changes to make it easier to configure the relationship between a parent package and child package. Another new item is the Change Data Capture Task which we discuss in the "Change Data Capture Support" section of this chapter.

Expression Task

Many of the developer experience enhancements in Integration Services affect both control flow and data flow, but there is one new feature that is exclusive to control flow. The Expression Task is new item available in the SSIS Toolbox when the control flow tab is in focus. The purpose of this task is to make it easier to assign a dynamic value to a variable.

Rather than use a Script Task to construct a variable value at run-time, you can now add an Expression Task to the workflow and use the SQL Server Integration Services Expression Language. When you edit the task, the Expression Builder opens. You start by referencing the variable and including the equals sign (=) as an assignment operator. Then provide a valid expression that resolves to a single value with the correct data type for the selected variable. Figure 6-6 illustrates an example of a variable assignment in an Expression Task.



```
@[User::InsertString] = "Inserted " + (DT_STR, 10, 1252)  @[User::RowCount] + " rows into table " +
@[User::TableName]
```

Evaluated value: Inserted 5 rows into table DimProduct

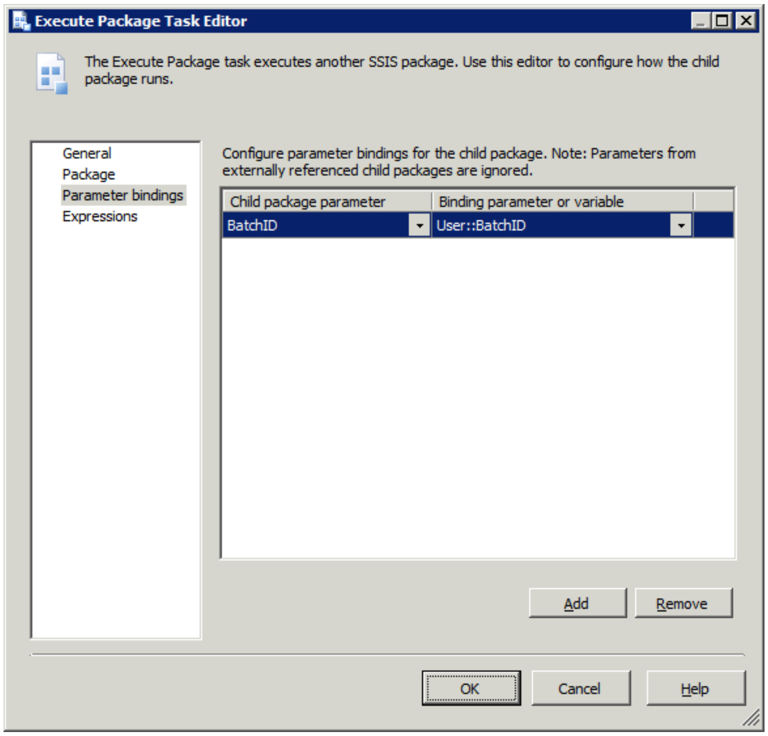
FIGURE 6-6 Variable assignment in an Expression Task.

Note The Expression Builder is an interface commonly used with other tasks and data flow components. Notice in Figure 6-6 that the list on the left side of the dialog box includes both variables and parameters. In addition, system variables are now accessible from a separate folder rather than listed together with user variables.

Execute Package Task

The Execute Package task has been updated to include a new property, ReferenceType, which you use to specify the location of the package to execute. If you select External Reference, you configure the path to the child package just as you do in earlier versions of Integration Services. If you instead select Project Reference, you then choose the child package from the drop-down list.

In addition, the Execute Package Task Editor has a new page for parameter bindings, as shown in Figure 6-7. You use this page to map a parameter from the child package to a parameter value or variable value in the parent package.


Execute Package Task Editor

The Execute Package task executes another SSIS package. Use this editor to configure how the child package runs.

Configure parameter bindings for the child package. Note: Parameters from externally referenced child packages are ignored.

Child package parameter	Binding parameter or variable
BatchID	User::BatchID

FIGURE 6-7 Parameter bindings between a parent package and a child package.

Data Flow

The data flow also has some significant updates. There are some new items, such as the Source and Destination Assistants and the DQS Cleansing transformation, and there are some improved items such as the Merge and Merge Join transformation. In addition, there are several new data flow components resulting from a partnership between Microsoft and Attunity for use when accessing ODBC connections and processing change data capture logs. We describe the change data capture components in the "Change Data Capture Support" section of this chapter. Some user interface changes have also been made to simplify the process and help you get your job done faster when designing the data flow.

Sources and Destinations

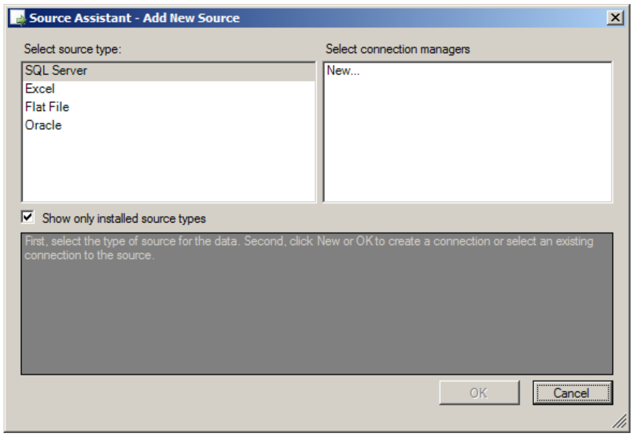
As we explore the changes in the data flow, let's start with sources and destinations.

Source and Destination Assistants

The Source Assistant and Destination Assistant are two new items available by default in the Favorites folder of the SSIS Toolbox when working with the data flow designer. These assistants help you easily create a source or a destination and its corresponding connection manager.

To create a SQL Server source in a data flow task, perform the following steps:

1. Add the Source Assistant to the data flow design surface by using drag-and-drop or by double-clicking the item in the SSIS Toolbox, which opens the Source Assistant - Add New Source dialog box as shown here.



Source Assistant - Add New Source

Select source type:

SQL Server
Excel
Flat File
Oracle

Select connection managers

New...

[✓] Show only installed source types

First, select the type of source for the data. Second, click New or OK to create a connection or select an existing connection to the source.

Note Clear the Show Installed Only checkbox to display the additional available source types that require installation of a client provider: DB2, SAP BI, Sybase, and Teradata.

2. In the Connection Managers list, select an existing connection manager or select <New> to create a new connection manager, and click OK.

3. If you selected the option to create a new connection manager, specify the server name, authentication method, and database for your source data in the Connection Manager dialog box, and click OK.

   The new data source appears on the data flow design surface and the connection manager appears in the Connection Managers tray. You next need to edit the data source to configure the data access mode, columns, and error output.

ODBC Source and Destination

The ODBC Source and ODBC Destination components, shown in Figure 6-8, are new to Integration Services in this release and are based on technology licensed by Attunity to Microsoft. Configuration of these components is similar to that of OLE DB sources and destinations. The ODBC Source supports Table Name and SQL Command as data access modes, whereas data access modes for the ODBC Destination are Table Name – Batch and Table Name – Row By Row.

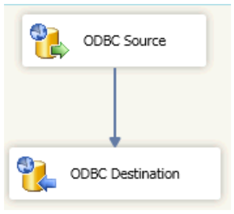


FIGURE 6-8 ODBC Source and ODBC Destination data flow components.

Flat File Source

You use the Flat File Source to extract data from a CSV or TXT file, but there were some data formats that this source did not previously support without requiring additional steps in the extraction process. For example, you could not easily use the Flat File Source with a file containing a variable number of columns. Another problem was the inability to use a character that has been designated as a qualifier as a literal value inside a string. The current version of Integration Services addresses both of these problems.

1. **Variable columns** A file layout with a variable number of columns is also known as a ragged right delimited file. Although Integration Services supports a ragged right format, a problem arises when one or more of the rightmost columns do not have values and the column delimiters for the empty columns are omitted from the file. This situation commonly occurs when the flat file contains data of mixed granularity, such as header and detail transaction records. Although a row delimiter exists on each row, Integration Services ignored the row delimiter and included data from the next row until it processed data for each expected column. Now the Flat File source correctly recognizes the row delimiter and handles the missing columns as NULL values.

Note If you expect data in a ragged right format to include a column delimiter for each missing column, you can disable the new processing behavior by changing the AlwaysCheckForRowDelimiters property of the Flat File connection manager to False.

2. **Embedded qualifiers** Another challenge with the Flat File source in previous versions of Integration Services was the use of a qualifier character inside a string encapsulated within qualifiers. For example, consider a flat file that contains the names of businesses. If a single quote is used as a text qualifier but also appears within the string as a literal value, the common practice is to use another single quote as an escape character, as shown here.

```
ID,BusinessName
404,'Margie''s Travel'
406, 'Kickstand Sellers’
```

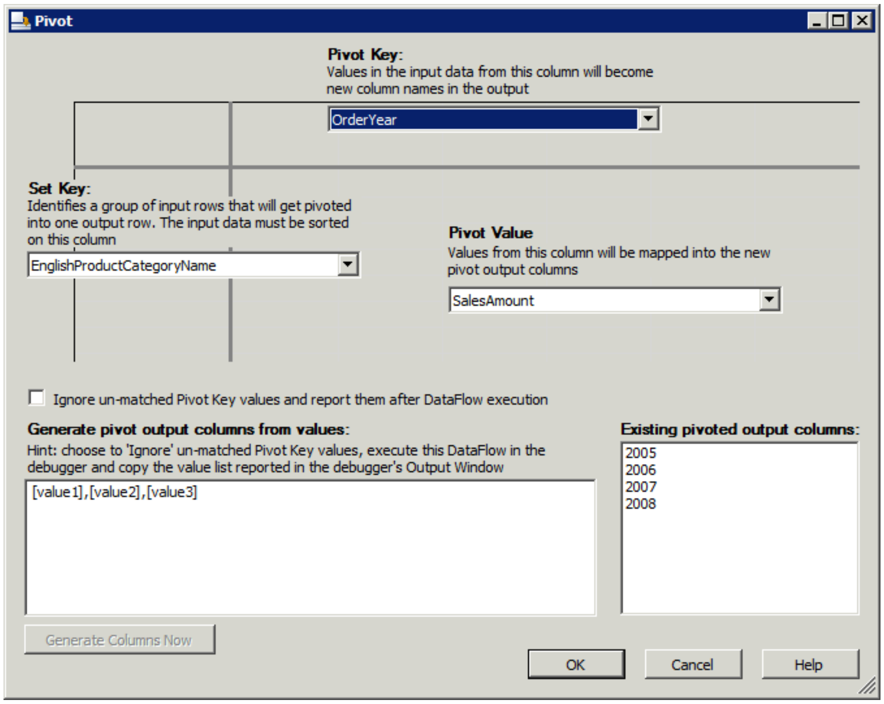
Note In the first data row in this example, previous versions of Integration Services would fail to interpret the second apostrophe in the BusinessName string as an escape character, but instead would process it as the closing text qualifier for the column. As a result, processing of the flat file returned an error because the next character in the row is not a column delimiter. This problem is now resolved in the current version of Integration Services with no additional configuration required for the Flat File source.

Transformations

Next we turn our attention to transformations.

Pivot Transformation

The user interface of the Pivot transformation in previous versions of Integration Services was a generic editor for transformations and not intuitive for converting input rows into a set of columns for each row. The new custom interface, shown in Figure 6-9, provides distinctly named fields and includes descriptions describing how each field is used as input or output for the pivot operation.


Pivot

Pivot Key:
Values in the input data from this column will become new column names in the output

OrderYear

Set Key:
Identifies a group of input rows that will get pivoted into one output row. The input data must be sorted on this column

EnglishProductCategoryName

Pivot Value
Values from this column will be mapped into the new pivot output columns

SalesAmount

☐ Ignore un-matched Pivot Key values and report them after DataFlow execution

Generate pivot output columns from values:
Hint: choose to 'Ignore' un-matched Pivot Key values, execute this DataFlow in the debugger and copy the value list reported in the debugger's Output Window

[value1],[value2],[value3]

Existing pivoted output columns:
2005
2006
2007
2008

FIGURE 6-9 Pivot transformation editor for converting a set of rows into a columns of a single row.

Row Count Transformation

Another transformation having a generic editor in previous versions is the Row Count transformation. The sole purpose of this transformation is to update a variable with the number of rows passing through the transformation. The new editor makes it very easy to change the one property for this transformation that requires configuration, as shown in Figure 6-10.

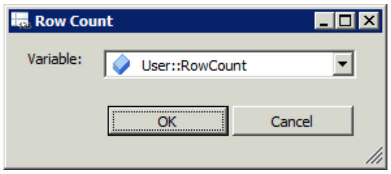


FIGURE 6-10 Row Count transformation editor for storing the current row count in a variable.

Merge and Merge Join Transformations

Both the Merge transformation and the Merge Join transformation allow you to collect data from two inputs and produce a single output of combined results. In earlier versions of Integration Services, these transformations could result in excessive memory consumption by Integration Services when data arrives from each input at different rates of speed. The current version of Integration Services better accommodates this situation by introducing a mechanism for these two transformations to better manage memory pressure in this situation. This memory management mechanism operates automatically with no additional configuration of the transformation necessary.

> **Note** If you develop custom data flow components for use in the data flow and if these components accept multiple inputs, you can use new methods in the Microsoft.SqlServer.Dts.Pipeline namespace to provide similar memory pressure management to your custom components. You can learn more about implementing these methods at "Developing Data Flow Components with Multiple Inputs," located at *http://msdn.microsoft.com/en-us/library/ff877983(v=sql.110).aspx*.

DQS Cleansing Transformation

The DQS Cleansing transformation is a new data flow component that you use in conjunction with Data Quality Services (DQS). Its purpose is to help you improve the quality of data by using rules that are established for the applicable knowledge domain. You can create rules to test data for common misspellings in a text field or to ensure that the column length conforms to a standard specification.

To configure the transformation, you select a data quality field schema that contains the rules to apply and then select the input columns in the data flow to evaluate. In addition, you configure error handling. However, before you can use the DQS Cleansing transformation, you must first install and configure DQS on a server and create a knowledge base that stores information used to detect data anomalies and to correct invalid data, which deserves a dedicated chapter. We explain not only how DQS works and how to get started with DQS, but also how to use the DQS Cleansing transformation in Chapter 7, "Data Quality Services."

Column References

The pipeline architecture of the data flow requires precise mapping between input columns and output columns of each data flow component that is part of a Data Flow Task. The typical workflow during data flow development is to begin with one or more sources, and then proceed with the addition of new components in succession until the pipeline is complete. As you plug each subsequent component into the pipeline, the package designer configures the new component's input columns to match the data type properties and other properties of the associated output columns from the preceding component. This collection of columns and related property data is also known as metadata.

If you later break the path between components to add another transformation to pipeline, the metadata in some parts of the pipeline could change because the added component can add columns, remove columns, or change column properties (such as convert a data type). In previous versions of Integration Services, an error would display in the data flow designer whenever metadata became invalid. On opening a downstream component, the Restore Invalid Column References Editor displayed to help you correct the column mapping, but the steps to perform in this editor were not always intuitive. In addition, because of each data flow component's dependency on access to metadata, it was often not possible to edit the component without first attaching it to an existing component in the pipeline.

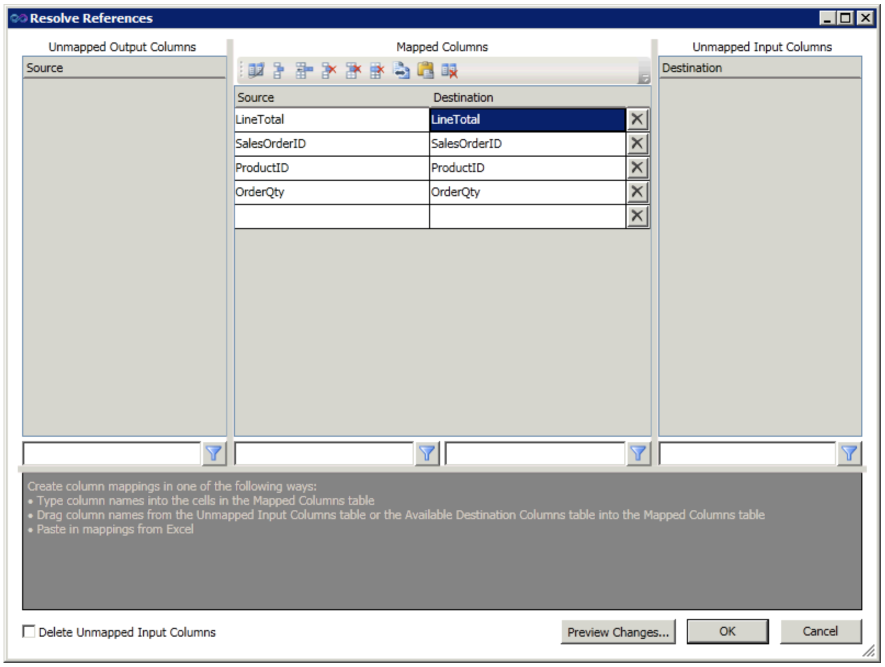
Components without Column References

Now Integration Services makes it easier to work with disconnected components. If you attempt to edit a transformation or destination that is not connected to a preceding component, a warning message box displays: "This component has no available input columns. Do you want to continue editing the available properties of this component?"

After you click Yes, the component's editor displays and you can configure the component as needed. However, the lack of input columns means that you will not be able to fully configure the component using the basic editor. If the component has an advanced editor, you can manually add input columns and then complete the component configuration. However, it is usually easier to use the interface to establish the metadata than to create it manually.

Resolve References Editor

The current version of Integration Services also makes it easier to manage the pipeline metadata if you need to add or remove components to an existing data flow. The data flow designer will display an error indicator next to any path that contains unmapped columns. If you right-click the path between components, you can select Resolve References to open a new editor that allows you to map the output columns to input columns by using a graphical interface, as shown in Figure 6-11.



Source	Destination
LineTotal	LineTotal
SalesOrderID	SalesOrderID
ProductID	ProductID
OrderQty	OrderQty

Create column mappings in one of the following ways:
• Type column names into the cells in the Mapped Columns table
• Drag column names from the Unmapped Input Columns table or the Available Destination Columns table into the Mapped Columns table
• Paste in mappings from Excel

FIGURE 6-11 Resolve References editor for mapping output to input columns.

In the Resolve References editor, you can drag a column from the Unmapped Output Columns list and add it to the Source list in the Mapped Columns area. Similarly, you can drag a column from the Unmapped Input Columns to the Destination list to link together the output and input columns. Another option is to simply type or paste in the names of the columns to map.

Tip When you have a long list of columns in any of the four groups in the editor, you can type a string in the filter box below the list to view only those columns matching the criteria that you specify. For example, if your input columns are based on data extracted from the Sales.SalesOrderDetail table in the AdventureWork2008R2 database, you can type unit in the filter box to view only the UnitPrice and UnitPriceDiscount columns.

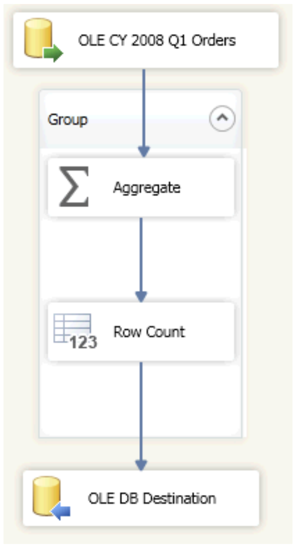
You can also manually delete a mapping by clicking the Delete Row button to the right of each mapping. After you have completed the mapping process, you can quickly delete any remaining unmapped input columns by selecting the Delete Unmapped Input Columns checkbox at the bottom of the editor. By eliminating unmapped input columns, you reduce the component's memory requirements during package execution.

Collapsible Grouping

Sometimes the data flow contains too many components to see at one time in the package designer, depending on your screen size and resolution. Now you can consolidate data flow components into groups and expand or collapse the groups. A group in the data flow is similar in concept to a sequence container in the control flow, although you cannot use the group to configure a common property for all components that it contains nor can you use it to set boundaries for a transaction or to set scope for a variable.

To create a group, follow these steps:

1. On the data flow design surface, use your mouse to draw a box around the components that you want to combine as a group. If you prefer, you can click each component while pressing the Ctrl key.
2. Right-click one of the selected components, and select Group. A group containing the components displays in the package designer, as shown below.



3. Click the arrow to the right of the group label to collapse the group.

Data Viewer

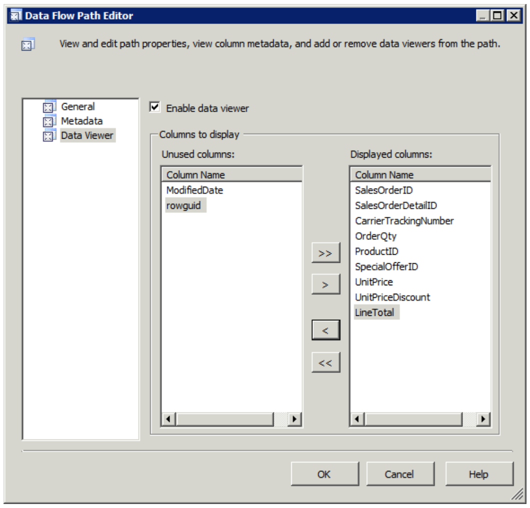
The only data viewer option available now in Integration Services is the grid view. The histogram, scatter plot, and chart views have been removed.

To use the data viewer, follow these steps:

1. Right-click the path and select Enable Data Viewer. All columns in the pipeline are automatically included.
2. If instead you want to display a subset of columns, right-click the new Data Viewer icon (a magnifying glass) on the data flow design surface, and select Edit.

3. In the Data Flow Path Editor, select Data Viewer in the list on the left.

4. Move columns from the Displayed Columns list to the Unused Columns list as applicable (shown below), and click OK.


Data Flow Path Editor

View and edit path properties, view column metadata, and add or remove data viewers from the path.

General
Metadata
Data Viewer

☑ Enable data viewer

Columns to display

Unused columns:

Column Name
ModifiedDate
rowguid

Displayed columns:

Column Name
SalesOrderID
SalesOrderDetailID
CarrierTrackingNumber
OrderQty
ProductID
SpecialOfferID
UnitPrice
UnitPriceDiscount
LineTotal

Change Data Capture Support

Change data capture is a feature introduced in the SQL Server 2008 database engine. When you configure a database for change data capture, the database engine stores information about insert, update, and delete operations on tables that you are tracking in corresponding change data capture tables. One purpose for tracking changes in separate tables is to perform extract, transform, and load (ETL) operations without adversely impacting the source table.

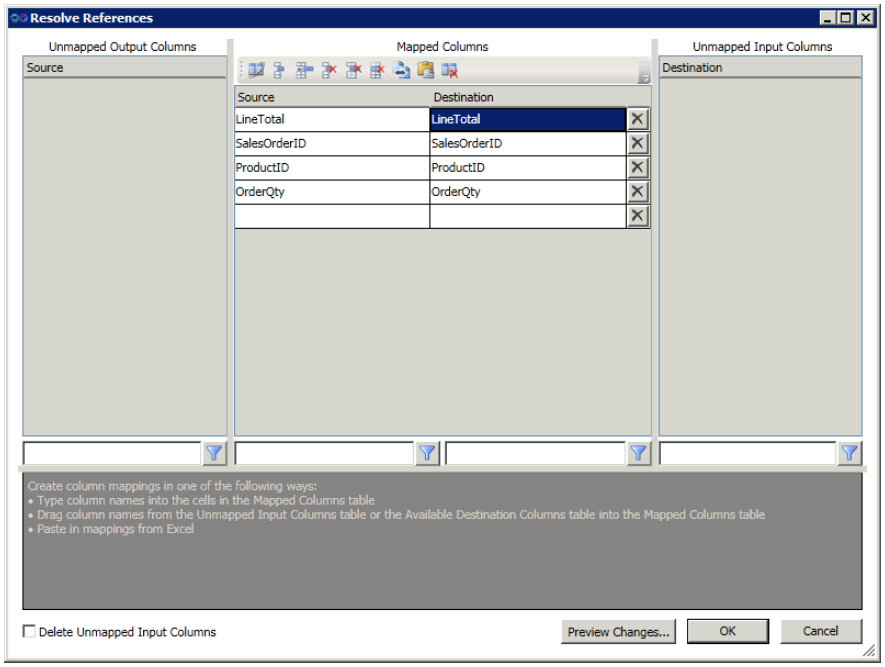
In the previous two versions of Integration Services, there were multiple steps required to develop packages that retrieve data from change data capture tables and load the results into destination. To expand Integration Services' data integration capabilities in SQL Server 2012 by supporting change data capture for SQL Server, Microsoft partnered with Attunity, a provider of real-time data integration software. As a result, new change data capture (CDC) components are available for use in the control flow and data flow, simplifying the process of package development for change data capture.

Note Change data capture support in Integration Services is available only in Enterprise, Developer, and Evaluation editions. To learn more about the change data capture feature in the database engine, see "Basics of Change Data Capture" at *http://msdn.microsoft.com/en-us/library/cc645937(SQL.110).aspx*.

CDC Control Flow

There are two types of packages that you must develop to manage change data processing with Integration Services, an initial load package for one-time execution and a trickle-feed package for ongoing execution on a scheduled basis. You use the same components in each of these packages, but configure the control flow differently. In each package type, you include a package variable with a string data type for use by the CDC components to reflect the current state of processing.

As shown in Figure 6-12, you begin the control flow with a CDC Control Task to mark the start of an initial load or to establish the log sequence number (LSN) range to process during a trickle-feed package execution. You then add a Data Flow Task that contains CDC components to perform the processing of the initial load or changed data. (We describe the components to use in this Data Flow Task later in this section.) Then you complete the control flow with another CDC Control Task to mark the end of the initial load or the successful processing of the LSN range for a trickle-feed package.

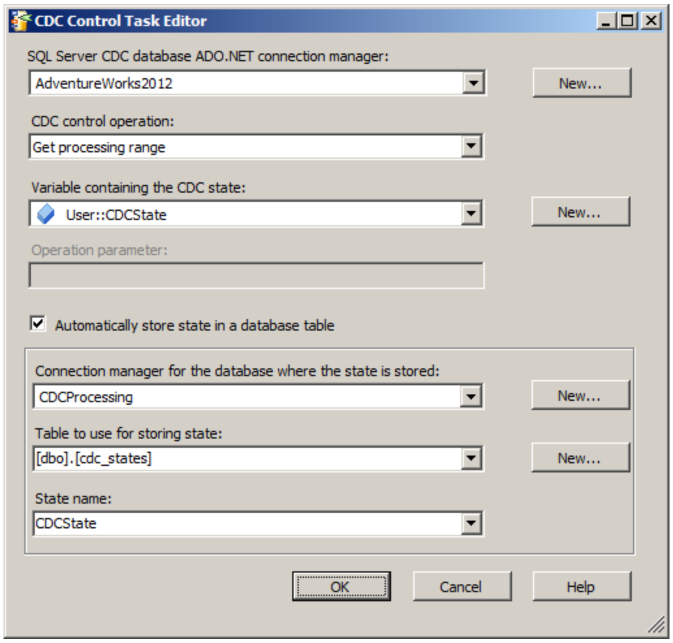


Source	Destination
LineTotal	LineTotal
SalesOrderID	SalesOrderID
ProductID	ProductID
OrderQty	OrderQty

Create column mappings in one of the following ways:
- Type column names into the cells in the Mapped Columns table
- Drag column names from the Unmapped Input Columns table or the Available Destination Columns table into the Mapped Columns table
- Paste in mappings from Excel

FIGURE 6-12 Trickle-feed control flow for change data capture processing.

Figure 6-13 shows the configuration of the CDC Control Task for the beginning a trickle-feed package. You use an ADO.NET Connection Manager to define the connection to a SQL Server database for which change data capture is enabled. You also specify a CDC control operation and the name of the CDC state variable. Optionally, you can use a table to persist the CDC state. If you do not use a table for state persistency, you must include logic in the package to write the state to a persistent store when change data processing completes and to read the state before beginning the next execution of change data processing.


CDC Control Task Editor

SQL Server CDC database ADO.NET connection manager: AdventureWorks2012

CDC control operation: Get processing range

Variable containing the CDC state: User::CDCState

Operation parameter:

☑ Automatically store state in a database table

Connection manager for the database where the state is stored: CDCProcessing

Table to use for storing state: [dbo].[cdc_states]

State name: CDCState

FIGURE 6-13 CDC Control Task Editor for retrieving the current LSN range for change data to process.

CDC Data Flow

To process changed data, you begin a Data Flow Task with a CDC Source and a CDC Splitter, as shown in Figure 6-14. The CDC Source extracts the changed data according to the specifications defined by the CDC Control Task, and then the CDC Splitter evaluates each row to determine whether the changed data is a result of an insert, update, or delete operation. Then you add data flow components to the each output of the CDC Splitter for downstream processing.

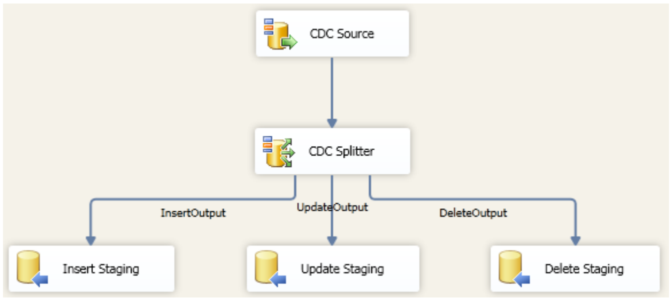
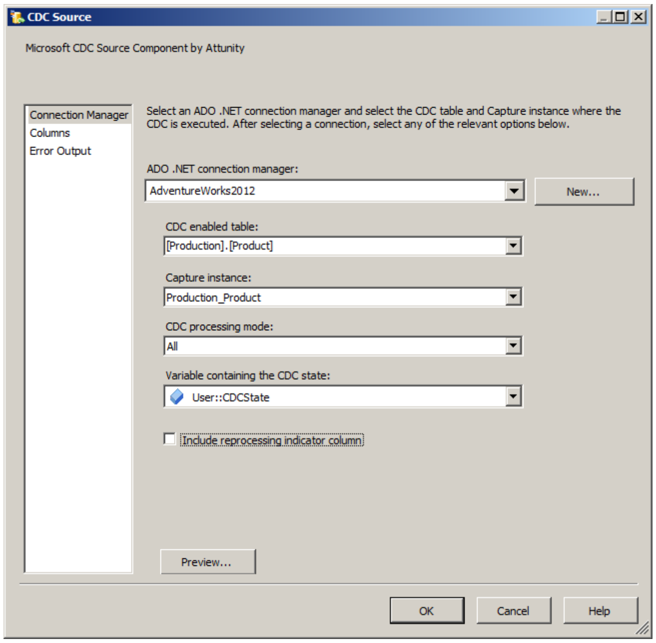


FIGURE 6-14 CDC Data Flow Task for processing changed data.

CDC Source

In the CDC Source editor, you specify an ADO.NET connection manager for the database and select a table and a corresponding capture instance. Both the database and table must be configured for change data capture in SQL Server. You also select a processing to control whether to process all change data or net changes only. The CDC state variable must match the variable you define in the CDC Control Task that executes prior to the Data Flow Task containing the CDC Source. Last, you can optionally select the Include Reprocessing Indicator Column to identify reprocessed rows for separate handling of error conditions.



Select an ADO .NET connection manager and select the CDC table and Capture instance where the CDC is executed. After selecting a connection, select any of the relevant options below.

ADO .NET connection manager: AdventureWorks2012

CDC enabled table: [Production].[Product]

Capture instance: Production_Product

CDC processing mode: All

Variable containing the CDC state: User::CDCState

☐ Include reprocessing indicator column

FIGURE 6-15 CDC Source editor for extracting change data from a CDC enabled table.

CDC Splitter

The CDC Splitter uses the value of the _$operation column to determine the type of change associated with each incoming row and assigns the row to the applicable output: InsertOutput, UpdateOutput, or Delete Output. You do not configure this transformation. Instead, you add downstream data flow components to manage the processing of each output separately.

Flexible Package Design

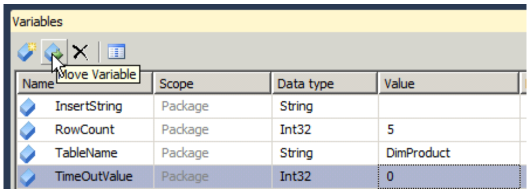
During the initial development stages of a package, you might find it easiest to work with hard-coded values in properties and expressions to ensure that your logic is correct. However, for maximum flexibility, you should use variables. In this section, we review the enhancements for variables and expressions—the cornerstones of flexible package design.

Variables

A common problem for developers when adding a variable to a package has been the scope assignment. If you had inadvertently selected a task in the control flow designer and then added a new variable in the Variables window, the variable was created within the scope of that task and could not be changed. You were required to delete the variable, clear the task selection on the design surface, and then add the variable again within the scope of the package.

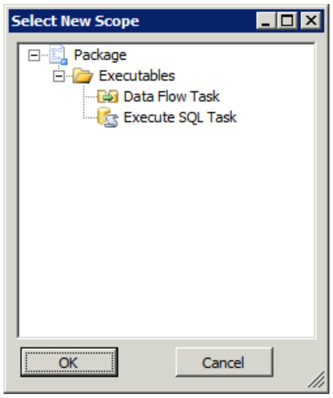
Integration Services now creates new variables with scope set to the package by default. To change the variable scope, follow these steps:

1. In the Variables window, select the variable to change, then click the Move Variable button in the Variables toolbar (the second button from the left), as shown here.



Name	Scope	Data type	Value
InsertString	Package	String	
RowCount	Package	Int32	5
TableName	Package	String	DimProduct
TimeOutValue	Package	Int32	0

2. In the Select New Scope dialog box, select the executable to have scope—the package, an event handler, container, or task—as shown here, and click OK.



Expressions

The expression enhancements in this release address a problem with expression size limitations and introduce new functions in the SQL Server Integration Services Expression Language.

Expression Result Length

Prior to the current version of Integration Services, if an expression result had a data type of DT_WSTR or DT_STR, any characters above a 4000-character limit would be truncated. Furthermore, if an expression contained an intermediate step that evaluated a result exceeding this 4000-character limit, the intermediate result would similarly be truncated. This limitation is now removed.

New Functions

The SQL Server Integration Services Expression Language now has four new functions:

- **LEFT** You can now more easily return the leftmost portion of a string rather than use the SUBSTRING function.

```
LEFT(character_expression,number)
```

- **REPLACENULL** You can use this function to replace NULL values in the first argument with the expression specified in the second expression.

```
REPLACENULL(expression, expression)
```

- **TOKEN** This function allows you to return a substring by using delimiters to separate a string into tokens and then specifying which occurrence to return.

```
TOKEN(character_expression, delimiter_string, occurrence)
```

- **TOKENCOUNT** This function uses delimiters to separate a string into tokens and then returns the count of tokens found within the string.

```
TOKENCOUNT(character_expression, delimiter_string)
```

Deployment Models

Up to now in this chapter, we have explored the changes to the package development process in BIDS, which have been substantial. Another major change to Integration Services is the concept of deployment models.

Supported Deployment Models

The latest version of Integration Services supports two deployment models:

- **Package deployment model** The package deployment model is the deployment model used in previous versions of Integration Services in which the unit of deployment is an individual package stored as a DTSX file. A package can be deployed to the file system or to the MSDB database in a SQL Server database instance. Although packages can be deployed as a group and dependencies can exist between packages, there is no unifying object in Integration Services that identifies a set of related packages deployed using the package model. To modify properties of package tasks at run-time, which is important when running a package in different environments such as development or production, you use configurations saved as DTSCONFIG files on the file system. You use either the DTExec or the DTExecUI utilities to execute a package on the Integration Services server, providing arguments on the command-line or in the graphical interface when you want to override package property values at run-time manually or by using configurations.

- **Project deployment model** With this deployment model type, the unit of deployment is a project, stored as an ISPAC file, which in turn is a collection of packages and parameters. You deploy the project to the Integration Services Catalog, which we describe in a separate section of this chapter. Instead of configurations, you use parameters (as described later in the "Parameters" section) to assign values to package properties at run-time. Before executing a package, you must create an execution object in the catalog and optionally assign parameter values or environment references to the execution object. When ready, you start the execution object by using a graphical interface in SQL Server Management Studio by executing a stored procedure or by running managed code.

In addition to the characteristics described above, there are additional differences between the package deployment model and the project deployment model. Table 6-1 compares these differences.

TABLE 6-1 Deployment Model Comparison

Characteristic	**Package Deployment Model**	**Project Deployment Model**
Unit of deployment	Package	Project
Deployment location	File system or MSDB database	Integration Services catalog
Run-time property value assignment	Configurations	Parameters
Environment-specific values for use in property values	Configurations	Environment variables
Package validation	Just before execution using: DTExec Managed code	Independent of execution using: SQL Server Management Studio interface Stored procedure Managed code
Package execution	DTExec DTExecUI	SQL Server Management Studio interface Stored procedure Managed code
Logging	Configure log provider or implement custom logging	No configuration required
Scheduling	SQL Server Agent job	SQL Server Agent job
CLR integration	Not required	Required

When you create a new project in BIDS, the project is by default established as a project deployment model. You can use the Convert To Package Deployment Model command on the Project menu (or from the context menu when you right-click the project in Solution Explorer) to switch to the package deployment model. The conversion works only if your project is compatible with the package deployment model. For example, it cannot use features that are exclusive to the project deployment model, such as parameters. After conversion, Solution Explore displays an additional label after the project name to indicate the project is now configured as a package deployment model, as shown in Figure 6-16. Notice all that the Parameters node is removed from the project while the Data Sources folder is added to the project.

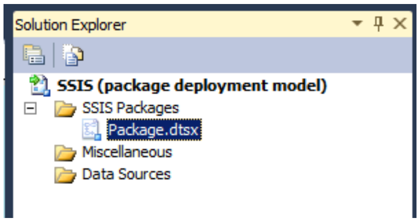


FIGURE 6-16 Package deployment model.

> **Tip** You can reverse the process by using the Project menu, or the project's context menu in Solution Explorer, to convert a package deployment model project to a project deployment model.

Project Deployment Model Features

In this section, we provide an overview of the project deployment model features to help you understand how you use these features in combination to manage deployed projects. Later in this chapter, we explain each of these features in more detail and provide links to additional information available online.

Although you can continue to work with the package deployment model if you prefer, the primary advantage of using the new project deployment model is the improvement in package management across in multiple environments. For example, a package is commonly developed on one server, then tested on a separate server, and eventually implemented on a production server. With the package deployment model, there are a variety of techniques that you can use to provide connection strings for the correct environment at run-time, each of which requires you to create at least one configuration file and optionally maintain SQL Server tables or environment variables. Although this approach is flexible, it can also be confusing and prone to error. The project deployment model continues to separate run-time values from the packages, but uses object collections in the Integration Services catalog to store these values and to define relationships between packages and these object collections known as parameters, environments, and environment variables.

- **Catalog** The catalog is a dedicated database that stores packages and related configuration information accessed at package run-time. You can manage package configuration and execution by using the catalog's stored procedures and views or by using the graphical interface in

SQL Server Management Studio.

- **Parameters** As Table 6-1 shows, the project deployment model relies on parameters to change task properties during package execution. Parameters can be created within a project scope or within a package scope. When you create parameters within a project scope, you use apply a common set of parameter values across all the packages contained in the project. You can then use parameters in expressions or tasks, much the same way that you use variables.
- **Environments** Each environment is a container of variables that you associate with a package at run-time. You can create multiple environments to use with a single package, but the package can only use variables from one environment during execution. For example, you can create environments for development, test, and production, and then execute a package using one of the applicable environments.
- **Environment variables** An environment variable contains a literal value that Integration Services assigns to a parameter during package execution. After deploying a project, you can associate a parameter with an environment variable. The value of the environment variable resolves during package execution.

Project Deployment Workflow

The project deployment workflow includes not only the process of converting design-time objects in BIDS into database objects stored in the Integration Services catalog, but also the process of retrieving database objects from the catalog to update a package design or to use an existing package as a template for a new package. To add a project to the catalog or to retrieve a project from the catalog, you use a project deployment file which has an ISPAC file extension. There are four stages of the project deployment workflow in which the ISPAC file plays a role: build, deploy, import, and convert. In this section, we review each of these stages.

Build

When you use the project deployment model for packages, you use BIDS to develop one or more packages as part of an Integration Services project. In preparation for deployment to the catalog, which serves as a centralized repository for packages and related objects, you build the Integrations Services project in BIDS to produce an ISPAC file. The ISPAC file is the project deployment file that contains project information, all packages in the Integration Services project, and parameters.

Before performing the build, there are two additional tasks that might be necessary:

- **Identify entry-point package** If one of the packages in the project is the package that triggers the execution of the other packages in the project, directly or indirectly, you should flag that package as entry-point package. You can do this by right-clicking the package in Solution Explorer and selecting Entry-point Package. An administrator uses this flag to identify the package to start when a package contains multiple projects.

- **Create project and package parameters** You use project-level or package-level parameters to provide values for use in tasks or expressions at run-time, which you learn more about how to do later in this chapter in the "Parameters" section. In BIDS, you assign parameter values to use as a default. You also mark a parameter as required, which prevents a package from executing until you assign a value to the variable.

During the development process in BIDS, you commonly execute a task or an entire package within BIDS to test results before deploying the project. BIDS creates an ISPAC file to hold the information required to execute the package and stores it in the bin folder for the Integration Services project. When you finish development and want to prepare the ISPAC file for deployment, use the Build menu or press F5.

Deploy

The deployment process uses the ISPAC file to create database objects in the catalog for the project, packages, and parameters, as shown in Figure 6-17. To do this, you use the Integration Services Deployment Wizard which prompts you for the project to deploy and the project to create or update as part of the deployment. You can also provide literal values or specify environment variables as default parameter values for the current project version. These parameter values that you provide in the wizard are stored in the catalog as server defaults for the project, and override the default parameter values stored in the package.

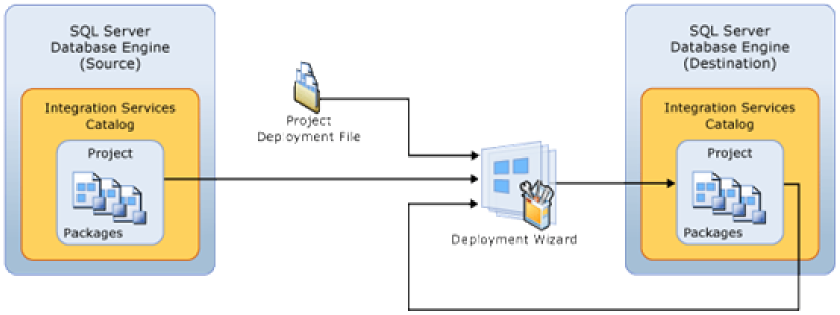


FIGURE 6-17 Deployment of the ISPAC file to the catalog.

You can launch the wizard from within BIDS by right-clicking the project in Solution Explorer, and selecting Deploy. However, if you have an ISPAC file saved to the file system, you can double-click the file to launch the wizard.

Import

When you want to update a package that has already been deployed or to use it as basis for a new package, you can import a project into BIDs from the catalog or from an ISPAC file, as shown in Figure 6-18. To import a project, you use the Integration Services Import Project Wizard which is available in the template list when you create a new project in BIDS.

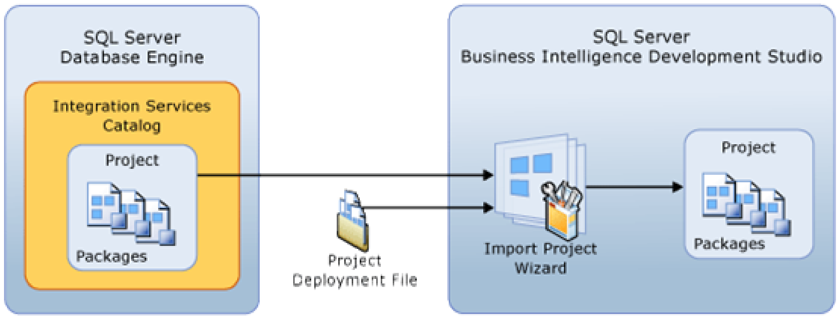


FIGURE 6-18 Import a project from the catalog or an ISPAC file.

Convert

If you have legacy packages and configuration files, you can convert them to the latest version of Integration Services, as shown in Figure 6-19. The Integration Services Project Conversion Wizard is available in both BIDS and in SQL Server Management Studio. Another option is to use the Integration Services Package Upgrade Wizard available on the Tools page of the SQL Server Installation Center.

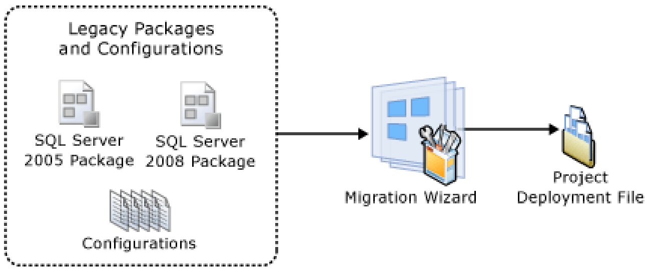


FIGURE 6-19 Convert existing DTSX files and configurations to an ISPAC file.

> **Note** You can use the Conversion Wizard to migrate packages created using SQL Server 2005 Integration Services and later. If you use SQL Server Management Studio, the original DTSX files are not modified, but used only as a source to produce the ISPAC file containing the upgraded packages.

In BIDS, open a package project, right-click the project in Solution Explorer, and select Convert To Project Deployment Model. The wizard upgrades the DTPROJ file for the project and the DTSX files for the packages.

The behavior of the wizard is different in SQL Server Management Studio. There you right-click the Projects node of the Integration Services catalog in Object Explorer, and select Import Packages. The wizard prompts you for a destination location and produces an ISPAC file for the new project and the upgraded packages.

Regardless of which method you use to convert packages, there are some common steps that occur as packages are upgraded:

- **Update Execute Package tasks** If a package in a package project contains an Execute Package task, the wizard changes the external reference to a DTSX file to a project reference to a package contained within the same project. The child package must be in the same package project that you are converting and must be selected for conversion in the wizard.
- **Create parameters** If a package in a package project uses a configuration, you can choose to convert the configuration to parameters. You can add configurations belonging to other projects to include them in the conversion process. Additionally, you can choose to remove configurations from the upgraded packages. The wizard uses the configurations to prompt you for properties to convert to parameters and also requires you to specify project scope or package scope for each parameter.
- **Configure parameters** The Conversion Wizard allows you to specify a server value for each parameter and whether to require the parameter at run-time.

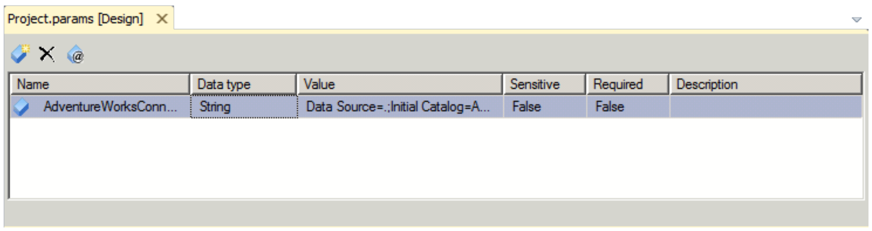
Parameters

As we explain in the previous section, parameters are the replacement for configurations in legacy packages, but only when you use the project deployment model. The purpose of configurations was to provide a way to change values in a package at run-time without requiring you to open the package and make the change directly. You can establish project-level parameters to assign a value to one or more properties across multiple packages, or you can have a package-level parameter when you need to assign a value to properties within a single package.

Project Parameters

A project parameter shares its values with all packages within the same project. To create a project parameter in BIDS, follow these steps:

1. In Solution Explorer, double-click Project.params.
2. Click the Add Parameter button on the toolbar in the Project.params window.
3. Type a name for the parameter in the Name text box, select a data type, and specify a value for the parameter as shown here. The parameter value that you supply here is known as the design default value.



Name	Data type	Value	Sensitive	Required	Description
AdventureWorksConn...	String	Data Source=.;Initial Catalog=A...	False	False	

Note The parameter value is a design-time value that can be overwritten during or after deployment to the catalog. You can use the Add Parameters To Configuration button on the toolbar (the third button from the left) to add selected parameters to Visual Studio project configurations which is useful when you for testing package executions under a variety of conditions.

4. Save the file.

Optionally, you can configure the following properties for each parameter:

- **Sensitive** By default, this property is to False. If you change it to True, the parameter value is encrypted when you deploy the project to the catalog. If anyone attempts to view the parameter value in SQL Server Management Studio or by accessing Transact-SQL views, the parameter value will display as NULL. This setting is important when you use a parameter to set a connection string property and the value contains specific credentials.
- **Required** By default, this property is also set to False. When the value is True, you must configure a parameter value during or after deployment before you can execute the package. The Integration Services engine will ignore the parameter default value that you specify on this screen when the Required property is True and deploy the package to the catalog.
- **Description** This property is optional, but allows you to provide documentation to an administrator responsible for managing packages deployed to the catalog.

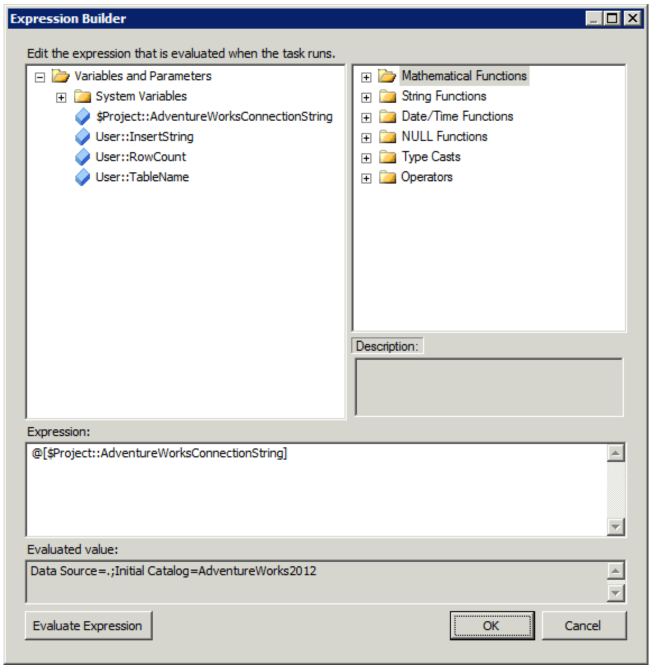
Package Parameters

Package parameters apply only to the package in which they are created and cannot be shared with other packages. The center tab in the package designer allows you to access the Parameters window for your package. The interface for working with package parameters is identical to the project parameters interface.

Parameter Usage

After creating project or package parameters, you are ready to implement the parameters in your package much like you implement variables. That is, anywhere you can use variables in expressions for tasks, data flow components, or connection managers, you can also use parameters.

As one example, you can reference a parameter in expressions, as shown in Figure 6-20. Notice the parameter appears in the Variables and Parameters list in the top left pane of the Expression Builder. You can drag the parameter to the Expression text box and use it alone or as part of a more complex expression. When you click the Evaluate Expression button, you can see the expression result based on the design default value for the parameter.



Expression:

```
@[$Project::AdventureWorksConnectionString]
```

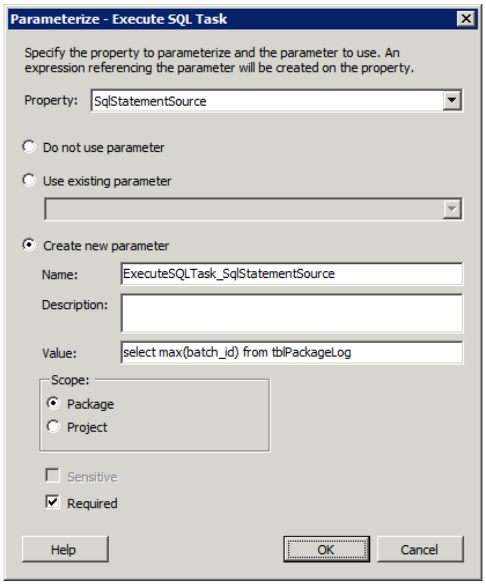
Evaluated value:

```
Data Source=.;Initial Catalog=AdventureWorks2012
```

FIGURE 6-20 Parameter usage in an expression.

Note This expression uses a project parameter which has a prefix of $Project. To create an expression that uses a package parameter, the parameter prefix is $Package.

You can also directly set a task property by right-clicking the task and selecting Parameterize on the context menu. The Parameterize dialog box displays as shown in Figure 6-21. You select a property, and then choose whether to create a new parameter or use an existing parameter. If you create a new parameter, you specify values for each of the properties that you access in the Parameters window. Additionally, you must specify whether to create the parameter within package scope or project scope.



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: ExecuteSQLTask_SqlStatementSource

Description:

Value: select max(batch_id) from tblPackageLog

Scope: [✓ Package] Project

☐ Sensitive

☑ Required

FIGURE 6-21 Parameterize task dialog box.

Post-Deployment Parameter Values

The design default values that you set for each parameter in BIDS are typically used only to supply a value for testing within the BIDS environment. You can replace these values during deployment by specifying server default values when you use the Deployment Wizard or by configuring execution values when creating an execution object for deployed projects.

Figure 6-22 illustrates the stage at which you create each type of parameter value. If a parameter has no execution value, the Integration Services engine will use the server default value when executing the package. Similarly, if there is no server default value, package execution uses the design default value. However, if a parameter is marked as required, then you must provide either a server default value or an execution value.

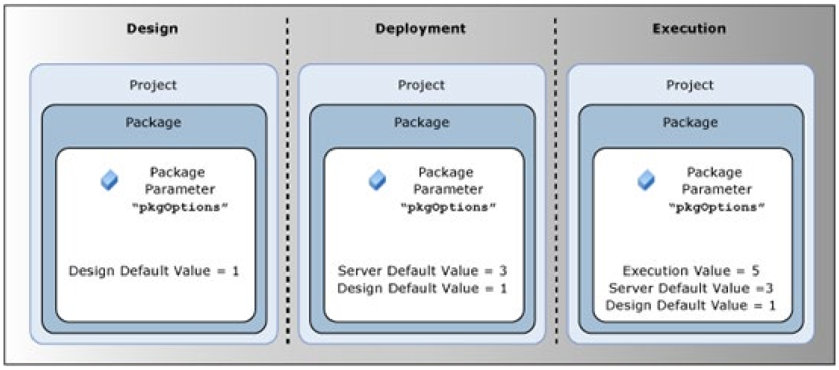
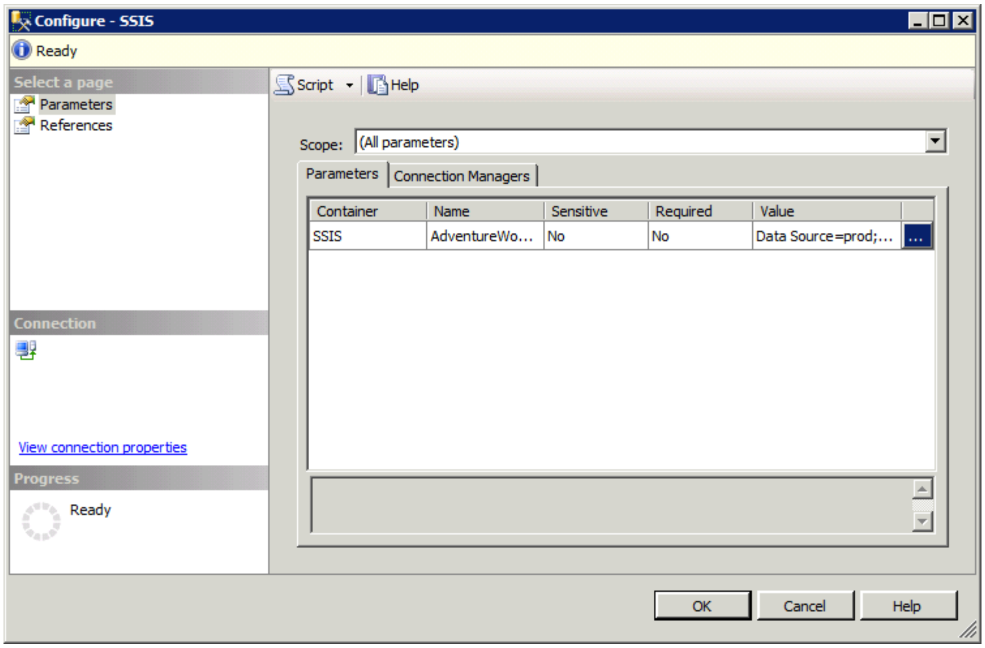


FIGURE 6-22 Parameter values by stage.

Note A package will fail when the Integration Services engine cannot resolve a parameter value. For this reason, it is recommended to validate projects and packages as described in the "Validation" section of this chapter.

Server Default Values

Server default values can be literal values or environment variable references (explained later in this chapter), which in turn are literal values. To configure server defaults in SQL Server Management Studio, you right-click the project or package in the Integration Services node in Object Explorer, select Configure, and change the Value property of the parameter, as shown in Figure 6-23. This server default value will persist even if you make changes to the design default value in BIDS and redeploy the project.



Scope: (All parameters)

Parameters | Connection Managers

Container	Name	Sensitive	Required	Value
SSIS	AdventureWo…	No	No	Data Source=prod;…

FIGURE 6-23 Server default value configuration.

Execution Parameter Values

The execution parameter value applies only to a specific execution of a package and overrides all other values. You must explicitly set the execution parameter value by using the catalog.set_execution_parameter_value stored procedure. There is no interface available in SQL Server Management Studio to set an execution parameter value.

```
set_execution_parameter_value [ @execution_id = execution_id
        , [ @object_type = ] object_type
        , [ @parameter_name = ] parameter_name
```

```
    , [ @parameter_value = ] parameter_value
```

To use this stored procedure, you must supply the following arguments:

- **execution_id** You must obtain the execution_id for the instance of the execution. You can use the catalog.executions view to locate the applicable execution_id.
- **object_type** The object type specifies whether you are setting a project parameter or a package parameter. Use a value of 20 for a project parameter and a value of 30 for a package parameter.
- **parameter_name** The name of the parameter must match the parameter stored in the catalog.
- **parameter_value** Here you provide the value to use as the execution parameter value.

Integration Services Catalog

The Integration Services catalog is a new feature to support the centralization of storage and administration of packages and related configuration information. Each SQL Server instance can host only one catalog. When you deploy a project using the project deployment model, the project and its components are added to the catalog and optionally placed in a folder that you specify in the Deployment Wizard. Each folder (or the root level if you choose not to use folders) organizes its contents into two groups—projects and environments, as shown in Figure 6-24.

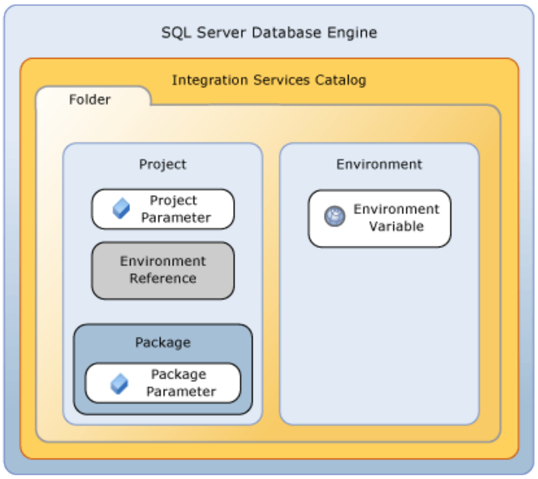
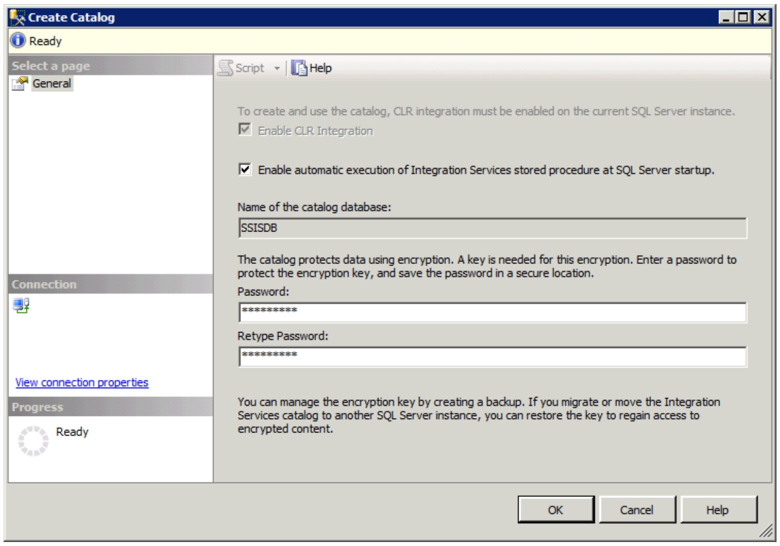


FIGURE 6-24 Catalog database objects.

Catalog Creation

Installation of Integration Services on a server does not automatically create the catalog. To do this, follow these steps:

1. In SQL Server Management Studio, connect to the SQL Server instance, right-click the Integration Services Catalog folder in Object Explorer, and select Create Catalog.
2. In the Create Catalog dialog box, you can optionally select the Enable Automatic Execution Of Integration Services Stored Procedure At SQL Server Startup checkbox. This stored procedure performs a cleanup operation when the service restarts and adjusts the status of packages that were executing when the service stopped.
3. Notice that the catalog database name cannot be changed from SSISDB, as shown below so the final step is to provide a strong password, and then click OK. The password creates a database master key that Integration Services uses to encrypt sensitive data stored in the catalog.



To create and use the catalog, CLR integration must be enabled on the current SQL Server instance.

☑ Enable CLR Integration

☑ Enable automatic execution of Integration Services stored procedure at SQL Server startup.

Name of the catalog database: SSISDB

The catalog protects data using encryption. A key is needed for this encryption. Enter a password to protect the encryption key, and save the password in a secure location.

Password: *********

Retype Password: *********

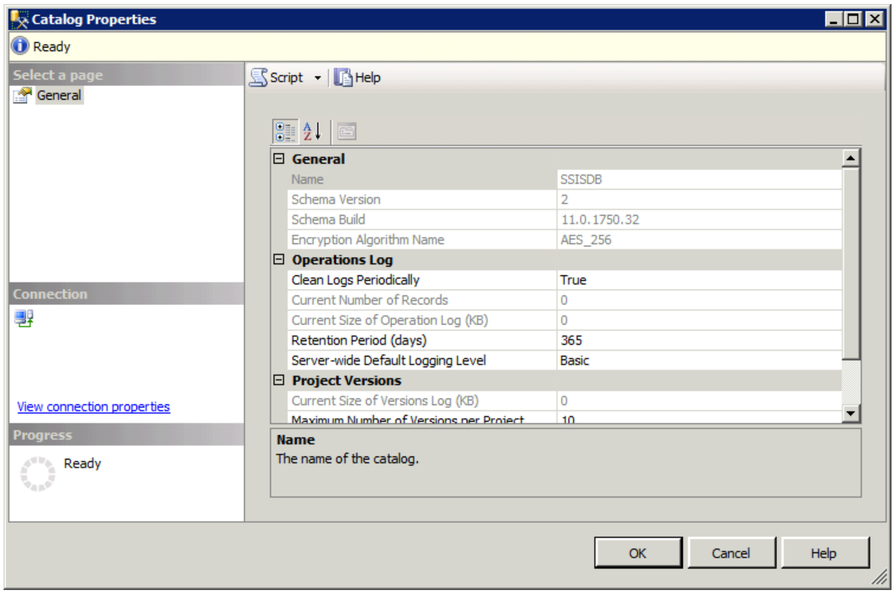
You can manage the encryption key by creating a backup. If you migrate or move the Integration Services catalog to another SQL Server instance, you can restore the key to regain access to encrypted content.

After you create the catalog, you will see it appear twice as the SSISDB database in Object Explorer. It displays under both the Databases node as well as the Integration Services node. In the Databases node, you can interact with it as you would any other database, using the interface to explore database objects. You use the Integration Services node to perform administrative tasks.

Note In most cases, there are multiple options available for performing administrative tasks with the catalog. You can use the graphical interface by opening the applicable dialog box for a selected catalog object, or you can use Transact-SQL views and stored procedures to view and modify object properties. For more information about the Tranact-SQL application programming interface (API), see *http://msdn.microsoft.com/en-us/library/ff878003(v=SQL.110).aspx*. You can also use PowerShell to perform administrative tasks by using the SSIS Catalog Managed Object Model. Refer to *http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.management.integrationservices(v=sql.110).aspx* for details about the API.

Catalog Properties

The catalog has several configurable properties. To access these properties, right-click SSISDB under the Integration Services node, and select Properties. The Catalog Properties dialog box, as shown in Figure 6-25, displays several properties.



General	
Name	SSISDB
Schema Version	2
Schema Build	11.0.1750.32
Encryption Algorithm Name	AES_256
Operations Log	
Clean Logs Periodically	True
Current Number of Records	0
Current Size of Operation Log (KB)	0
Retention Period (days)	365
Server-wide Default Logging Level	Basic
Project Versions	
Current Size of Versions Log (KB)	0
Maximum Number of Versions per Project	10

Name
The name of the catalog.

FIGURE 6-25 Catalog Properties dialog box.

Encryption

Notice in Figure 6-25 that the default encryption algorithm is AES_256. If you put the SSISDB database in single-user mode, you can choose one of the other encryption algorithms available:

- DES
- TRIPLE_DES
- TRIPLE_DES_3KEY
- DESX
- AES_128
- AES_192

Integration Services uses encryption to protect sensitive parameter values. When anyone uses the

SQL Server Management Studio interface or the Transact-SQL API to query the catalog, the parameter value will display only a NULL value.

Operations

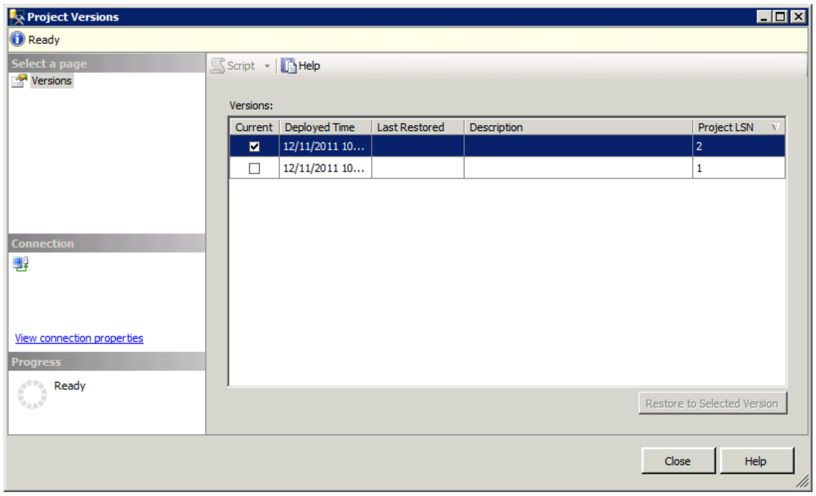
Operations include activities such as package execution, project deployment, and project validation, to name a few. Integration Services stores information about these operations in tables in the catalog. You can use the Transact-SQL API to monitor operations, or you can right-click the SSISDB database on the Integration Services node in Object Explorer and select Active Operations. The Active Operations dialog box displays the operation identifier, its type, name, the operation start time, and the caller of the operation. You can select an operation and click the Stop button to end the operation.

Periodically, older data should be purged from these tables to keep the catalog from growing unnecessarily large. By configuring the catalog properties, you can control the frequency of the SQL Server Agent job that purges the stale data by specifying how many days of data to retain. If you prefer, you can disable the job.

Project Versioning

Each time you redeploy a project with the same name to the same folder, the previous version remains in the catalog until ten versions are retained. If necessary, you can restore a previous version by following these steps:

1. In Object Explorer, locate the project under the SSISDB node.
2. Right-click the project and select Versions.
3. In the Project Versions dialog box, shown here, select the version to restore and click the Restore To Selected Version button.



Versions:

Current	Deployed Time	Last Restored	Description	Project LSN
✓	12/11/2011 10...			2
☐	12/11/2011 10...			1

4. Click Yes to confirm, and then click OK to close the information message box. Notice the selected version is now flagged as the current version, and that the other version remains available as an option for restoring.

You can modify the maximum number of versions to retain by updating the applicable catalog property. If you increase this number above the default value of ten, you should continually monitor the size of the catalog database to ensure that it does not grow too large. To manage the size of the catalog, you can also decide whether to remove older versions periodically with a SQL Server agent job.

Environment Objects

After you deploy projects to the catalog, you can create environments to work in tandem with parameters to change parameter values at execution time. An environment is a collection of environment variables. Each environment variable contains a value to assign to a parameter. To connect an environment to a project, you use an environment reference. Figure 6-26 illustrates the relationship between parameters, environments, environment variables, and environment references.

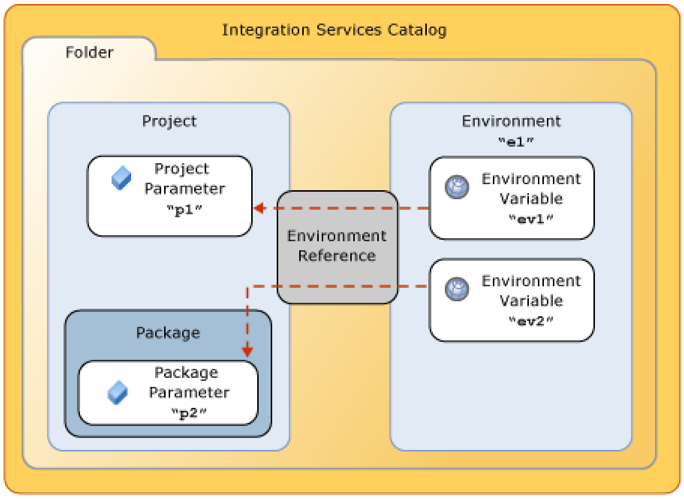


FIGURE 6-26 Environment objects in the catalog.

Environments

One convention that you might use is to create one environment for each server that you will use for package execution. For example, you might have one environment for development, one for testing, and one for production. To create a new environment using the SQL Server Management Studio interface, follow these steps:

1. In Object Explorer, expand the SSISDB node, and locate the Environments folder that corresponds to the Projects folder containing your project.

2. Right-click the Environments folder and select Create Environment.

3. In the Create Environment dialog box, type a name, optionally type a description, and click OK.

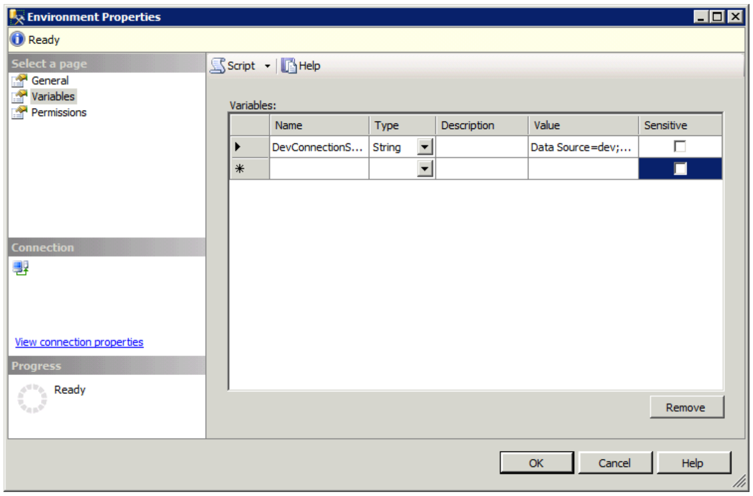
Environment Variables

For each environment, you can create a collection of environment variables. The properties that you configure for an environment variable are the same ones that you configure for a parameter, which is understandable when you consider that you use the environment variable to replace the parameter value at run-time. To create an environment variable, follow these steps:

1. In Object Explorer, locate the environment under the SSISDB node.

2. Right-click the environment and select Properties to open the Environment Properties dialog box.

3. Click Variables to display the list of existing environment variables, if any, as shown below.



Variables:

	Name	Type	Description	Value	Sensitive
▶	DevConnectionS...	String		Data Source=dev;...	☐
*					☐

4. On an empty row, type a name for the environment variable in the Name text box, select a data Type, type a Description (optional), type a Value for the environment variable, select the Sensitive checkbox if you want the value to be encrypted in the catalog. Continue adding environment variables on this page and click OK when finished.

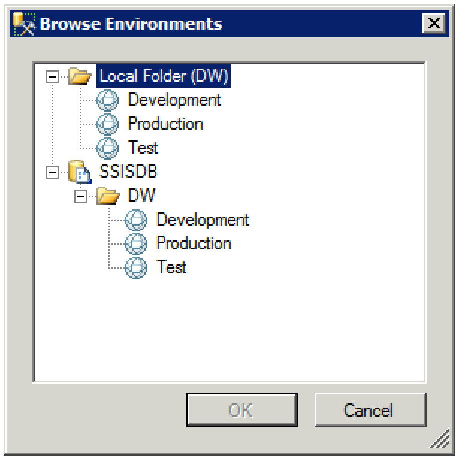
5. Repeat this process by adding the same set of environment variables to other environments that you intend to use with the same project.

Environment References

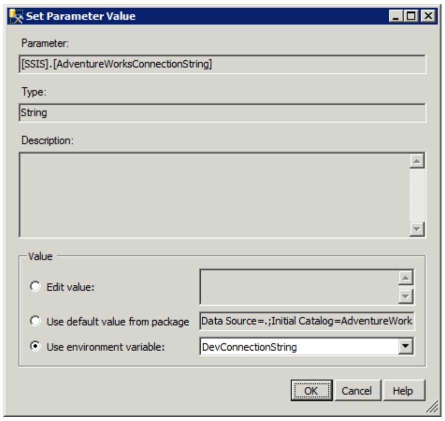
To connect environment variable to a parameter, you create an environment reference. There are two types of environment references—relative and absolute. When you create a relative environment reference, the parent folder for the environment folder must also be the parent folder for the project folder. If you later move the package to another without also moving the environment, the package execution will fail. An alternative is to use an absolute reference which maintains the relationship between the environment and the project without requiring them to have the same parent folder.

The environment reference is a property of the project. To create an environment reference, follow these steps:

1. In Object Explorer, locate the project under the SSISDB node.
2. Right-click the project and select Configure to open the Configure <Project> dialog box.
3. Click References to display the list of existing environment references, if any.
4. Click the Add button and select an environment in the Browse Environments dialog box. Use the Local Folder node for a relative environment reference or use the SSISDB node for an absolute environment reference.



5. Click OK twice to create the reference. Repeat steps 4 and 5 to add reference for all other applicable environments.
6. In the Configure <Project> dialog box, click Parameters to switch to the parameters page.
7. Click the ellipsis button to the right of the Value text box to display the Set Parameter Value dialog box, select the Use Environment Variable option, and select the applicable variable in the drop-down list, as shown below.


Set Parameter Value

Parameter: [SSIS].[AdventureWorksConnectionString]

Type: String

Description:

Value

( ) Edit value:

( ) Use default value from package: Data Source=.;Initial Catalog=AdventureWork

(•) Use environment variable: DevConnectionString

8. Click OK twice.

You can create multiple references for a project, but only one environment will be active during package execution. At that time, Integration Services will evaluate the environment variable based on the environment associated with the current execution instance as explained in the next section.

Administration

After the development and deployment processes are complete, it's time to become familiar with the administration tasks that enable operations on the server to keep running.

Validation

Before executing packages, you can use validation to verify that projects and packages are likely to run successfully, especially if you have configured parameters to use environment variables. The validation process ensures that server default values exist for required parameters, that environment references are valid, and that data types for parameters are consistent between project and package configurations and their corresponding environment variables, to name a few of the validation checks.

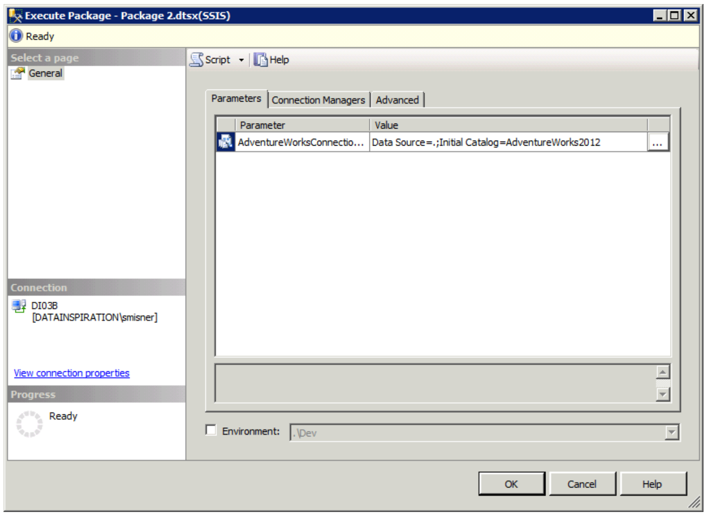
To perform the validation, right-click the project or package in the catalog, click Validate, and select the environments to include in the validation: all, none, or a specific environment. Validation occurs asynchronously, so the Validation dialog box closes while the validation processes. You can open the Integration Services Dashboard report to check the results of validation. Your other options are to right-click the SSISDB node in Object Explorer and select Active Operations or to use of the Transact-SQL API to monitor an executing package.

Package Execution

After deploying a project to the catalog and optionally configuring parameters and environment references, you are ready to prepare your packages for execution. This step requires you to create a SQL

Server object called an execution. An execution is a unique combination of a package and its corresponding parameter values, whether the values are server defaults or environment references. To configure and start an execution instance, follow these steps:

1. In Object Explorer, locate the entry-point package under the SSISDB node.
2. Right-click the project and select Execute to open the Execute Package dialog box, shown below.



Parameter	Value
AdventureWorksConnectio…	Data Source=.;Initial Catalog=AdventureWorks2012

Environment: .\Dev

3. Here you have two choices. You can either click the ellipsis button to the right of the Value and specify a literal execution value for the parameter, or you can select the Environment checkbox at the bottom of the dialog box and select an environment in the corresponding drop-down list.

You can continue configuring the execution instance by updating properties on the Connections Manager tab and by overriding property values and configuring logging on the Advanced tab. For more information about the options available in this dialog box, see *http://msdn.microsoft.com/en-us/library/hh231080(v=SQL.110).aspx*.

When you click OK to close the Execute Package dialog box, the package execution begins. Because package execution occurs asynchronously, the dialog box does not need to stay open during execution. You can use the Integration Services Dashboard report to monitor the execution status, or right-click the SSISDB node and select Active Operations. Another option is the use of the Transact-SQL API to monitor an executing package.

More often, you will schedule package execution by creating a Transact-SQL script that starts execution and save the script to a file that you can then schedule using a SQL Server agent job. You add a job step using the Operating System (CmdExec) step type and then configure the step to use the sqlcmd.exe utility and pass the package execution script to the utility as an argument. You run the job using the SQL Server Agent service account or a proxy account. Whichever account you use, it must have permissions to create and start executions.

Logging and Troubleshooting Tools

Now that Integration Services centralizes package storage and executions on the server and has access to information generated by operations, server-based logging is supported and operations reports are available in SQL Server Management Studio to help you monitor activity on the server and troubleshoot problems when they occur.

Package Execution Logs

In legacy Integration Services packages, there are two options you can use to obtain logs during package execution. One option is to configure log providers within each package and associate log providers with executables within the package. The other option is to use a combination of Execute SQL statements or script components to implement a custom logging solution. Either way, the steps necessary to enable logging are tedious in legacy packages.

With no configuration required, Integration Services stores package execution data in the [catalog].[excecutions] table. The most important columns in this table include the start and end times of package execution as well as the status. However, the logging mechanism also captures information related to the Integration Services environment such as physical memory, the page file size, and available CPUs. Other tables provide access to parameter values used during execution, the duration of each executable within a package, and messages generated during package execution. You can easily write ad hoc queries to explore package logs or build your own custom reports using Reporting Services for ongoing monitoring of the log files.

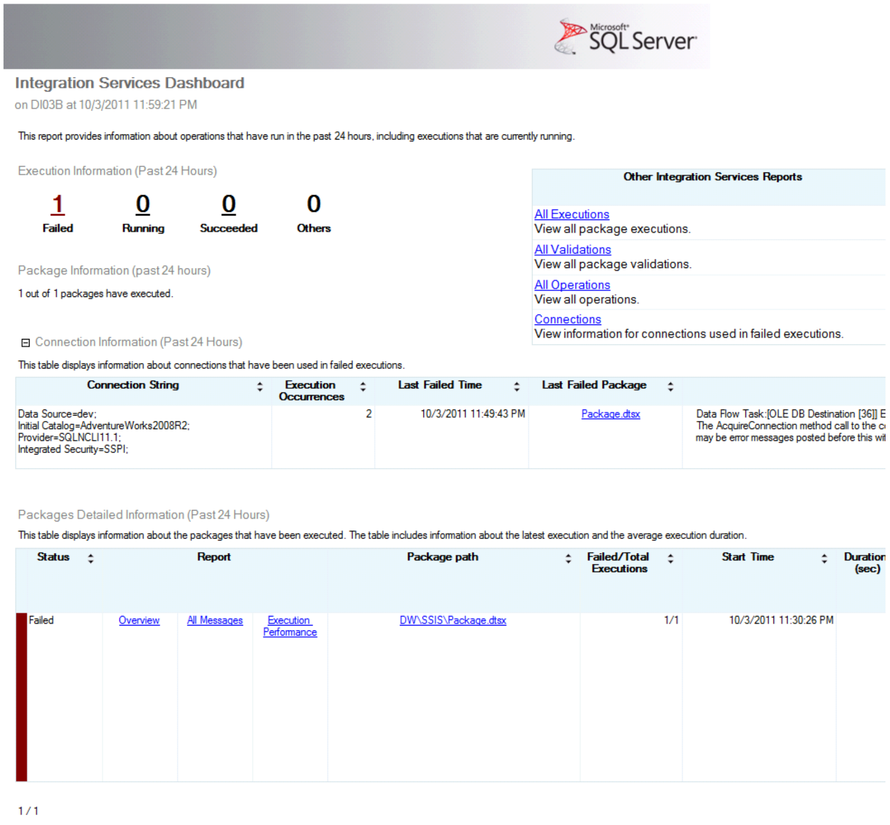
Note For a thorough walkthrough of the various tables in which package execution log data is stored, see "SSIS Logging in Denali," a blog post by Jamie Thomson at *http://sqlblog.com/blogs/jamie_thomson/archive/2011/07/16/ssis-logging-in-denali.aspx*.

Data Taps

A data tap is similar in concept to a data viewer, except that it captures data at a specified point in the pipeline during package execution outside of BIDS. You can use the T-SQL stored procedure catalog.add_data_tap to tap into the data flow during execution if the package has been deployed to SSIS. The captured data from the data flow is stored in a CSV file which you can review after package execution completes. No changes to your package are necessary to use this feature.

Reports

Before you build custom reports from the package execution log tables, review the built-in reports now available in SQL Server Management Studio for Integration Services. These reports provide information on package execution results for the past 24 hours (as shown in Figure 6-27), performance, and error messages from failed package executions. Hyperlinks in each report allow you to drill through from summary to detailed information to help you diagnose package execution problems.



Integration Services Dashboard
on DI03B at 10/3/2011 11:59:21 PM

This report provides information about operations that have run in the past 24 hours, including executions that are currently running.

Execution Information (Past 24 Hours)

1	0	0	0
Failed	Running	Succeeded	Others

Package Information (past 24 hours)
1 out of 1 packages have executed.

Other Integration Services Reports

All Executions
View all package executions.
All Validations
View all package validations.
All Operations
View all operations.
Connections
View information for connections used in failed executions.

Connection Information (Past 24 Hours)
This table displays information about connections that have been used in failed executions.

Connection String	Execution Occurrences	Last Failed Time	Last Failed Package	
Data Source=dev; Initial Catalog=AdventureWorks2008R2; Provider=SQLNCLI11.1; Integrated Security=SSPI;	2	10/3/2011 11:49:43 PM	Package.dtsx	Data Flow Task:[OLE DB Destination [36]] E The AcquireConnection method call to the c may be error messages posted before this wi

Packages Detailed Information (Past 24 Hours)
This table displays information about the packages that have been executed. The table includes information about the latest execution and the average execution duration.

Status	Report			Package path	Failed/Total Executions	Start Time	Duration (sec)
Failed	Overview	All Messages	Execution Performance	DW\SSIS\Package.dtsx	1/1	10/3/2011 11:30:26 PM	

1 / 1

FIGURE 6-27 Integration Services Operations Dashboard.

To view the reports, you right-click the SSISDB node in Object explorer, point to Reports, point to Standard Reports, and then choose from the following list of reports:

- All Executions
- All Validations
- All Operations
- Connections

Security

Packages and related objects are stored securely in the catalog using encryption. Only members of the new SQL Server database role ssis_admin or members of the existing sysadmin role have permissions to all objects in the catalog. Members of these roles can perform operations such as creating the catalog, creating folders in the catalog, and executing stored procedures, to name a few.

Members of the administrative roles delegate administrative permissions to users who need to manage a specific folder. Delegation is useful when you do not want to give these users access to the higher privileged roles. To give a user folder-level access, you grant the MANAGE_OBJECT_PERMISSIONS permission to the user.

For general permissions management, open the Properties dialog box for a folder (or any other securable object) and go to the Permissions page. On that page, you can select a security principal by name and then set explicit Grant or Deny permissions as appropriate. You can use this method to secure folders, projects, environments, and operations.

Package File Format

Although legacy packages stored as DTSX files are formatted as XML, their structure is not compatible with differencing tools and source control systems that you might use to compare packages. In the current version of Integration Services, the package file format is pretty-printed, with properties formatted as attributes rather than as elements. Moreover, attributes are listed alphabetically and attributes configured with default values have been eliminated. Collectively, these changes not only help you more easily locate information in the file, but you can more easily compare packages with automated tools and more reliably merge packages that have no conflicting changes.

Another significant change to the package file format is the replacement of the meaningless numeric lineage identifiers with a refid attribute with a text value that represents the path to the referenced object. For example, a refid for the first input column of an Aggregate transformation in a data flow task called Data Flow Task in a package called Package looks like this:

```
Package\Data Flow Task\Aggregate.Inputs[Aggregate Input 1].Columns[LineTotal]
```

Last, annotations are no longer stored as binary streams. Instead, they appear in the XML file as clear text. With better access to annotations in the file, the more likely that annotations can be programmatically extracted from a package for documentation purposes.

Chapter 7

Data Quality Services

The quality of data is a critical success factor for many data projects, whether for general business operations or business intelligence. Bad data creeps into business applications as a result of user entry, corruption during transmission, business processes, or even conflicting data standards across data sources. The Data Quality Services (DQS) feature of SQL Server 2012 is a set of technologies that you use to measure and manage data quality through a combination of computer-assisted and manual processes. When your organization has access to high quality data, your business process can operate more effectively and managers can rely on this data for better decision-making. By centralizing data quality management, you also reduce the amount of time that people spend reviewing and correcting data.

Data Quality Services Architecture

In this section, we describe the two primary components of DQS, the DQS server and Data Quality Client. The DQS architecture also includes components that are built into other SQL Server 2012 features. For example, Integration Services has the DQS Cleansing transformation that you use to apply data cleansing rules to a data flow pipeline. In addition, Master Data Services supports DQS matching so that you can de-duplicate data before adding it as master data. We explain more about these components in the "Integration" section of this chapter. All DQS components can co-exist on the same server, or can be installed on separate servers.

Data Quality Services Server

The DQS server is the core component of the architecture that manages storage of knowledge and executes knowledge-related processes. It consists of a DQS engine and multiple databases stored in a local SQL Server 2012 instance. These databases contain knowledge bases, stored procedures for managing the DQS server and its contents, and data about cleansing, matching, and data profiling activities.

Installation of the DQS Server is a multi-step process. You start by using SQL Server Setup and at minimum selecting the Database Engine and Data Quality Services on the Feature Selection page. Then you continue installation by opening the Data Quality Services folder in the Microsoft SQL Server 2012 program group on the Start menu, and launching Data Quality Server Installer. A command window opens and a prompt appears for the database master key password. You must supply a strong password having at least 8 characters and including at lease one uppercase letter, one lowercase letter, and one special character.

After you provide a valid password, installation of the DQS server continues for several minutes. In addition to creating and registering assemblies on the server, the installation process creates the following databases on a local instance of SQL Server 2012:

- **DQS_MAIN** As its name implies, this is the primary database for the DQS sever. It contains the published knowledge bases as well as the stored procedures that support the DQS engine. Following installation, this database also contains a sample knowledge base called DQS data which you can use to cleanse country data or data related to geographical locations in the United States.
- **DQS_PROJECTS** This database is for internal use by the DQS server to store data related to managing knowledge bases and data quality projects.
- **DQS_STAGING_DATA** You can use this database as intermediate storage for data that you want to use as source data for DQS operations. DQS can also use this database to store processed data that you can later export.

Before users can use client components with the DQS server, a user with sysadmin privileges must create a SQL Server login for each user and map each user to the DQS_MAIN database using one of the following database roles created at installation of the DQS server:

- **dqs_administrator** A user assigned to this role has all privileges available to the other roles plus full administrative privileges with the exception of adding new users. Specifically, a member of this role can stop any activity or stop a process within an activity, and can perform any configuration task using Data Quality Client.
- **dqs_kb_editor** A member of this role can perform any DQS activity except administration. A user must be a member of this role to create or edit a knowledge base.
- **dqs_kb_operator** This role is the most limited of the database roles, allowing its members only to edit and execute data quality projects and to view activity monitoring data.

Important You must use SQL Server Configuration Manager to enable the TCP/IP protocol for the SQL Server instance hosting the DQS databases before remote clients can connect to the DQS server.

Data Quality Client

Data Quality Client is the primary user interface for the DQS server that you install as a stand-alone application. Business users can use this application to interactively work with data quality projects, such as cleansing or data profiling. Data stewards use Data Quality Client to create or maintain knowledge bases, and DQS administrators use it to configure and manage the DQS server.

To install this application, use SQL Server Setup and select Data Quality Client on the Feature Selection page. It requires .NET Framework 4.0, which installs automatically if necessary.

Note If you plan to import data from Microsoft Excel, you must install Excel on the Data Quality Client computer. DQS supports both 32-bit and 64-bit versions of Excel 2003, but supports only the 32-bit version of Excel 2007 or 2010 unless you save the workbook as an XLS or CSV file.

If you have been assigned to one of the DQS database roles, or if you have sysadmin privileges on the SQL Server instance hosting the DQS server, you can open Data Quality Client, which is found in Data Quality Services folder of the Microsoft SQL Server 2012 program group on the Start menu. You must then identify the DQS server to establish the client-server connection. If you click the Options button, you can select a checkbox to encrypt the connection.

After opening Data Quality Client, the home screen provides access to the following three types of tasks:

- **Knowledge Base Management** You use this area of Data Quality Client to create a new knowledge base, edit an existing knowledge base, use knowledge discovery to enhance a knowledge base with additional values, or create a matching policy for a knowledge base.
- **Data Quality Projects** In this area, you create and run data quality projects to perform data cleansing or data matching tasks.
- **Administration** This area allows you to view the status of knowledge base management activities, data quality projects, and DQS cleansing transformations used in Integration Services packages. In addition, it provides access to configuration properties for the DQS server, logging, and reference data services.

Knowledge Base Management

In DQS, you create a knowledge base to store information about your data, including valid and invalid values and rules to apply for validating and correcting data. You can generate a knowledge base from sample data or you can manually create one. You can reuse a knowledge base with multiple data quality projects and enhance it over time with the output of cleansing and matching projects. You can give responsibility for maintaining the knowledge base to data stewards. There are three activities that you or data stewards perform using the Knowledge Base Management area of Data Quality Client: Domain Management, Knowledge Discovery, and Matching Policy.

Domain Management

After creating a knowledge base, you manage its contents and rules through the Domain Management activity. A knowledge base is a logical collection of domains with each domain corresponding to a single field. You can create separate knowledge bases for customers and products, or you could combine these subject areas into a single knowledge base. After you create a domain, you define trusted values, invalid values, and examples of erroneous data. In addition to this set of values, you also manage properties and rules for a domain.

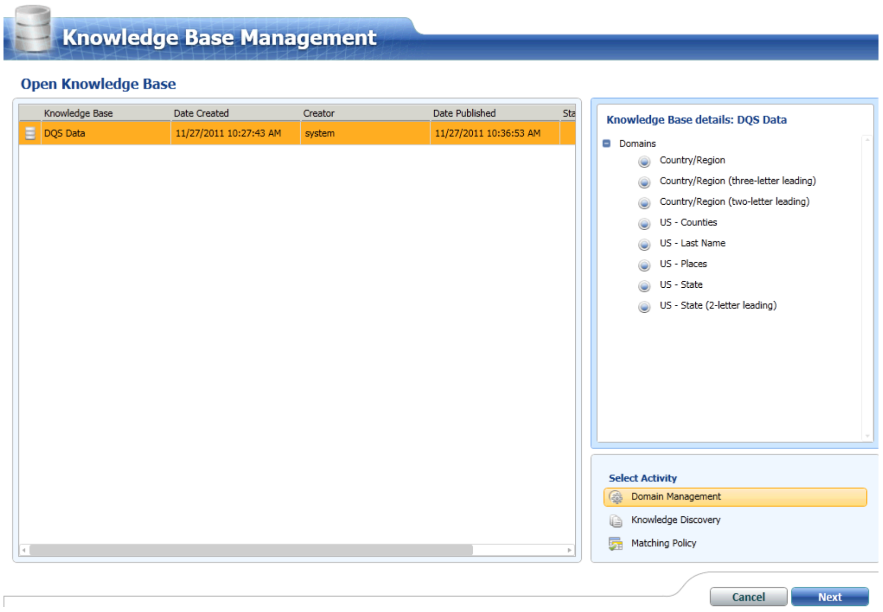
To prevent potential conflicts resulting from multiple users from working on the same knowledge base at the same time, DQS locks the knowledge base when you begin a new activity. To unlock an activity, you must publish or discard the results of your changes.

Note If you have multiple users who have responsibility for managing knowledge, keep in mind that only one user at a time can perform the Domain Management activity for a single knowledge base. Therefore, you might consider using separate knowledge bases for each area of responsibility.

DQS Data Knowledge Base

Before creating your own knowledge base, you can explore the automatically installed knowledge base, DQS Data, to gain familiarity with the Data Quality Client interface and basic knowledge base concepts. By exploring DQS Data, you can learn about the types of information that a knowledge base can contain.

To get started, open Data Quality Client and click the Open Knowledge Base button on the home screen. DQS Data displays as the only knowledge base on the Open Knowledge Base screen. When you select DQS Data in the list of existing knowledge bases, you can see the collection of domains associated with this knowledge base, as shown in Figure 7-1.



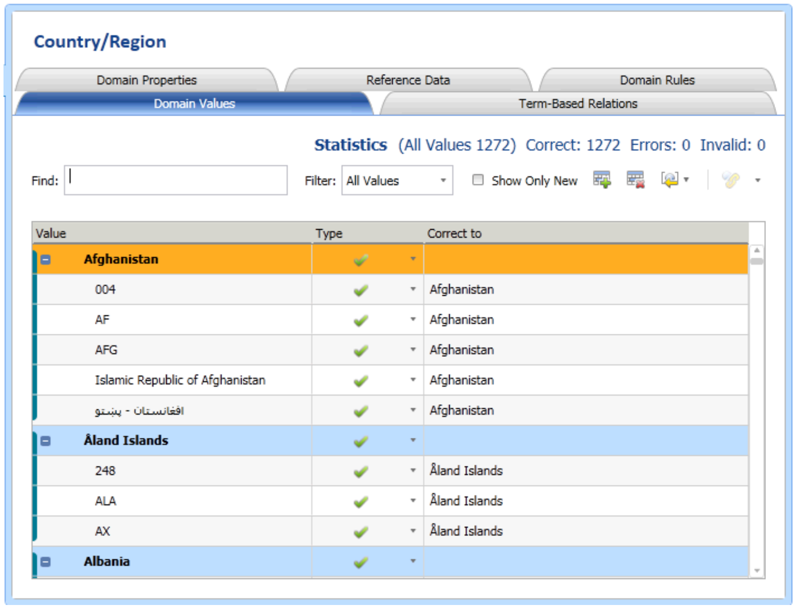
Knowledge Base	Date Created	Creator	Date Published	Sta
DQS Data	11/27/2011 10:27:43 AM	system	11/27/2011 10:36:53 AM	

Knowledge Base details: DQS Data

Domains
- Country/Region
- Country/Region (three-letter leading)
- Country/Region (two-letter leading)
- US - Counties
- US - Last Name
- US - Places
- US - State
- US - State (2-letter leading)

FIGURE 7-1 Knowledge base details for DQS Data.

Choose the Domain Management activity in the bottom right corner of the window, and click Next to explore the domains in this knowledge base. On the Domain Management screen, you can choose a domain, such as Country/Region, and access the knowledge associated with that domain. For example, you can click the Domain Values tab to see how values that represent a specific country or region will be corrected when you use DQS to perform data cleansing.


Country/Region

Domain Properties | Reference Data | Domain Rules | Domain Values | Term-Based Relations

Statistics (All Values 1272) Correct: 1272 Errors: 0 Invalid: 0

Find: | Filter: All Values | Show Only New

Value	Type	Correct to
Afghanistan	✓	
004	✓	Afghanistan
AF	✓	Afghanistan
AFG	✓	Afghanistan
Islamic Republic of Afghanistan	✓	Afghanistan
افغانستان - پښتو	✓	Afghanistan
Åland Islands	✓	
248	✓	Åland Islands
ALA	✓	Åland Islands
AX	✓	Åland Islands
Albania	✓	

FIGURE 7-2 Domain values for the Country/Region domain.

The DQS Data knowledge base by default contains only domain values. You can add other knowledge to make this data more useful for your own data quality projects. For example, you could add domain rules to define conditions that identify valid data or create term-based relations to define how to make corrections to terms found in a string value. More information about these other knowledge categories is provided in the "New Knowledge Base" section of this chapter.

Tip When you finish reviewing the knowledge base, click the Cancel button and click Yes to confirm that you do not want to save your work.

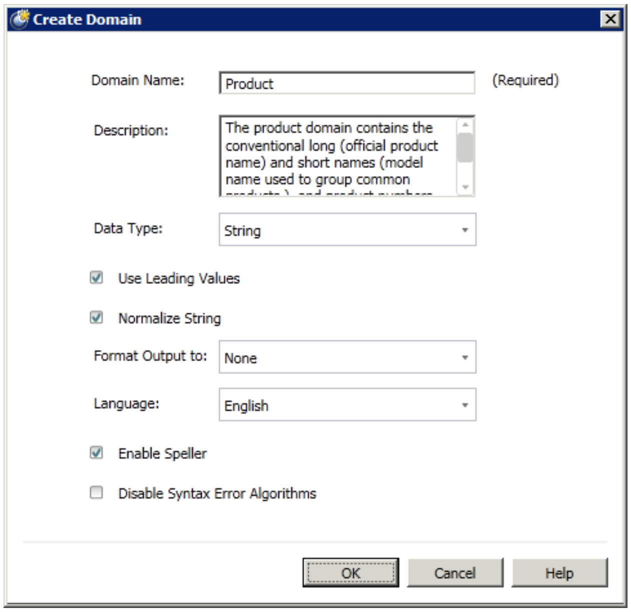
New Knowledge Base

On the home screen of Data Quality Client, click the New Knowledge Base button to start the process of creating a new knowledge base. At minimum, you provide a name for the knowledge base, but you can optionally add a description. Then you must specify whether you want to create an empty knowledge base or to create your knowledge base from an existing one, such as DQS Data. Another option is to create a knowledge base by importing knowledge from a DQS file, which you create by using the export option from any existing knowledge base.

After creating the knowledge base, you select the Domain Management activity, and click Next so that you can add one or more domains in the knowledge base. To add a new domain, you can create the domain manually or import a domain from an existing knowledge base using the applicable button in the toolbar on the Domain Management screen. As another option, you can create a domain from the output of a data quality project.

Domain When you create a domain manually, you start by defining the following properties for the domain (as shown in Figure 7-3):

- **Domain Name** The name that you provide for the domain must be unique within the knowledge base and must be 256 characters or less.
- **Description** You can optionally add a description to provide more information about the contents of the domain. The maximum number of characters for the description is 2,048.
- **Data Type** Your options here are String, Date, Integer, or Decimal.
- **Use Leading Values** When you select this option, the output from a cleansing or matching data quality project will use the leading value in a group of synonyms. Otherwise, the output will be the input value or its corrected value.
- **Normalize String** This option appears only when you select String as the Data Type. You use it to remove special characters from the domain values during the data processing stage of knowledge discovery, data cleansing, and matching activities. Normalization might be helpful for improving the accuracy of matches when you want to de-duplicate data because punctuation might be used inconsistently in strings that otherwise is a match.
- **Format Output To** When you output the results of a data quality project, you can apply formatting to the domain values if you change this setting from None to one of the available format options which will depend on the domain's data type. For example, you could choose Mon-yyyy for a Date data type, #,##0.00 for a Decimal data type, #,##0 for an Integer data type, or Capitalize for a String data type.
- **Language** This option applies only to String data types. You use it specify the language to apply when you enable the Speller.
- **Enable Speller** You can use this option to allow DQS to check the spelling of values for a domain with a String data type. The Speller will flag suspected syntax, spelling, and sentence structure errors with a red underscore when you are working on the Domain Values or Term-Based Relations tabs of the Domain Management activity, the Manage Domain Values step of the Knowledge Discovery activity, or the Manage And View Results step of the Cleansing activity.
- **Disable Syntax Error Algorithms** DQS can check string values for syntax errors before adding each value to the domain during data cleansing.


Create Domain

Domain Name: Product (Required)

Description: The product domain contains the conventional long (official product name) and short names (model name used to group common products), and product numbers

Data Type: String

☑ Use Leading Values

☑ Normalize String

Format Output to: None

Language: English

☑ Enable Speller

☐ Disable Syntax Error Algorithms

FIGURE 7-3 Domain properties in Create Domain dialog box.

Domain Values After you create the domain, you can add knowledge to the domain by setting domain values. Domain values represent the range of possible values that might exist in a data source for the current domain, including both correct and incorrect values. The inclusion of incorrect domain values in a knowledge base allows you to establish rules for correcting those values during cleansing activities.

You use buttons on the Domain Values tab to add a new domain value manually, import new valid values from Excel, or import new string values with type Correct or Error from a cleansing data quality project.

Tip To import data from Excel, you can use any of the following file types: XLS, XLSX, or CSV. DQS attempts to add every value found in the file, but only if the value does not already exist in the domain. DQS imports values in the first column as domain values and values in other columns as synonyms, setting the value in the first column as the leading value. DQS will not import a value if it violates a domain rule, if the data type does not match that of the domain, or if the value is null.

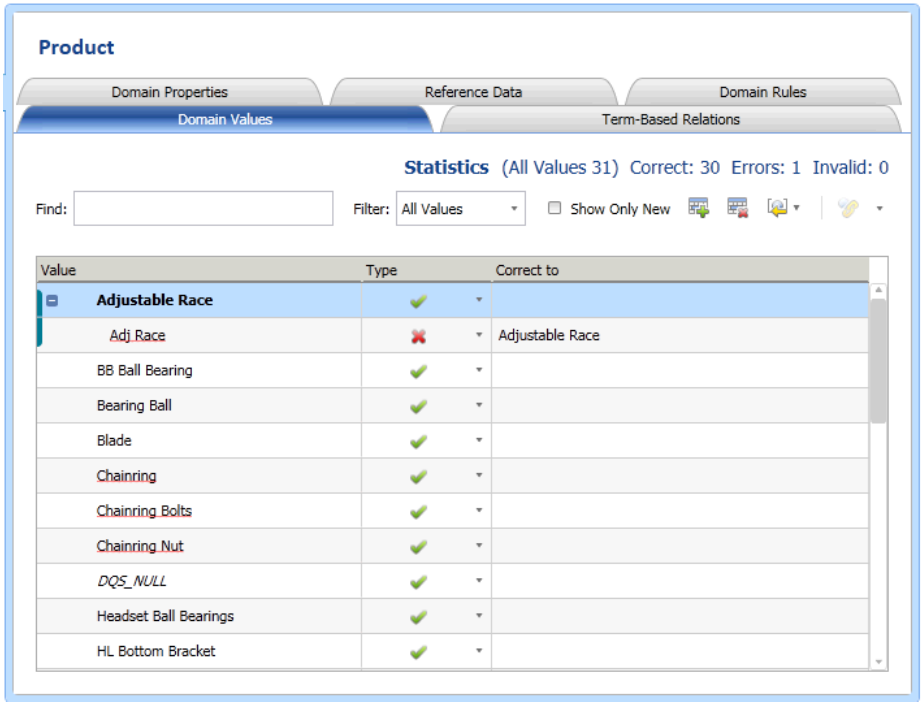
When you add or import values, you can adjust the Type to one of the following settings for each domain value:

- **Correct** You use this setting for domain values that you know is a member of the domain and have no syntax errors.
- **Error** You assign a Type of Error to a domain value that you know is a member of the domain, but has an incorrect value such as a misspelling or undesired abbreviation. For example, in a

Product domain, you might add a domain value of "Mtn 500 Silver 52" with the Error type and include a corrected value of "Mountain-500 Silver, 52." By adding known error conditions to the domain, you can speed up the data cleansing process by having DQS automatically identify and fix known errors which allows you to focus on new errors found during data processing. However, you can flag a domain value as an Error value without providing a corrected value.

- **Invalid** You designate a domain value as Invalid when it is not a member of the domain and you have no correction to associate with it. For example, a value of "United States" would be an invalid value in a Product domain.

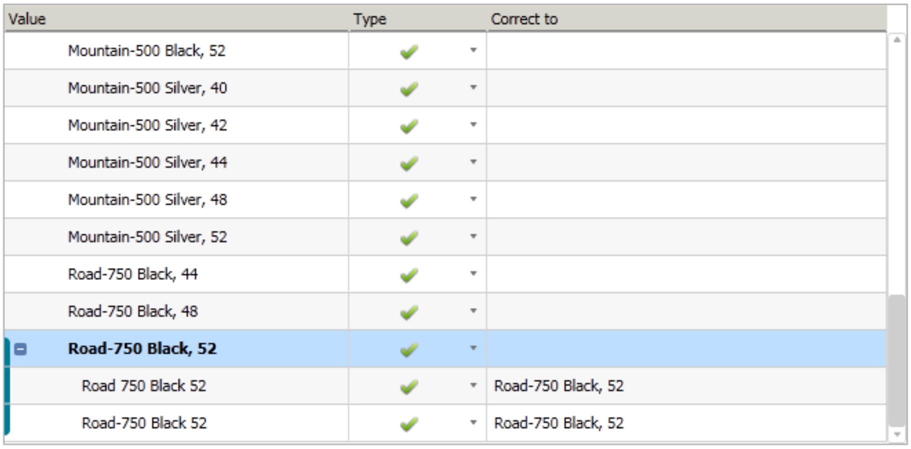
After you publish your domain changes to the knowledge base and later return to the Domain Values tab, you can see the relationship between correct and incorrect values in the domain values list, as shown in Figure 7-4 for the product Adjustable Race. Notice also the underscore in the user interface to identify a potential misspelling of "Adj Race."



Value	Type	Correct to
Adjustable Race	✓	
Adj Race	✗	Adjustable Race
BB Ball Bearing	✓	
Bearing Ball	✓	
Blade	✓	
Chainring	✓	
Chainring Bolts	✓	
Chainring Nut	✓	
DQS_NULL	✓	
Headset Ball Bearings	✓	
HL Bottom Bracket	✓	

FIGURE 7-4 Domain value with Error state and corrected value.

If there are multiple correct domain values that correspond to a single entity, you can organize them as a group of synonyms by selecting them and clicking the Set Selected Domain Vales As Synonyms button in the Domain Values toolbar. Furthermore, you can designate one of the synonyms as a leading value by right-clicking the value and selecting Set As Leading in the context menu. When DQS encounters any of the other synonyms in a cleansing activity, it will replace the non-leading synonym values with the leading value. Figure 7-5 shows an example of synonyms for the leading value Road-750 Black, 52 after the domain changes have been published.

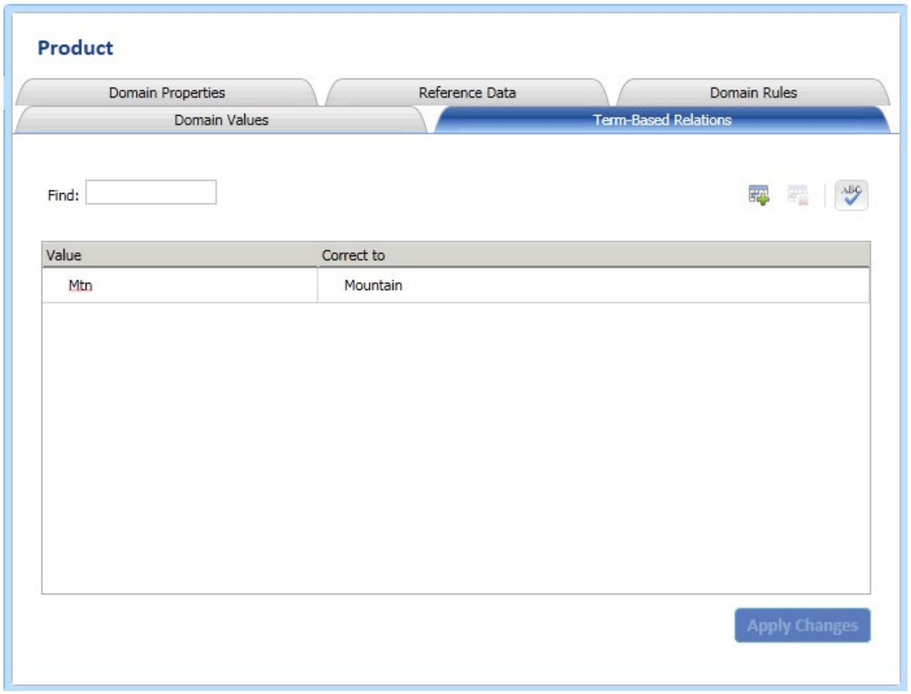


Value	Type	Correct to
Mountain-500 Black, 52	✓	
Mountain-500 Silver, 40	✓	
Mountain-500 Silver, 42	✓	
Mountain-500 Silver, 44	✓	
Mountain-500 Silver, 48	✓	
Mountain-500 Silver, 52	✓	
Road-750 Black, 44	✓	
Road-750 Black, 48	✓	
Road-750 Black, 52	✓	
Road 750 Black 52	✓	Road-750 Black, 52
Road-750 Black 52	✓	Road-750 Black, 52

FIGURE 7-5 Synonym values with designation of a leading value.

Data Quality Client keeps track of the changes you make to domain values during your current session. To review your work, you click the Show/Hide The Domain Values Changes History Panel button, which is accessible by clicking the last button in the Domain Values toolbar.

Term-Based Relations Another option you have for adding knowledge to a domain is term-based relations which you use to make it easier to find and correct common occurrences in your domain values. Rather than set up synonyms for variations of a domain value on the Domain Values tab, you can define a list of string values and specify the corresponding Correct To value on the Term-Based Relations tab. For example, in the product domain, you could have various products that contain the abbreviation Mtn, such as "Mtn-500 Silver, 40" and "Mtn End Caps." To have DQS automatically correct any occurrence of Mtn within a domain value string to Mountain, you can create a term-based relation by specifying a Value/Correct To pair, as shown in Figure 7-6.



Value	Correct to
Mtn	Mountain

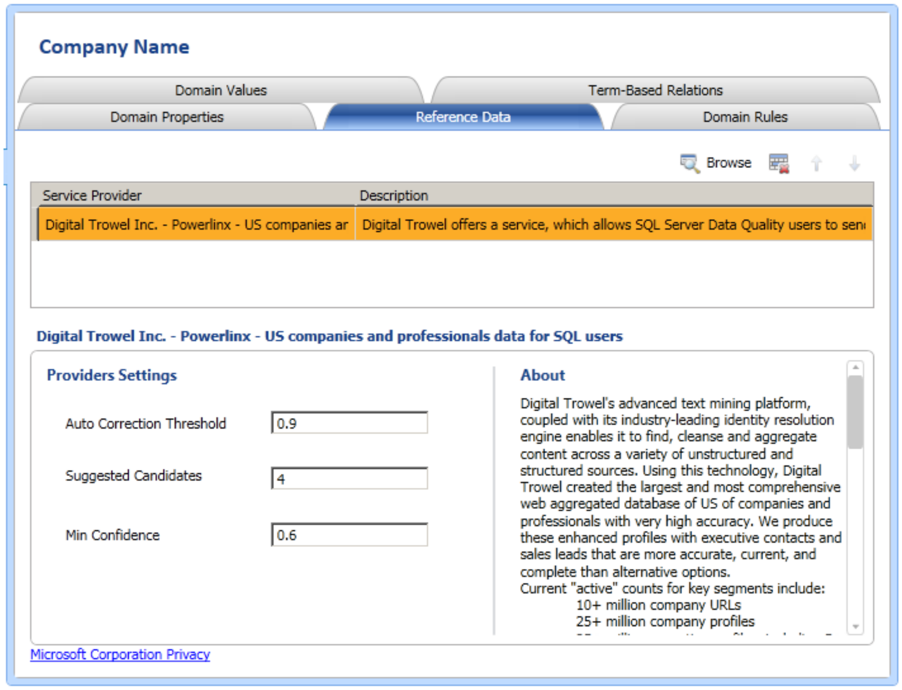
FIGURE 7-6 Term-based relation with a value paired to a corrected value.

Reference Data You can subscribe to a reference data service (RDS) that DQS uses to cleanse, standardize, and enhance your data. Before you can set the properties on the Reference Data tab for your knowledge base, you must configure the RDS provider as described in the "Configuration" section of this chapter.

After you configure your DataMarket Account key in the Configuration area, you click the Browse button on the Reference Data tab to select a reference data service provider for which you have a current subscription. On the Online Reference Data Service Providers Catalog page, you map the domain to an RDS schema column. A letter M displays next to mandatory columns in the RDS schema that you must include in the mapping.

Next, you configure the following settings for the RDS (as shown in Figure 7-7):

- **Auto Correction Threshold** Specify the threshold for the confidence score. During data cleansing, DQS autocorrects records having a score higher than this threshold.
- **Suggested Candidates** Specify the number of candidates for suggested values to retrieve from the reference data service.
- **Min Confidence** Specify the threshold for the confidence score for suggestions. During data cleansing, DQS ignores suggestions with a score lower than this threshold.



Service Provider	Description
Digital Trowel Inc. - Powerlinx - US companies ar	Digital Trowel offers a service, which allows SQL Server Data Quality users to sen

Digital Trowel Inc. - Powerlinx - US companies and professionals data for SQL users

Providers Settings

Auto Correction Threshold	0.9
Suggested Candidates	4
Min Confidence	0.6

About

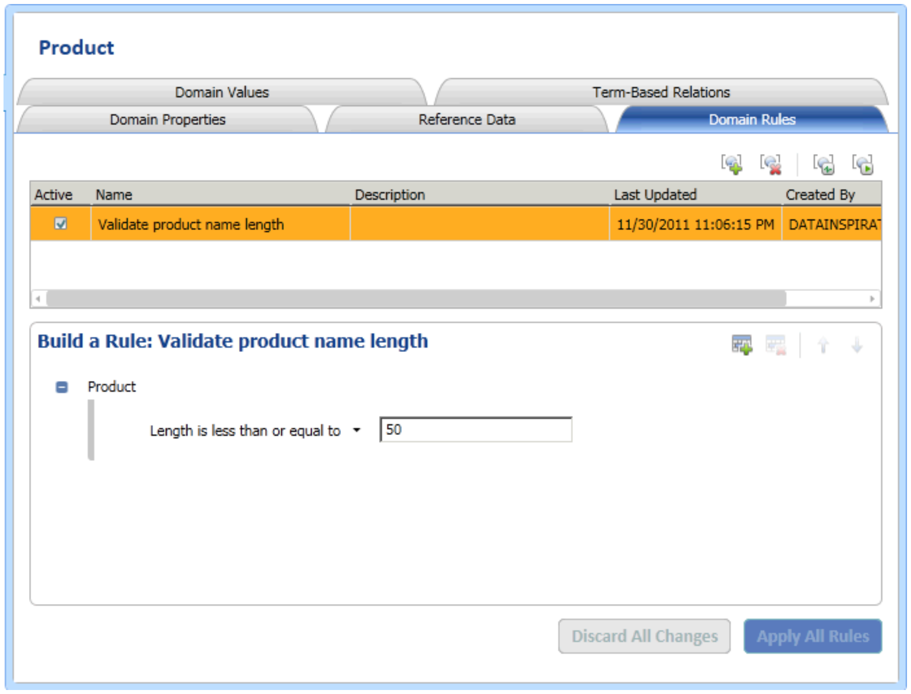
Digital Trowel's advanced text mining platform, coupled with its industry-leading identity resolution engine enables it to find, cleanse and aggregate content across a variety of unstructured and structured sources. Using this technology, Digital Trowel created the largest and most comprehensive web aggregated database of US of companies and professionals with very high accuracy. We produce these enhanced profiles with executive contacts and sales leads that are more accurate, current, and complete than alternative options.
Current "active" counts for key segments include:
10+ million company URLs
25+ million company profiles

Microsoft Corporation Privacy

FIGURE 7-7 Reference data service provider settings.

Domain Rules You use domain rules to establish the conditions that determine whether a domain value is valid. However, it is important to note that domain rules are not used to correct data.

After you click the Add A New Domain Rule button on the Domain Rules tab, you provide a name for the rule and an optional description. Then you define the conditions for the rule in the Build A Rule pane. For example, if there is a maximum length for a domain value, you can create a rule by selecting "Length Is Less Than Or Equal To" in the rule drop-down list and then type in the condition value, as shown in Figure 7-8.



Active	Name	Description	Last Updated	Created By
✓	Validate product name length		11/30/2011 11:06:15 PM	DATAINSPIRAT

Build a Rule: Validate product name length

Product

Length is less than or equal to 50

FIGURE 7-8 Domain rule to validate the length of domain values.

You can create compound rules by adding multiple conditions to the rule and specifying whether the conditions have AND or OR logic. As you build the rule, you can use the Run The Selected Domain Rule On Test Data button. You must manually enter one or more values as test data for this procedure, then click the Test The Domain Rule On All The Terms button. You will see icons display to indicate whether a value is correct, in error, or invalid. Then when you finish building the rule, you click the Apply All Rules button to update the status of domain values according to the new rule.

Note You can temporarily disable a rule by clearing its corresponding Active checkbox.

Composite Domain Sometimes a complex string in your data source contains multiple terms, each of which having different rules or different data types and thus requiring separate domains. However, you still need to validate the field as a whole. For example, with a product name like Mountain-500 Black, 40, you might want to validate the model of Mountain-500, the color Black, and the size 40 separately to confirm the validity of the product name. To address this situation, you can create a composite domain.

Before you can create a composite domain, you must have at least two domains in your knowledge base. Begin the Domain Management activity, and click the Create a Composite Domain button on the toolbar. Type a name for the composite domain, provide a description if you like, and then select the domains to include in the composite domain.

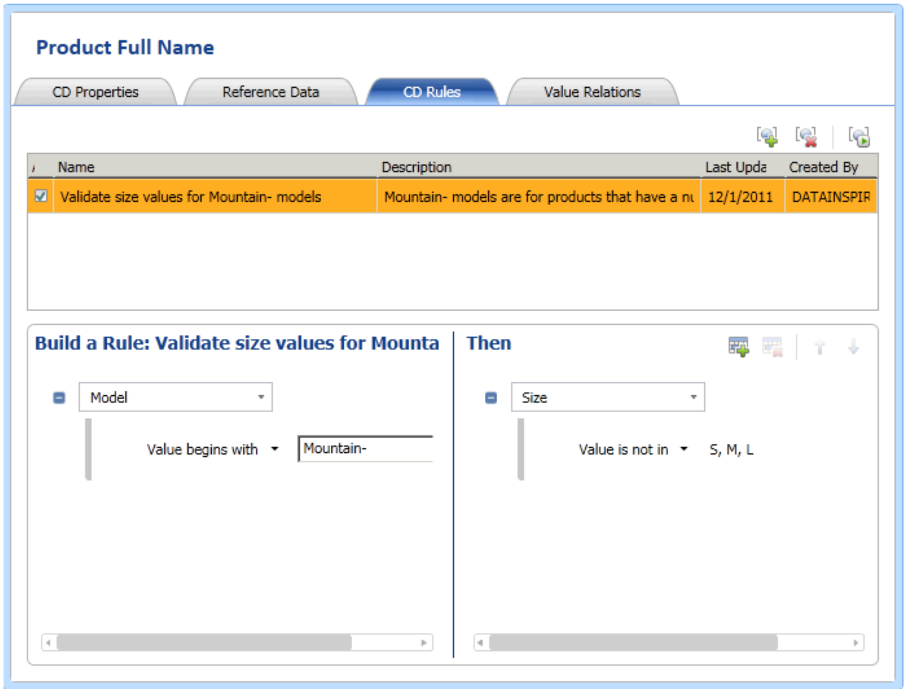
Note After you add a domain to a composite domain, you cannot add it to a second composite domain.

CD Properties You can always change add or remove domains in the composite domain on the CD Properties tab. You can also select one of the following parsing methods in the Advanced section of this tab:

- **Reference Data** If you map the composite domain to an RDS, you can specify that DQS use the RDS for parsing each domain in the composite domain.
- **In Order** You use this setting when you want DQS to parse the field values in the same order that the domains are listed in the composite domain.
- **Delimiters** When your field contains delimiters, you can instruct DQS to parse values based on the delimiter you specify, such as a tab, a comma, or a space. When you use delimiters for parsing, you have the option to use knowledge base parsing. With knowledge base parsing, DQS identifies the domains for known values in the string, and then uses domain knowledge to determine how to add unknown values to other domains. For example, let's say that you have a field in the data source containing the string Mountain-500 Brown, 40. If DQS recognizes Mountain-500 as a value in the Model domain and 40 as a value in the Size domain, but does not recognize Brown in the Color domain, it will add Brown to the Color domain.

Reference Data You use this tab to specify an RDS provider and map the individual domains of a composite domain to separate fields of the provider's RDS schema. For example, you might have company information for which you want to validate address details, and can combine the domains Address, City, State, and Zip as a composite domain that you map to the RDS schema.

CD Rules You use the CD Rules tab to define cross-domain rules for validating, correcting, and standardizing domain values for a composite domain. A cross-domain rule uses similar conditions available to domain rules, but must hold true for all domains in the composite domain rather than for a single domain. Each rule contains an If clause and a Then clause, with each clause containing one or more conditions and applicable to separate domains. For example, you can develop a rule that invalidates a record if the Size value is S, M, or L when the Model value beings with Mountain-, as shown in Figure 7-9. As with domain rules, you can create rules with multiple conditions and you can test a rule before finalizing its definition.


Product Full Name

CD Properties | Reference Data | CD Rules | Value Relations

Name	Description	Last Upda	Created By
✓ Validate size values for Mountain- models	Mountain- models are for products that have a nu	12/1/2011	DATAINSPIR

Build a Rule: Validate size values for Mounta

Model
Value begins with Mountain-

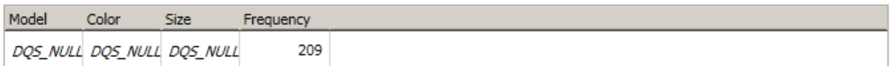
Then

Size
Value is not in S, M, L

FIGURE 7-9 Cross-domain rule to validate corresponding size values in composite domain.

Note If you create a Then clause that uses a definitive condition, DQS will not apply the rule to both domain values and their synonyms. Definitive conditions are Value Is Equal To, Value Is Not Equal To, Value Is In, or Value Is Not In. Furthermore, if you use the Value Is Equal To condition for the Then clause, DQS not only validates data, but also corrects data using this rule.

Value Relations After completing a knowledge discovery activity, you can view the number of occurrences for each combination of values in a composite domain, as shown in Figure 7-10.



Model	Color	Size	Frequency
DQS_NULL	*DQS_NULL*	*DQS_NULL*	209

FIGURE 7-10 Value relations for a composite domain.

Linked Domain You can use a linked domain to handle situations that a regular domain cannot support. For example, if you create a data quality project for a data source that has two fields that use the same domain values, you must set up two domains to complete the mapping of fields to domains. Rather than maintain two domains with the same values, you can create a linked domain that inherits the properties, values, and rules of another domain.

One way to create a linked domain is to open the knowledge base containing the source domain, right-click the domain, and select Create A Linked Domain. Another way to create a linked domain is during an activity that requires you to map fields. You start by mapping the first field to a domain and then attempt to map the second field to the same domain. Data Quality Client will prompt you to create a linked domain, at which time you provide a domain name and description.

After you create the linked domain, you can use the Domain Management activity to complete tasks such as adding domain values or setting up domain rules by accessing either domain. The changes will automatically be made in the other domain. However, you can change the domain properties in the original domain only.

End of Domain Management Activity

DQS locks the knowledge base when you begin the Domain Management activity to prevent others from making conflicting changes. If you cannot complete all the changes that you need to make in a single session, you click the Close button to save your work and keep the knowledge base locked. Use the Finish button when your work is complete. Data Quality Client will display a prompt for you to confirm the action to take. Click Publish to make your changes permanent, unlock the database, and make the knowledge base available to others. Otherwise, click No to save your work, keep the database locked, and exit the Domain Management activity.

Knowledge Discovery

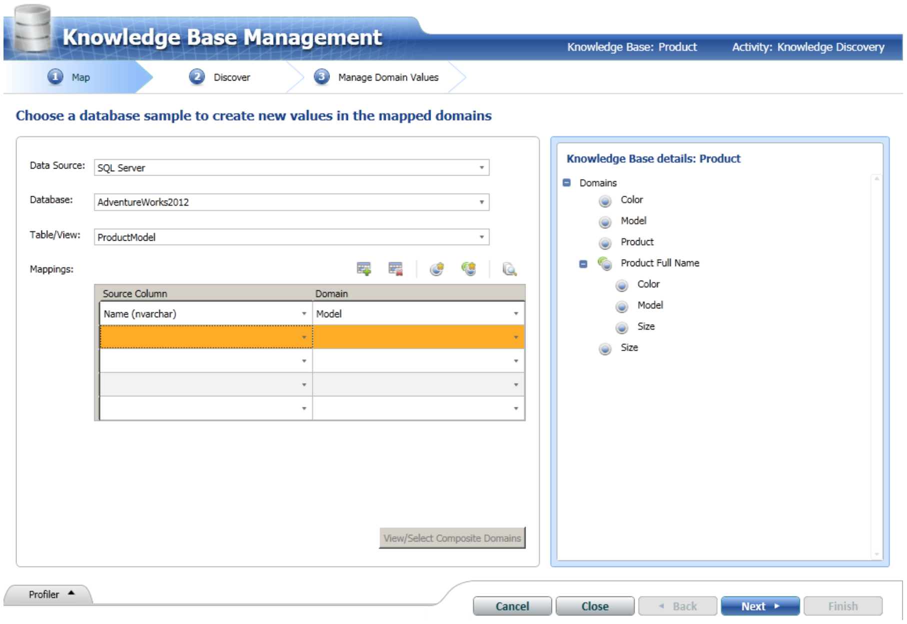
As an alternative to manually adding knowledge to your knowledge base, you can also use the Knowledge Discovery activity to partially automate that process. You can perform this activity multiple times as often as needed to add domain values to the knowledge base from one or more data sources.

To start, you open the knowledge base in the Domain Management area of Data Quality Client, and select the Knowledge Discovery activity. Then you complete a series of three steps: Map, Discover, and Manage Domain Values.

Map

In the Map step, you identify the source data that you want DQS to analyze. This data must be available on the DQS server, either in a SQL Server table or view or in an Excel file. However, but it does not need to be from the same source that you intend to cleanse or de-duplicate with DQS.

The purpose of this step is to map columns in the source data to a domain or composite domain in the knowledge base, as shown in Figure 7-11. You can choose to create a new domain or composite domain at this point when necessary. When you finish mapping all columns, click the Next button to proceed to the Discover step.



Choose a database sample to create new values in the mapped domains

Data Source: SQL Server

Database: AdventureWorks2012

Table/View: ProductModel

Mappings:

Source Column	Domain
Name (nvarchar)	Model

FIGURE 7-11 Source column to domain mapping for the Knowledge Discovery activity.

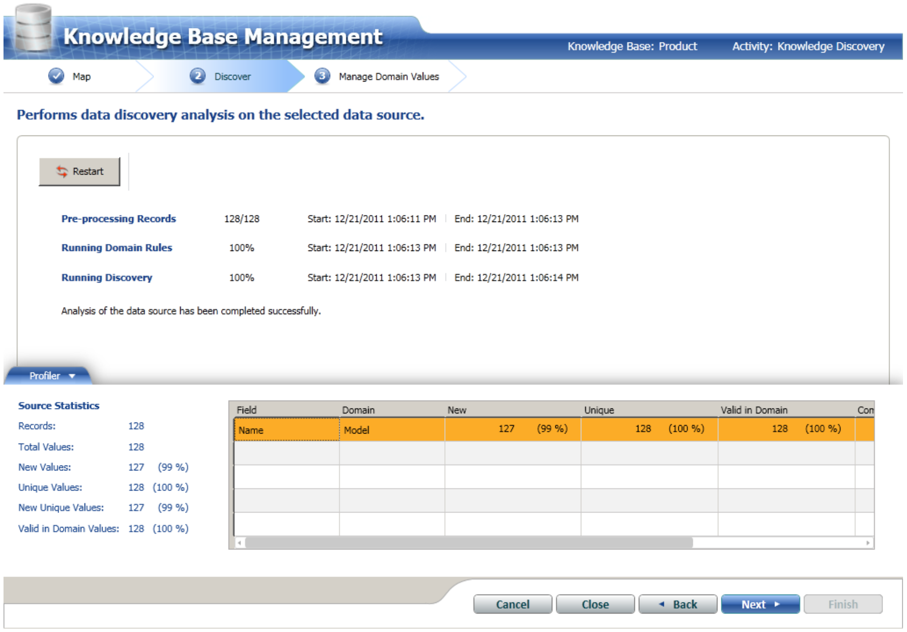
Discover

On the Discover step, you use the Start button to begin the knowledge discovery process. As the process executes, the Discover step displays the current status for the following three phases of processing (as shown in Figure 7-12):

- **Pre-processing Records** During this phase, DQS loads and indexes records from the source in preparation for profiling the data. The status of this phase displays as the number of pre-processed records as compared to the total number of records in the source. In addition, DQS updates all data profiling statistics except the Valid In Domain column. These statistics are visible in the Profiler tab and include total number of records in the source, the total number of values for the domain by field, and the number of unique values by field. DQS also compares the values from the source with the values from the domain to determine which values are found only in the source and identified as new.

- **Running Domain Rules** DQS uses the domain rules for each domain to update the Valid In Domain column in the Profiler, and displays the status as a percentage of completion.

- **Running Discovery** DQS analyzes the data to add to the Manage Domain Values step and identifies syntax errors. As this phase executes, the current status displays as a percentage of completion.



Performs data discovery analysis on the selected data source.

Pre-processing Records	128/128	Start: 12/21/2011 1:06:11 PM	End: 12/21/2011 1:06:13 PM
Running Domain Rules	100%	Start: 12/21/2011 1:06:13 PM	End: 12/21/2011 1:06:13 PM
Running Discovery	100%	Start: 12/21/2011 1:06:13 PM	End: 12/21/2011 1:06:14 PM

Analysis of the data source has been completed successfully.

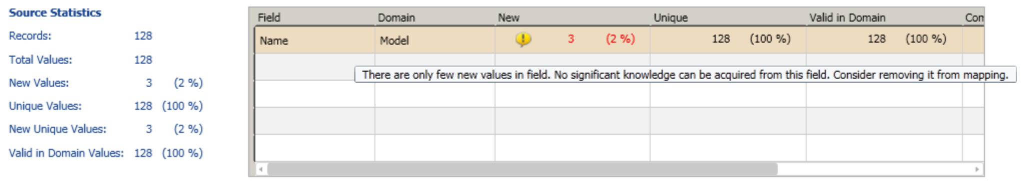
Source Statistics

Records:	128	
Total Values:	128	
New Values:	127	(99 %)
Unique Values:	128	(100 %)
New Unique Values:	127	(99 %)
Valid in Domain Values:	128	(100 %)

Field	Domain	New	Unique	Valid in Domain	Con
Name	Model	127 (99 %)	128 (100 %)	128 (100 %)	

FIGURE 7-12 Source statistics resulting from data discovery analysis.

You use the source statistics on the Profiler tab to assess the completeness and uniqueness of the source data. If the source yields few new values for a domain or has a high number of invalid values, you might consider using a different source. The Profile tab might also display notifications, as shown in Figure 7-13, to alert you to such conditions.



Source Statistics

Records:	128	
Total Values:	128	
New Values:	3	(2 %)
Unique Values:	128	(100 %)
New Unique Values:	3	(2 %)
Valid in Domain Values:	128	(100 %)

Field	Domain	New	Unique	Valid in Domain	Con
Name	Model	⚠ 3 (2 %)	128 (100 %)	128 (100 %)	

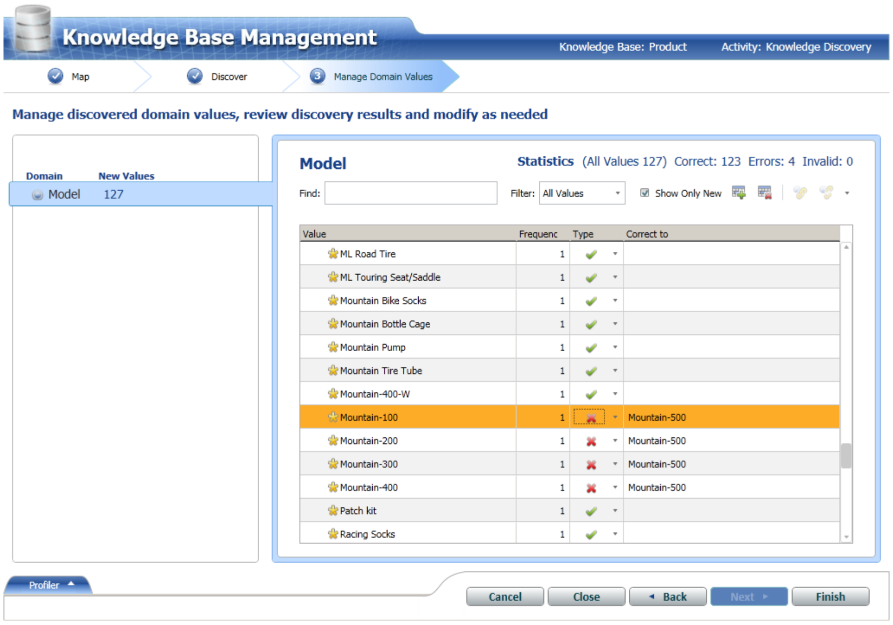
There are only few new values in field. No significant knowledge can be acquired from this field. Consider removing it from mapping.

FIGURE 7-13 Source column to domain mapping for the Knowledge Discovery activity.

Manage Domain Values

On the third step of the Knowledge Discovery activity, you review the results of data discovery analysis. The unique new domain values display in a list with a Type setting and suggested correction where applicable. You can any necessary changes to the values, type, and corrected values, and you can also add new domain values, delete values, and work with synonyms just like you can when working on the Doman Values tab of the Domain Management activity.

Notice in Figure 7-14 that the several product models beginning with “Mountain-” were marked as Error and DQS has proposed a corrected value of “Mountain-500” for each of the new domain values because “Mountain-500” was the only pre-existing product model in the domain. DQS determined that similar product models found in the source must be misspellings and proposed corrections to the source values to match them to the pre-existing domain value. In this scenario, if the new product models are all correct, you can change the Type setting to Correct. When you make this change, Data Quality Client automatically removes the Correct To value.



Manage discovered domain values, review discovery results and modify as needed

Domain	New Values
Model	127

Model — **Statistics** (All Values 127) Correct: 123 Errors: 4 Invalid: 0

Find: Filter: All Values ☑ Show Only New

Value	Frequenc	Type	Correct to
ML Road Tire	1	✓	
ML Touring Seat/Saddle	1	✓	
Mountain Bike Socks	1	✓	
Mountain Bottle Cage	1	✓	
Mountain Pump	1	✓	
Mountain Tire Tube	1	✓	
Mountain-400-W	1	✓	
Mountain-100	1	✗	Mountain-500
Mountain-200	1	✗	Mountain-500
Mountain-300	1	✗	Mountain-500
Mountain-400	1	✗	Mountain-500
Patch kit	1	✓	
Racing Socks	1	✓	

FIGURE 7-14 Source column to domain mapping for the Knowledge Discovery activity.

After reviewing each domain and making corrections, click the Finish button to end the Knowledge Discovery activity. Then you click the Publish button to complete the activity, update the knowledge base with the new domain values, and leave the knowledge base in an unlocked state. If you click the No button instead of the Publish button, the results of the activity are discarded and the knowledge base is unlocked.

Matching Policy

Another aspect of adding knowledge to a knowledge base is defining a matching policy. This policy is necessary for data quality projects that use matching to correct data problems such as misspelled customer names or inconsistent address formats. A matching policy contains one or more matching rules that DQS uses to determine the probability of a match between two records.

You begin by opening a knowledge base in the Domain Management area of Data Quality Client and selecting the Matching Policy activity. The process to create a matching policy is consists of three steps: Map, Matching Policy, and Matching Results.

Tip You might consider creating a knowledge base that you use only for matching projects. In the matching-only knowledge base, you include only domains that have values that are both discrete and uniquely identify a record, such as names and addresses.

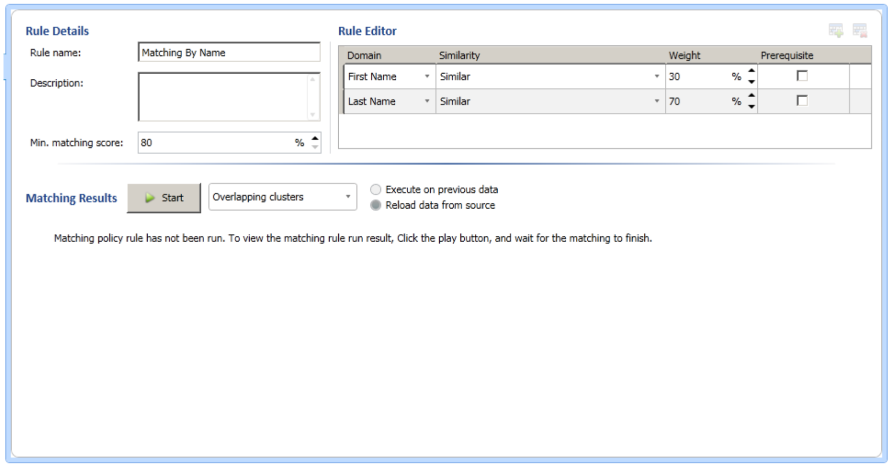
Map

The first step of the Matching Policy activity is similar to the first step of the Knowledge Discovery activity. You start the creation of a matching policy by mapping a field from an Excel or SQL Server data source to a domain or composite domain in the selected knowledge base. If a corresponding domain does not exist in the knowledge base, you have the option to create one. You must select a source field in this step if you want to reference that field in a matching rule in the next step. Use the Next button to continue to the Discover step.

Matching Policy

In the Matching Policy step, you set up one or more matching rules that DQS uses to assign a matching score for each pair of records that it compares. DQS considers the records as a match when this matching score is greater than the minimum matching score that you establish for the matching policy.

To begin, you click the Create A Matching Rule button. Next, you assign a name, an optional description, and a minimum matching score to the matching rule. The lowest minimum matching score that you can assign is 80 percent, unless you change the DQS configuration on the Administration page. In the Rule Editor toolbar, you click the Add A New Domain Element button, select a domain, and configure the matching rule parameters for the selected domain, as shown in Figure 7-15.



Rule name: Matching By Name
Description:
Min. matching score: 80 %

Rule Editor

Domain	Similarity	Weight	Prerequisite
First Name	Similar	30 %	☐
Last Name	Similar	70 %	☐

Overlapping clusters
○ Execute on previous data
● Reload data from source

Matching policy rule has not been run. To view the matching rule run result, Click the play button, and wait for the matching to finish.

FIGURE 7-15 Creation of a matching rule for a matching policy.

You can choose from the following values when configuring the Similarity parameter:

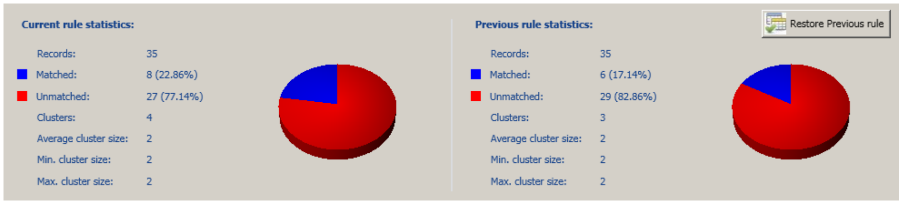
- **Similar** You select this value when you want DQS to calculate a matching score for a field in two records and set the similarity score to 0 (to indicate no similarity) when the matching score is less than 60. If the field has a numeric data type, you can set a threshold for similarity using a percentage or integer value. If the field has a date data type, you can set the threshold using a numeric value for day, month, or year.

- **Exact** When you want DQS to identify two records to be a match only when the same field in each record is identical, you select this value. DQS will assign a matching score of 100 for the domain when the fields are identical. Otherwise, it assigns a matching score of 0.

Whether you add one or more domains to a matching rule, you must configure the Weight parameter for each domain that you do not set as a prerequisite. DQS uses the weight to determine how the individual domain's matching score affects the overall matching score. The sum of the weight values must be equal to 100.

When you select the Prerequisite check box for a domain, DQS sets the Similarity parameter to Exact and considers values in a field to be a match only when they are identical in the two compared records. Regardless of the result, a prerequisite domain has no effect on the overall matching score for a record. Using the prerequisite option is an optimization that speeds up the matching process.

You can test the rule by clicking the Start button on the Matching Policy page. If the results are not what you expect, you can modify the rule and test the rule again. When you re-test the matching policy, you can choose to either execute the matching policy on the processed matches from a previous execution or on data that DQS reloads from the source. The Matching Results tab, shown in Figure 7-16, displays the results of the current test and the previous test so that you can determine whether your changes improve the match results.


Current rule statistics:

	Records:	35
🟦	Matched:	8 (22.86%)
🟥	Unmatched:	27 (77.14%)
	Clusters:	4
	Average cluster size:	2
	Min. cluster size:	2
	Max. cluster size:	2

Previous rule statistics:

	Records:	35
🟦	Matched:	6 (17.14%)
🟥	Unmatched:	29 (82.86%)
	Clusters:	3
	Average cluster size:	2
	Min. cluster size:	2
	Max. cluster size:	2

FIGURE 7-16 Comparison of results from consecutive executions of a matching rule.

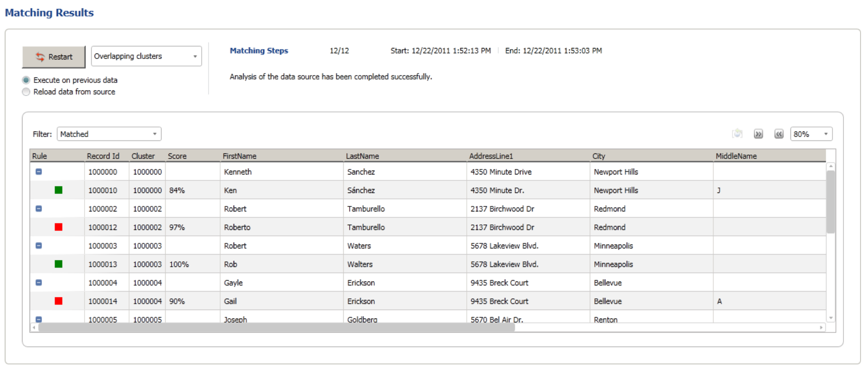
A review of the Profiler tab can help you decide how to modify a match rule. For example, if a field has a high percentage of unique records, you might consider eliminating the field from a match rule or lower the weight value. On the other hand, having a low percentage of unique records is useful only if the field has a high level of completeness. If both uniqueness and completeness are low, you should exclude the field from the matching policy.

You can use the Restore Previous Rule button to revert the rule settings if you prefer. When you are satisfied with the results, click the Next button to continue to the next step.

Matching Results

On the Matching Results step, you choose whether to review results as overlapping clusters or non overlapping clusters. With overlapping clusters, you might see separate clusters that contain the same records whereas with non overlapping clusters you see only clusters with records in common. Then you click the Start button to apply all matching rules to your data source.

When processing completes, you can view a table that displays a filtered list of matched records with a color code to indicate the applicable matching rule, as shown in Figure 7-17. Each cluster of records has a pivot record that DQS randomly selects from the cluster as the record to keep. Furthermore, each cluster includes one or more matched records along with its matching score. You can change the filter to display unmatched records or you can apply a separate filter to view matched records having scores greater than or equal to 80, 85, 90, 95, or 100 percent respectively.



12/12 Start: 12/22/2011 1:52:13 PM | End: 12/22/2011 1:53:03 PM

Analysis of the data source has been completed successfully.

[● Execute on previous data]
○ Reload data from source

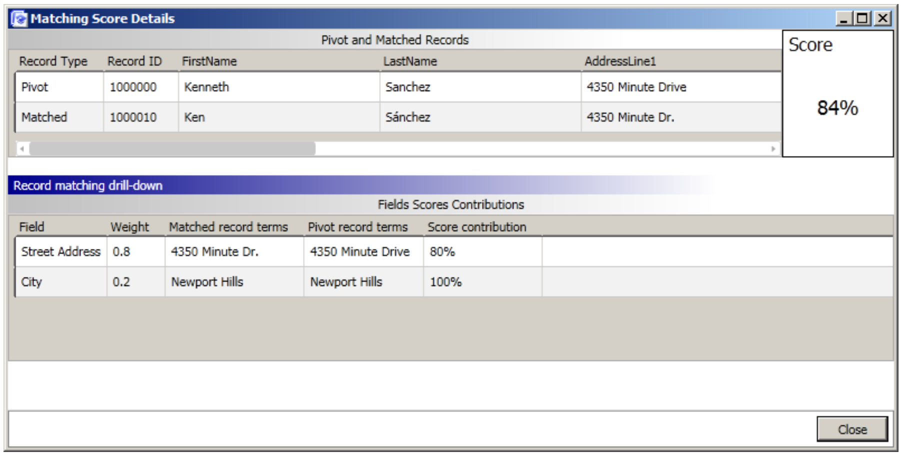
Filter: Matched

80%

Rule	Record Id	Cluster	Score	FirstName	LastName	AddressLine1	City	MiddleName
⊟	1000000	1000000		Kenneth	Sanchez	4350 Minute Drive	Newport Hills	
🟩	1000010	1000000	84%	Ken	Sánchez	4350 Minute Dr.	Newport Hills	J
⊟	1000002	1000002		Robert	Tamburello	2137 Birchwood Dr	Redmond	
🟥	1000012	1000002	97%	Roberto	Tamburello	2137 Birchwood Dr	Redmond	
⊟	1000003	1000003		Robert	Waters	5678 Lakeview Blvd.	Minneapolis	
🟩	1000013	1000003	100%	Rob	Walters	5678 Lakeview Blvd.	Minneapolis	
⊟	1000004	1000004		Gayle	Erickson	9435 Breck Court	Bellevue	
🟥	1000014	1000004	90%	Gail	Erickson	9435 Breck Court	Bellevue	A
⊟	1000005	1000005		Joseph	Goldberg	5670 Bel Air Dr.	Renton	

FIGURE 7-17 Matched records based on two matching rules.

You can review the color codes on the Matching Rules tab, and evaluate statistics about the matching results on the Matching Results tab. You can also double-click a matched record in the list to see the pivot and matched record fields side-by-side, the score for each field, and the overall score for the match, as shown in Figure 7-18.



Pivot and Matched Records

Record Type	Record ID	FirstName	LastName	AddressLine1
Pivot	1000000	Kenneth	Sanchez	4350 Minute Drive
Matched	1000010	Ken	Sánchez	4350 Minute Dr.

Score

84%

Record matching drill-down

Fields Scores Contributions

Field	Weight	Matched record terms	Pivot record terms	Score contribution
Street Address	0.8	4350 Minute Dr.	4350 Minute Drive	80%
City	0.2	Newport Hills	Newport Hills	100%

FIGURE 7-18 Matching score details displaying field scores and an overall score.

If necessary, you can return to the previous step to fine-tune a matching rule and then return to this step. Before you click the Restart button, you can choose the Reload Data From Source option to copy the source data into a staging table where DQS re-indexes it. Your other option is to choose Execute On Previous Data to use the data in the staging table without re-indexing it, which could process the matching policy more quickly.

Once you are satisfied with the results, you can click the Finish button and then you have the option to publish the matching policy to the knowledge base. You can now use the matching policy with a matching data quality project.

Data Quality Projects

One you have a knowledge base in place, you can create a data quality project to use the knowledge it contains to cleanse source data or use its matching policy to find matching records in source data. After you run the data quality project, you can export its results to a SQL Server database or to a CSV file. As another option, you can import the results to a domain in the Domain Management activity.

Regardless of which type of data quality project that you want to create, you start the project in the same way by clicking the New Data Quality Project button on the home screen of Data Quality Client. When you create a new project, you provide a name for the data quality project, an optional description, and select a knowledge base. You can then select the Cleansing activity for any knowledge base, but you can select the Matching activity for a knowledge base only when that knowledge base has a matching policy. To launch the activity's wizard, you click the Create button. At this point, the project is locked and inaccessible to other users.

Cleansing Projects

A cleansing data quality project begins with an analysis of source data using knowledge contained in a knowledge base and a categorization of that data into groups of correct and incorrect data. After DQS completes the analysis and categorization process, you can approve, reject, or change the proposed corrections.

When you create a cleansing data quality project, the wizard leads you through four steps: Map, Cleanse, Manage And View Results, and Export.

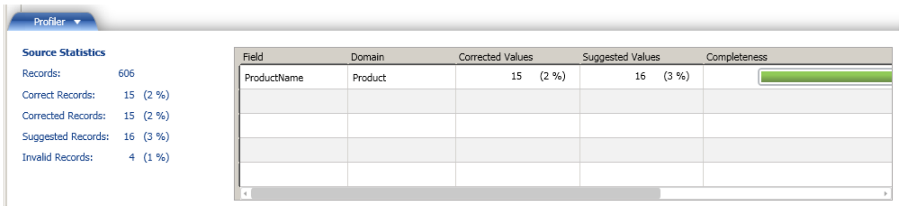
Map

The first step in the cleansing data quality project is to map the data source to domains in the selected knowledge base, following the same process that you use for the Knowledge Discovery or Matching Policy activities. The data source can be either an Excel file or a table or view in a SQL Server database. When you finish mapping all columns, click the Next button to proceed to the Cleanse step.

Note If you map a field to a composite domain, only the rules associated with the composite domain will apply rather than the rules for the individual domains assigned to the composite domain. Furthermore, if the composite domain is mapped to a reference data service, DQS sends the source data to the reference data service for parsing and cleansing. Otherwise, DQS performs the parsing using the method that you specified for the composite domain.

Cleanse

To begin the cleansing process, click the Start button. When the analysis process completes, you can view the statistics in the Profiler tab, as shown in Figure 7-19. These statistics reflect the results of categorization that DQS performs: correct records, corrected records, suggested records, and invalid records. DQS uses advanced algorithms to cleanse data and calculates a confidence score to determine the category applicable to each record in the source data. You can configure confidence thresholds for auto-correction and auto-suggestion. If a record's confidence score falls below either of these thresholds and is neither correct nor invalid, DQS categorizes the record as new and leaves it for you to manually correct if necessary in the next step.



Source Statistics

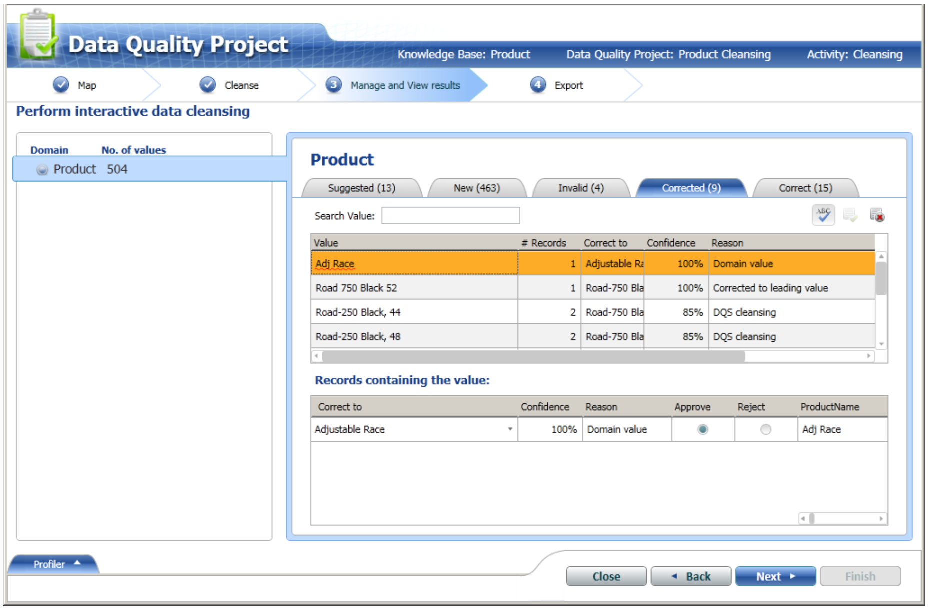
Records:	606
Correct Records:	15 (2 %)
Corrected Records:	15 (2 %)
Suggested Records:	16 (3 %)
Invalid Records:	4 (1 %)

Field	Domain	Corrected Values	Suggested Values	Completeness
ProductName	Product	15 (2 %)	16 (3 %)	🟩

FIGURE 7-19 Profiler statistics after the cleansing process completes.

Manage and View Results

On the next step of the cleansing data quality project, you see separate tabs for each group of records categorized by DQS: Suggested, New, Invalid, Corrected, and Correct. The tab labels show the number of records allocated to each group. When you open a tab, you can see a table of domain values for that group and the number of records containing each domain value, as shown in Figure 7-20.



Domain	No. of values
Product	504

Value	# Records	Correct to	Confidence	Reason
Adj Race	1	Adjustable Ra	100%	Domain value
Road 750 Black 52	1	Road-750 Bla	100%	Corrected to leading value
Road-250 Black, 44	2	Road-750 Bla	85%	DQS cleansing
Road-250 Black, 48	2	Road-750 Bla	85%	DQS cleansing

Correct to	Confidence	Reason	Approve	Reject	ProductName
Adjustable Race	100%	Domain value	●	○	Adj Race

FIGURE 7-20 Categorized results after automated data cleansing.

When applicable, you can also see the proposed corrected value, the confidence score, and the reason for the proposed correction for each value. If you select a row in the table, you can see the individual records that contain the original value. You must use the horizontal scroll bar to see all the fields for the individual records.

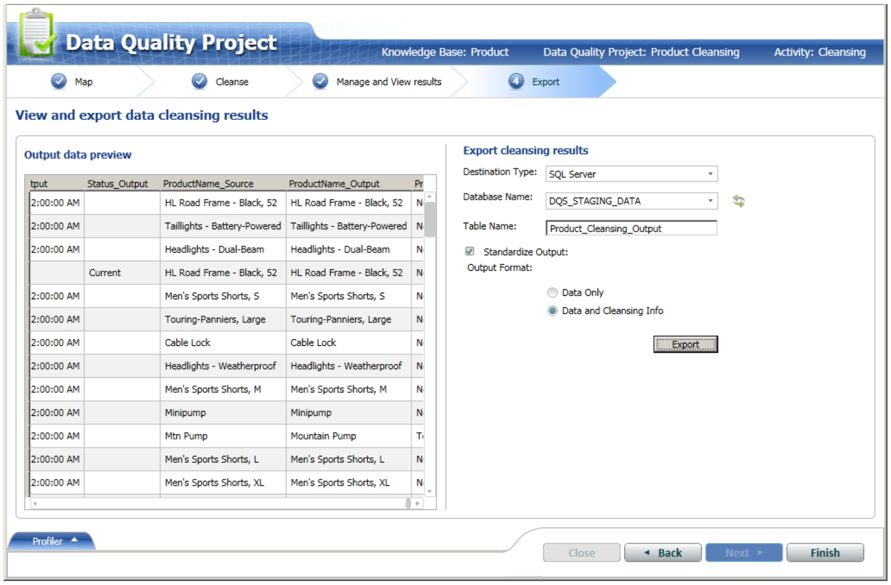
When you enable the Speller feature for a domain, the cleansing process will identify potential spelling errors by displaying a wavy red underscore below the domain value. You can right-click the value to see suggestions and select one or add the potential error to the dictionary.

If DQS identifies a value in the source data as a synonym, it suggests a correction to leading value. This feature is useful for standardization of your data. You must first enable the domain for leading values and define synonyms in the knowledge base before running a cleansing data quality project.

After reviewing the proposed corrections, you can either approve or reject the change for each value or for each record individually. Another option is to use the Approve All Terms button or the Reject All Terms button in the toolbar. You can also replace a proposed correction by typing a new value in the Correct To box, and then approve the manual correction. In most cases, approved values move to the Corrected tab and rejected values move to the Invalid tab. However, if you approve a value on the New tab, it moves to the Correct tab.

Export

At no time during the cleansing process does DQS change source data. Instead, you can export the results of the cleansing data quality project to a SQL Server table or to a CSV file. A preview of the output displays in the final step of the project, as shown in Figure 7-21.



View and export data cleansing results

Output data preview

tput	Status_Output	ProductName_Source	ProductName_Output	Pr
2:00:00 AM		HL Road Frame - Black, 52	HL Road Frame - Black, 52	N
2:00:00 AM		Taillights - Battery-Powered	Taillights - Battery-Powered	N
2:00:00 AM		Headlights - Dual-Beam	Headlights - Dual-Beam	N
	Current	HL Road Frame - Black, 52	HL Road Frame - Black, 52	N
2:00:00 AM		Men's Sports Shorts, S	Men's Sports Shorts, S	N
2:00:00 AM		Touring-Panniers, Large	Touring-Panniers, Large	N
2:00:00 AM		Cable Lock	Cable Lock	N
2:00:00 AM		Headlights - Weatherproof	Headlights - Weatherproof	N
2:00:00 AM		Men's Sports Shorts, M	Men's Sports Shorts, M	N
2:00:00 AM		Minipump	Minipump	N
2:00:00 AM		Mtn Pump	Mountain Pump	T
2:00:00 AM		Men's Sports Shorts, L	Men's Sports Shorts, L	N
2:00:00 AM		Men's Sports Shorts, XL	Men's Sports Shorts, XL	N

Export cleansing results

Destination Type: SQL Server
Database Name: DQS_STAGING_DATA
Table Name: Product_Cleansing_Output
☑ Standardize Output:
Output Format:
○ Data Only
● Data and Cleansing Info
Export

FIGURE 7-21 Review of the cleansing results to export.

Before you export the data, you must decide whether you want to export the data only or both data and cleansing information. Cleansing information includes the original value, the cleansed value, the reason for a correction, a confidence score, and the categorization of the record. By default, DQS will use the output format for the domain as defined in the knowledge base unless you clear the Standardize Output check box. When you click the Export button, Data Quality Client exports the data to the specified destination. You can then click the Finish button to close and unlock the data quality project.

Matching Projects

By using a matching policy defined for a knowledge base, a matching data quality project can identify both exact and approximate matches in a data source. Ideally, you run the matching process after running the cleansing process and exporting the results. You can then specify the export file or destination table as the source for the matching project.

A matching data quality project consists of three steps: Map, Matching, and Export.

Map

The first step for a matching project begins in the same way as a cleansing project by requiring you to map fields from the source to a domain. However, in a matching project, you must map a field to each domain specified in the knowledge base's matching policy.

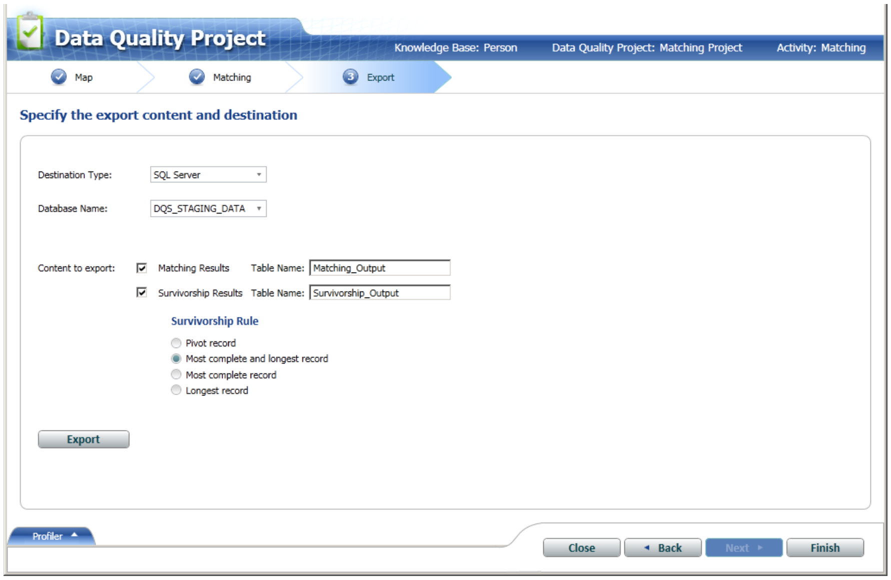
Matching

On the Matching step, you choose whether to generate overlapping clusters or nonoverlapping clusters, and then click the Start button to launch the automated matching process. When the process completes, you can review a table of matching results by cluster. The interface is similar in functionality to the one you use when reviewing the matching results during the creation of a matching policy. However, in the matching project, an additional column includes a checkbox that you use to reject a record as a match.

Export

After reviewing the matching results, you can export the results as the final step of the matching project. As shown in Figure 7-22, you must choose the destination and the content to export. You have the following two options for export content:

- **Matching Results** This content type includes both matched and unmatched records. The matched records include several columns related to the matching process, including the cluster identifier, the matching rule that identified the match, the matching score, the approval status, and a flag to indicate the pivot record.
- **Survivorship Results** This content type includes only the survivorship record and unmatched records. You must select a survivorship rule when choosing this export option to specify which of the matched records in a cluster is preserved in the export. All other records in a cluster are discarded. If more than one record satisfies the survivorship criteria, DQS keeps the record with the lowest record identifier.



Specify the export content and destination

Destination Type: SQL Server

Database Name: DQS_STAGING_DATA

Content to export: [✓] Matching Results Table Name: Matching_Output

[✓] Survivorship Results Table Name: Survivorship_Output

Survivorship Rule

- ○ Pivot record
- ● Most complete and longest record
- ○ Most complete record
- ○ Longest record

FIGURE 7-22 Selection of content to export.

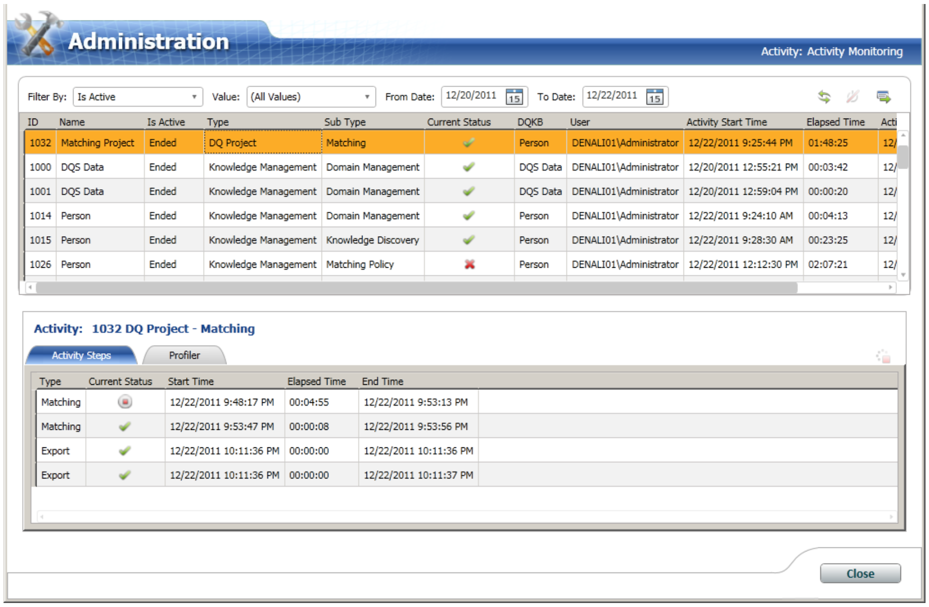
Important When you click the Finish button, the project is unlocked and available for later use. However, DQS uses the knowledge base contents at the time that you finished the project and ignores any subsequent changes to the knowledge base. To access any changes to the knowledge base, such as a modified matching policy, you must create a new matching project.

Administration

In the Administration feature of Data Quality Client, you can perform Activity Monitoring and Configuration tasks. Activity monitoring is accessible by any user that can open Data Quality Client. On the other hand, only an administrator can access the configuration tasks.

Activity Monitoring

You can use the Activity Monitoring page to review the status of current and historic activities performed on the DQS server, as shown in Figure 7-23. DQS administrators can terminate an activity or a step within an activity when necessary by right-clicking on the activity or step.



Filter By: Is Active Value: (All Values) From Date: 12/20/2011 To Date: 12/22/2011

ID	Name	Is Active	Type	Sub Type	Current Status	DQKB	User	Activity Start Time	Elapsed Time	Acti
1032	Matching Project	Ended	DQ Project	Matching	✓	Person	DENALI01\Administrator	12/22/2011 9:25:44 PM	01:48:25	12/
1000	DQS Data	Ended	Knowledge Management	Domain Management	✓	DQS Data	DENALI01\Administrator	12/20/2011 12:55:21 PM	00:03:42	12/
1001	DQS Data	Ended	Knowledge Management	Domain Management	✓	DQS Data	DENALI01\Administrator	12/20/2011 12:59:04 PM	00:00:20	12/
1014	Person	Ended	Knowledge Management	Domain Management	✓	Person	DENALI01\Administrator	12/22/2011 9:24:10 AM	00:04:13	12/
1015	Person	Ended	Knowledge Management	Knowledge Discovery	✓	Person	DENALI01\Administrator	12/22/2011 9:28:30 AM	00:23:25	12/
1026	Person	Ended	Knowledge Management	Matching Policy	✗	Person	DENALI01\Administrator	12/22/2011 12:12:30 PM	02:07:21	12/

Activity: 1032 DQ Project - Matching

Type	Current Status	Start Time	Elapsed Time	End Time
Matching	●	12/22/2011 9:48:17 PM	00:04:55	12/22/2011 9:53:13 PM
Matching	✓	12/22/2011 9:53:47 PM	00:00:08	12/22/2011 9:53:56 PM
Export	✓	12/22/2011 10:11:36 PM	00:00:00	12/22/2011 10:11:36 PM
Export	✓	12/22/2011 10:11:36 PM	00:00:00	12/22/2011 10:11:37 PM

FIGURE 7-23 Status of activities and status of activity steps of the selected activity.

Activities that appear on this page include knowledge discovery, domain management, matching policy, cleansing projects, matching projects, and the cleansing transformation in an Integration Services package. You can see who initiated each activity, the start and end time of the activity, and the elapsed time. To facilitate locating specific activities, you can use a filter to find activities by date range and by status, type, subtype, knowledge base, or user. When you select an activity on this page, you can view the related activity details such as the steps and profiler information.

You can use the Export The Selected Activity To Excel button to export the activity details, process steps, and profiling information to Excel. The export file separates this information into four worksheets:

- **Activity** This sheet includes the details about the activity, including the name, type, subtype, current status, elapsed time, and so on.
- **Processes** This sheet includes information about each activity step, including current status, start and end time, and elapsed time.
- **Profiler – Source** The contents of this sheet depend on the activity sub type. For the Cleansing sub type, you see the number of total records, correct records, corrected records, and invalid records. For the Knowledge Discovery, Domain Management, Matching Policy, and Matching sub types, you see the number of records, total values, new values, unique value, and new unique values.
- **Profiler – Fields** This sheet's contents also depend on the activity sub type. For the Cleansing

and SSIS Cleansing sub types, the sheet contains the following information by field: domain, corrected values, suggested values, completeness, and accuracy. For the Knowledge Discovery, Domain Management, Matching Policy, and Matching sub types, the sheet contains the following information by field: domain, new value count, unique value count, count of values that are valid in the domain, and completeness.

Configuration

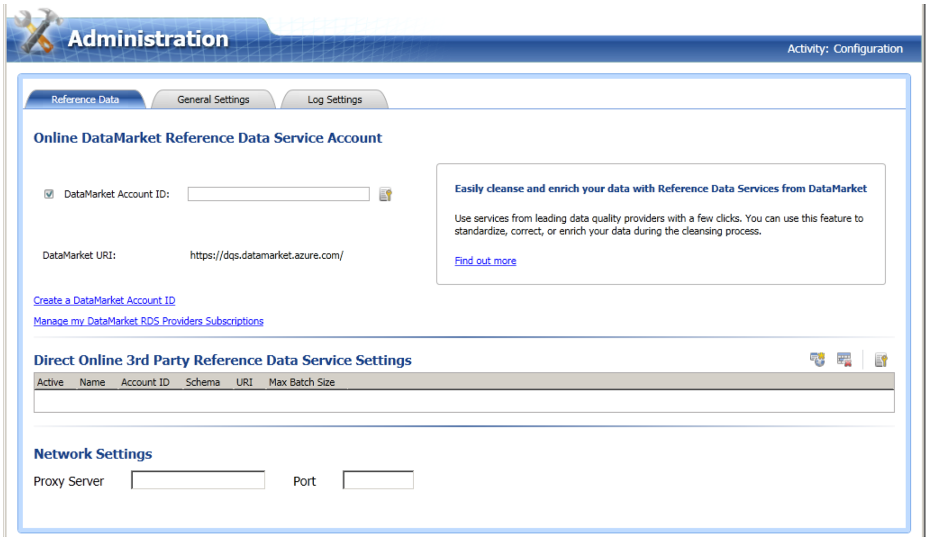
The Configuration area of Data Quality Client allows you to set up reference data providers, set properties for the DQS server, and configure logging. You must be a DQS administrator to perform configuration tasks. You access this area from the Data Quality Client home screen by clicking the Configuration button.

Reference Data

Rather than maintain domain values and rules in a knowledge base, you can subscribe to a reference data service through Windows Azure Marketplace. Most reference data services are available as a monthly paid subscription, but some providers offer a free trial and Digital Trowel Inc. provides a free service to cleanse and standardize data for US public and private companies. When you subscribe to a service, you receive an account key that must register in Data Quality Client before you can use reference data in your data quality activities.

The Reference Data tab is the first tab that displays in the Configuration area, as shown in Figure 7-24. Here you type or paste your account key in the DataMarket Account ID box, and then click the Validate DataMarket Account ID button to the right of the box. You might need to provide a Proxy Server and port number if your DQS Server requires a proxy server to connect to the Internet.

Note If you do not have a reference data service subscription, you can use the Create A DataMarket Account ID link to open the Windows Azure Marketplace site in your browser. You must have a Windows Live ID to access site. Click the Data link at the top of the page, and then click the Data Quality Services link in the Category list. You can view the current list of reference data service providers at https://datamarket.azure.com/browse/Data?Category=dqs.



Online DataMarket Reference Data Service Account

DataMarket Account ID:

DataMarket URI: https://dqs.datamarket.azure.com/

Easily cleanse and enrich your data with Reference Data Services from DataMarket

Use services from leading data quality providers with a few clicks. You can use this feature to standardize, correct, or enrich your data during the cleansing process.

Find out more

Create a DataMarket Account ID

Manage my DataMarket RDS Providers Subscriptions

Direct Online 3rd Party Reference Data Service Settings

Active	Name	Account ID	Schema	URI	Max Batch Size

Network Settings

Proxy Server Port

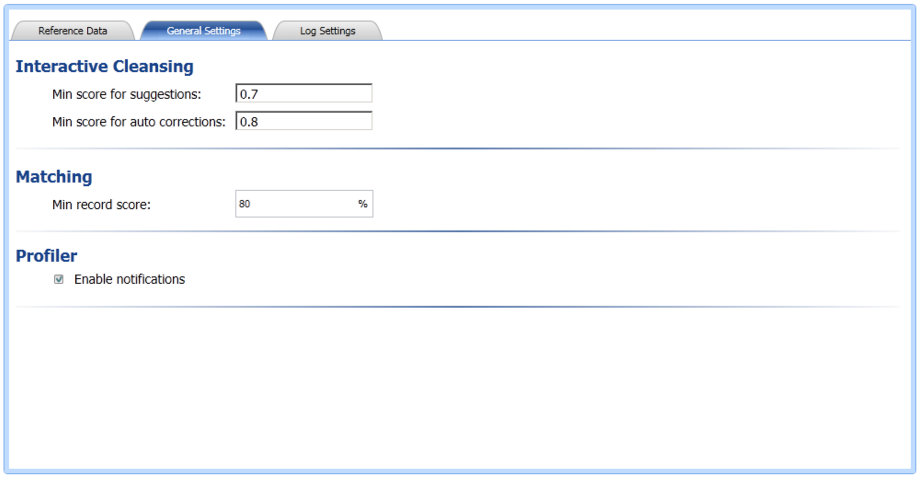
FIGURE 7-24 Reference data service account configuration.

As an alternative to using a DataMarket subscription for reference data, you can configure settings for a third-party reference data service by clicking the Add New Reference Data Service Provider button and supplying the requisite details: a name for the service, a comma-delimited list of fields as a schema, a secure URI for the reference data service, a maximum number of records per batch, and a subscriber account identifier.

General Settings

You use the General Settings tab, shown in Figure 7-25, to configure the following settings:

- **Interactive Cleansing** Specify the minimum confidence score for suggestions and the minimum confidence score for auto corrections. DQS uses these values as thresholds when determining how to categorize records for a cleansing data quality project.
- **Matching** Specify the minimum matching score for DQS to use for a matching policy.
- **Profiler** Use this checkbox to enable or disable profiling notifications. These notifications appear in the Profiler tab when you are performing a knowledge base activity or running a data quality project.



Interactive Cleansing

Min score for suggestions: 0.7

Min score for auto corrections: 0.8

Matching

Min record score: 80 %

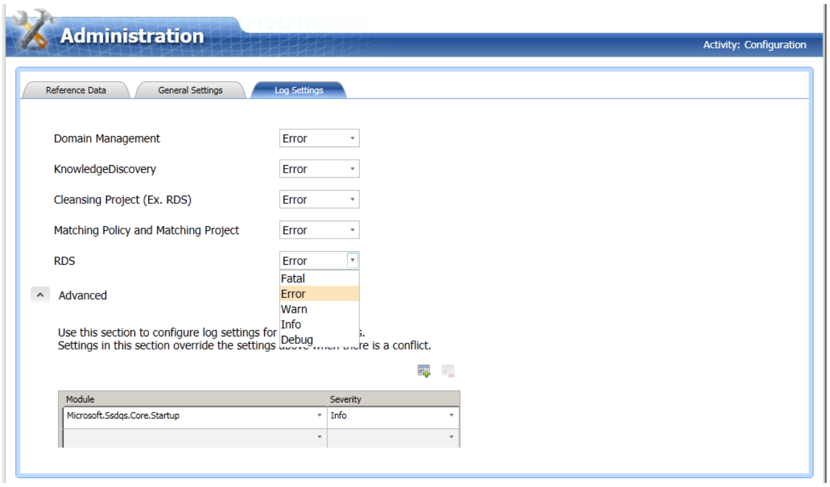
Profiler

☑ Enable notifications

FIGURE 7-25 General settings to set score thresholds and enable notifications.

Log Settings

Log files are useful for troubleshooting problems that might occur. By default, the DQS log files capture events with an Error severity level, but you can change the severity level to Fatal, Warn, Info, or Debug by activity, as shown in Figure 7-26.



Domain Management: Error

KnowledgeDiscovery: Error

Cleansing Project (Ex. RDS): Error

Matching Policy and Matching Project: Error

RDS: Error

Use this section to configure log settings for s.
Settings in this section override the settings re is a conflict.

Module	Severity
Microsoft.Ssdqs.Core.Startup	Info

FIGURE 7-26 Log settings for DQS server.

DQS generates the following three types of log files:

- **DQS Server** You can find server-related activity in the DQServerLog.DQS_MAIN.log file in the Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\Log folder.

- **Data Quality Client** You can view client-related activity in the DQClientLog.log file in the %APPDATA%\SSDQS\Log folder.
- **DQS Cleansing Transformation** When you execute a package containing the DQS Cleansing transformation, DQS logs the cleansing activity in the DQSSSISLog.log file available in the %APPDATA%\SSDQS\Log folder.

In addition to configure log severity settings by activity, you can also configure them at the module level in the Advanced section of the Log Settings tab. By using a more granular approach to log settings, you can get better insight into a problem that you are troubleshooting. The Microsoft.Ssdqs.Core.Startup is configured with a default severity of Info to track events related to starting and stopping the DQS service. You can use the drop-down list in the Advanced section to select another module and specify the log severity level that you want.

Integration

DQS cleansing and matching functionality is built into two other SQL Server 2012 features, Integration Services and Master Data Services, so that you can more effectively manage data quality across your organization. In Integration Services, you can use DQS components to routinely perform data cleansing in a scheduled package. In Master Data Services, you can compare external data to master data to find matching records based on the matching policy that you define for a knowledge base.

Integration Services

In earlier versions of SQL Server, you could use an Integration Services package to automate the process of cleansing data by using Derived Column or Script transformations, but the creation of a data flow to perform complex cleansing could be tedious. Now you can take advantage of DQS to use the rules or reference data in a knowledge base for data cleansing. The integration between Integration Services and DQS allows you to perform the same tasks that a cleansing data quality project supports, but on a scheduled basis. Another advantage of using the DQS Cleansing transformation in Integration Services is the ability to cleanse data from a source other than Excel or a SQL Server database.

Because the DQS functionality in Integration Services is built into the product, you can begin using it right away without additional installation or configuration. Of course, you must have both a DQS server and a knowledge base available. To get started, you add a DQS connection manager to the package, add a Data Flow Task, and then add a DQS Cleansing transformation to the data flow.

DQS Connection Manager

You use the DQS Connection Manager to establish a connection from the Integration Services package to a DQS server. When you add the connection manager to your package, you supply the server name. The connection manager interface includes a button to test the connection to the DQS server. You can identify a DQS connection manager by its icon, as shown in Figure 7-27.

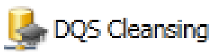


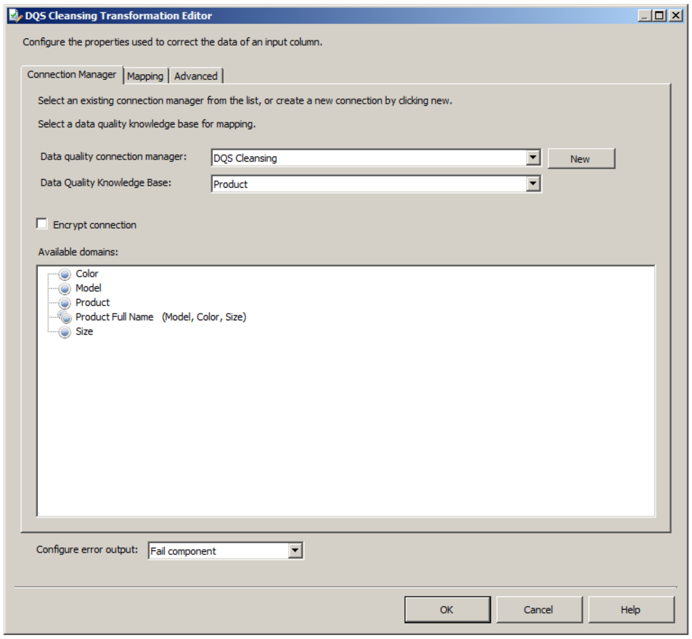
FIGURE 7-27 DQS Connection Manager.

DQS Cleansing Transformation

As we explain in the "Cleansing Projects" section of this chapter, DQS uses advanced algorithms to cleanse data and calculates confidence score to categorize records as Correct, Corrected, Suggested, or Invalid. The DQS Cleansing transformation in a data flow transfers data to the DQS server, which in turn executes the cleansing process and sends the data back to the transformation with a corrected value, when applicable, and a status. You can then add a Conditional Split transformation to the data flow to route each record to a separate destination based on its status.

Note If the knowledge base maps to a reference data service, the DQS server might also forward the data to the service for cleansing and enhancement.

Configuration of the DQS Cleansing transformation begins with selection of connection manager and a knowledge base. After you select the knowledge base, the available domains and composite domains display, as shown in Figure 7-28.


DQS Cleansing Transformation Editor

Configure the properties used to correct the data of an input column.

Connection Manager | Mapping | Advanced

Select an existing connection manager from the list, or create a new connection by clicking new.

Select a data quality knowledge base for mapping.

Data quality connection manager: DQS Cleansing

Data Quality Knowledge Base: Product

Encrypt connection

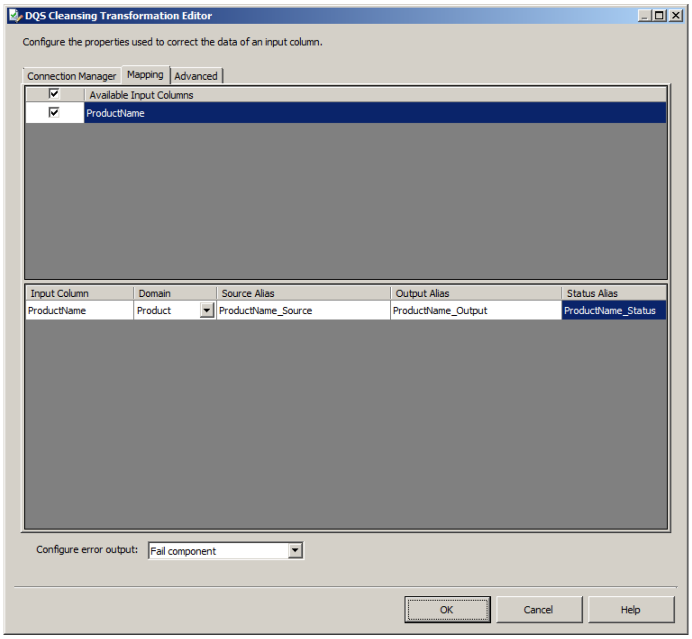
Available domains:

- Color
- Model
- Product
- Product Full Name (Model, Color, Size)
- Size

Configure error output: Fail component

FIGURE 7-28 Connection manager configuration for the DQS Cleansing transformation.

On the Mapping tab, you map each column in the data flow pipeline to its respective domain, as shown in Figure 7-29. If you are mapping a column to a composite domain, the column must contain the domain values as a comma-delimited string in the same order in which the individual domains appear in the composite domain.


DQS Cleansing Transformation Editor

Configure the properties used to correct the data of an input column.

☑	Available Input Columns
☑	ProductName

Input Column	Domain	Source Alias	Output Alias	Status Alias
ProductName	Product	ProductName_Source	ProductName_Output	ProductName_Status

Configure error output: Fail component

FIGURE 7-29 Mapping a pipeline column to a domain.

For each input column, you also define the aliases for output columns: source, output, and status. The transformation editor supplies default values for you, but you can change these aliases if you like. During package execution, the column with the source alias contains the original value in the source whereas the column with the output alias contains the same value for correct and invalid records or the corrected value for suggested or corrected records. The column with the status alias contains values to indicate the outcome of the cleansing process: Auto Suggest, Correct, Invalid, or New.

On the Advanced tab of the transformation editor, you can configure the following options:

- **Standardize Output** This option, which is enabled by default, automatically standardizes the data in the output alias column according to the Format Output settings for each domain. In addition, this option will change synonyms to leading values if you enable Use Leading Values for the domain.

- **Enable Field-Level Columns** You can optionally include the confidence score or the reason for a correction as additional columns in the transformation output.

- **Enable Record-Level Columns** If a domain maps to a reference data service that returns additional data columns during the cleansing process, you can include this data as an appended column. In addition, you can include a column to contain the schema for the appended data.

Master Data Services

If you are using Master Data Services (MDS) for master data management, you can use the data matching functionality in DQS to de-duplicate master data. You must first enable DQS integration on the Web Configuration page of Master Data Services Configuration Manager and you must create a matching policy for a DQS knowledge base. Then you add data to an Excel worksheet and use the MDS Add-in for Excel to combine that data with MDS-managed data in preparation for matching. The matching process adds columns to the worksheet similar to the columns you view during a matching data quality project, including the matching score. We provide more information about how to use the DQS matching with MDS in Chapter 8, "Master Data Services."

Chapter 8

Master Data Services

The first release of Master Data Services (MDS) appeared in SQL Server 2008 R2 to support master data management. In the current release, you find improvements in some areas of the user interface and a new feature for managing your master data, the MDS Add-in for Excel. In addition, there are deprecated features and discontinued features that change the way that you work with MDS. Collectively, these changes to MDS simplify the implementation, workflows and administration of MDS.

Getting Started

MDS is available as a feature in SQL Server 2012 Setup rather than as a separate installer as it was in SQL Server 2008. If you have an existing SQL Server 2008 MDS installation, you must decide whether to upgrade MDS with or without a database engine upgrade. In this section, we explain the considerations for each option. In addition, we describe the post-installation configuration steps to perform whether you have a new or upgraded MDS installation.

Upgrade Considerations

When you upgrade an existing MDS implementation, you can keep the MDS database in an SQL Server 2008 R2 database instance or you can migrate it to a SQL Server 2012 database instance. Regardless of which choice you make, you should backup your MDS database before you start the upgrade process.

If you choose not to upgrade the MDS database, you must install SQL Server 2012 side-by-side with SQL Server 2008 R2, although the two versions of SQL Server do not have to be on the same computer. Furthermore, when you install SQL Server 2012, you need only to install the MDS feature, which adds files to the Program Files\Microsoft SQL Server \110\Master Data Services. You must then use Master Data Services Configuration Manager to upgrade the MDS database. It will continue to reside in the SQL Server 2008 R2 instance, but the upgrade process modifies the schema of the MDS database to support the new features of MDS in SQL Server 2012.

On the other hand, if you choose to upgrade the database engine to SQL Server 2012, you must first uninstall MDS by using the Uninstall command in the Programs and Features area of Control Panel. Then you use the SQL Server 2012 Setup wizard to perform the upgrade. After starting setup, choose Installation, then select Upgrade from SQL Server 2005, SQL Server 2008 or SQL Server 2008 R2 and complete the wizard. Then use the SQL Server 2012 Setup wizard again to add the MDS feature to your existing installation.

Configuration

Whether you perform a new installation of MDS or upgrade from a previous version, you must use the Master Data Services Configuration Manager. You can open this tool from the Master Data Services folder in the Microsoft SQL Server 2012 program group on the Start menu. You use it to create or upgrade the MDS database, configure MDS system settings, and create a web application for MDS.

Important If you are upgrading from a previous version of MDS, you must log in using the Administrator account that was used to create the original MDS database. You can identify this account by finding the user with an ID value of 1 in the mdm.tblUser table in the MDS database.

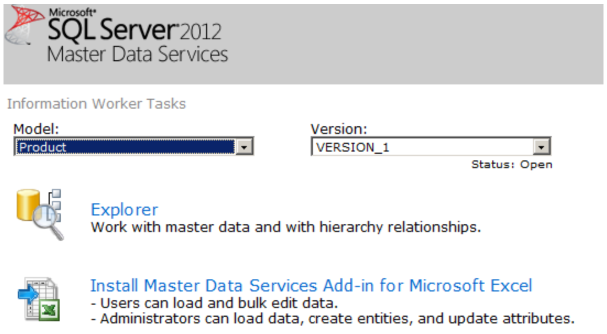
The first configuration step to perform following installation is to configure the MDS database. On the Database Configuration page of Master Data Services Configuration Manager, perform one of the following tasks:

- New installation   Click the Create Database button, and complete the Database Wizard. In the wizard, you specify the SQL Server instance and authentication type and provide credentials having permissions to create a database on the selected instance. You also provide a database name and specify collation. Last, you specify a Windows account to establish as the MDS administrator account.
- Upgrade   If you want to keep your database in a SQL Server 2008 R2 instance, click the Repair Database button if it is enabled. Then, whether you want to store your MDS database in SQL Server 2008 R2 or SQL Server 2012, click the Upgrade Database button. The Upgrade Database Wizard displays the SQL Server instance and MDS database name, and the progress of the update. The upgrade process recreates tables using the new schema and stored procedures.

Note The upgrade process excludes business rules that you use to generate values for the code attribute. We explain the new automatic code generation in the "Entity" section later in this chapter. Furthermore, the upgrade process does not include model deployment packages. You must create new packages in your SQL Server 2012 installation.

When the new or upgraded database is created, you see the System Settings display on the Database Configuration page. You use these settings to control timeouts for the database or the web service, to name a few. If you upgraded your MDS installation, there are two new system settings that you can configure:

- Show Add-in For Excel Text On Website Home Page   This setting controls whether users see a link to install the MDS Add-in on the MDS home page, as shown in Figure 8-1.
- Add-in For Excel Install Path On Website Home Page   This setting defaults to the MDS Add-in download page on the Microsoft web site.



Information Worker Tasks

Model: Product

Version: VERSION_1

Status: Open

Explorer
Work with master data and with hierarchy relationships.

Install Master Data Services Add-in for Microsoft Excel
- Users can load and bulk edit data.
- Administrators can load data, create entities, and update attributes.

FIGURE 8-1 Add-in for Excel text on MDS Website home page.

Master Data Manager

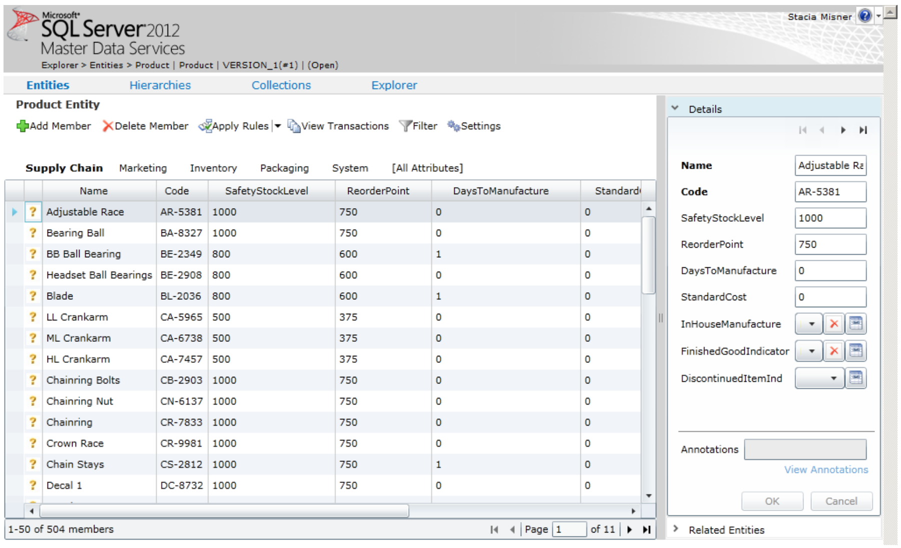
Master Data Manager is the Web application that data stewards use to manage master data and administrators use to manage model objects and configure security. For the most part, it works as it did in the previous version of MDS, although the Explorer and Integration Management functional areas now use a Silverlight 5.0. As a result, you will find certain tasks are easier and faster to perform. In keeping with this goal of enabling easier and faster processes, you will find the current version of MDS also introduces a new staging process and slight changes to the security model.

Explorer

In the Explorer functional area, you add or delete members for an entity, update attribute values for members, arrange those members within a hierarchy, and optionally organize groups of members as collections. In SQL Server 2012, MDS improves the workflow for performing these tasks.

Entity Management

When you open an entity in the Explorer area, you see a new interface, as shown in Figure 8-2. The set of buttons now display with new icons and with descriptions that clarify their purpose. When you click the Add Member button, you type in all attribute values for the new member in the Details pane. To delete a member, select the member and then click the Delete button. You can also more easily edit attribute values for a member by selecting it in the grid and then typing a new value for the attribute in the Details pane.



Explorer > Entities > Product | Product | VERSION_1(#1) | (Open)

Product Entity

Name	Code	SafetyStockLevel	ReorderPoint	DaysToManufacture	Standard
Adjustable Race	AR-5381	1000	750	0	0
Bearing Ball	BA-8327	1000	750	0	0
BB Ball Bearing	BE-2349	800	600	1	0
Headset Ball Bearings	BE-2908	800	600	0	0
Blade	BL-2036	800	600	1	0
LL Crankarm	CA-5965	500	375	0	0
ML Crankarm	CA-6738	500	375	0	0
HL Crankarm	CA-7457	500	375	0	0
Chainring Bolts	CB-2903	1000	750	0	0
Chainring Nut	CN-6137	1000	750	0	0
Chainring	CR-7833	1000	750	0	0
Crown Race	CR-9981	1000	750	0	0
Chain Stays	CS-2812	1000	750	1	0
Decal 1	DC-8732	1000	750	0	0

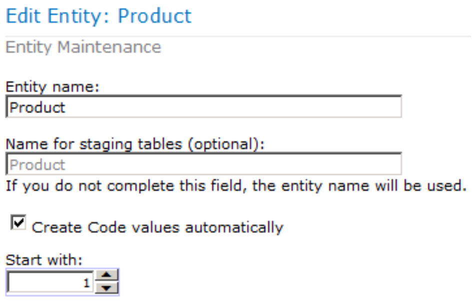
1-50 of 504 members　　Page 1 of 11

Details

Name	Adjustable Ra
Code	AR-5381
SafetyStockLevel	1000
ReorderPoint	750
DaysToManufacture	0
StandardCost	0
InHouseManufacture	
FinishedGoodIndicator	
DiscontinuedItemInd	
Annotations	

FIGURE 8-2 Member management for a selected entity.

Rather than use a business rule to automatically create values for the Code attribute as you do in SQL Server 2008 R2, you can now configure the entity to automatically generate the code value. This automatic assignment of a value applies whether you are adding a member manually in Master Data Manager, or importing data through the staging process. To do this, you must have permission to access the System Administration function area and open the entity. On the Entity Maintenance page, select the Create Code Values Automatically check box, as shown in Figure 8-3. You can optionally change the number in the Start With box. If you already have members added to the entity, MDS increments the maximum value by one when you add a new member.



Edit Entity: Product

Entity Maintenance

Entity name:
Product

Name for staging tables (optional):
Product
If you do not complete this field, the entity name will be used.

☑ Create Code values automatically

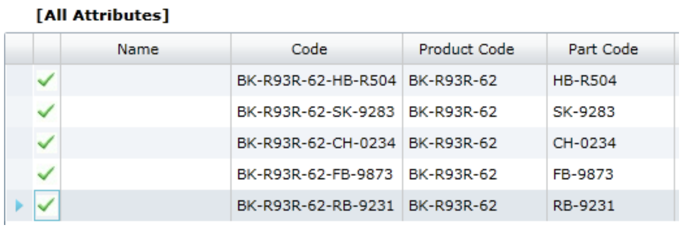
Start with:
1

FIGURE 8-3 Check box to automatically generate Code value.

> **Note** The automatic generation of a value occurs only when you leave the Code value blank. You always have the option to override it with a different value when you add a new member.

Many-to-Many Mapping

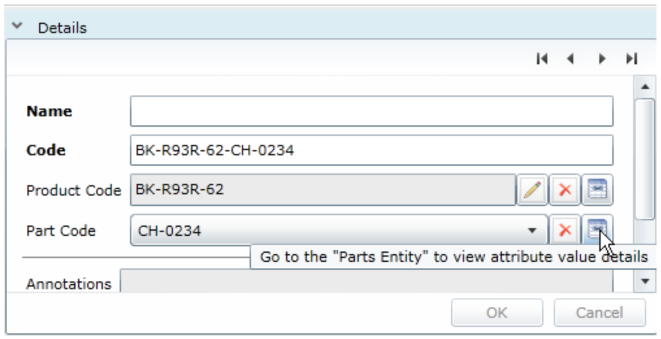
Another improvement in the current version of MDS is the ability to use the Explorer functional area in Master Data Manager to view entities for which you have defined many-to-many mappings. To see how this works, consider a scenario in which you have products that you want to decompose into separate parts and you can associate any single part with multiple products. You manage the relationships between products and parts in MDS by creating three entities: products, parts, and a mapping entity to define the relationship between products and parts, as shown in Figure 8-4. Notice that you need only a code value and attributes to store code values for the two related entities, but no name value.


[All Attributes]

		Name	Code	Product Code	Part Code
	✓		BK-R93R-62-HB-R504	BK-R93R-62	HB-R504
	✓		BK-R93R-62-SK-9283	BK-R93R-62	SK-9283
	✓		BK-R93R-62-CH-0234	BK-R93R-62	CH-0234
	✓		BK-R93R-62-FB-9873	BK-R93R-62	FB-9873
▸	✓		BK-R93R-62-RB-9231	BK-R93R-62	RB-9231

FIGURE 8-4 Arrangement of members in a derived hierarchy.

When you open the Product entity and select a member, you can open the Related Entities pane on the right side of the screen where a link displays, such as ProductParts (Attribute: Product Code). When you click this link, a new browser window opens to display the ProductParts entity window with a filter applied to show records related only to the selected product. From that screen, you can click the Go To The "<Name> Entity" To View Attribute Details button, as shown in Figure 8-5, to open yet another browser window that displays the entity member and its attribute values.


Details

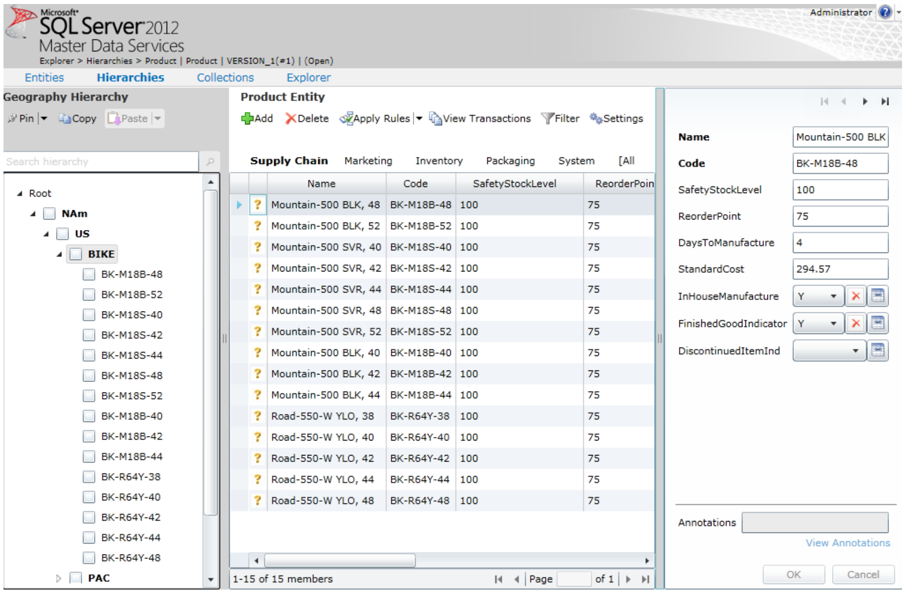
Name	
Code	BK-R93R-62-CH-0234
Product Code	BK-R93R-62
Part Code	CH-0234
Annotations	

Go to the "Parts Entity" to view attribute value details

FIGURE 8-5 Click through to related entity available in Details pane.

Hierarchy Management

The new interface in the Explorer functional area, shown in Figure 8-5, makes it easier for you to move members within a hierarchy when you want to change the parent for a member. A hierarchy pane displays a tree view of the hierarchy. There you select the check box for each member to move. Then you click the Copy button at the top of the hierarchy pane, select the check box of the member to which you want to move the previously selected members, and then click the Paste button. In a derived hierarchy, you must paste members to the same level only.



- Root
  - NAm
    - US
      - BIKE
        - BK-M18B-48
        - BK-M18B-52
        - BK-M18S-40
        - BK-M18S-42
        - BK-M18S-44
        - BK-M18S-48
        - BK-M18S-52
        - BK-M18B-40
        - BK-M18B-42
        - BK-M18B-44
        - BK-R64Y-38
        - BK-R64Y-40
        - BK-R64Y-42
        - BK-R64Y-44
        - BK-R64Y-48
      - PAC

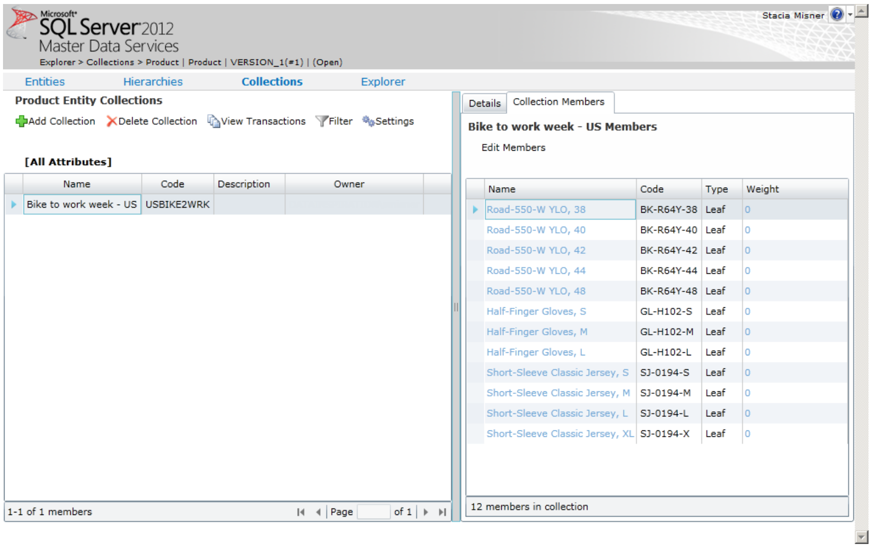
Name	Code	SafetyStockLevel	ReorderPoin
Mountain-500 BLK, 48	BK-M18B-48	100	75
Mountain-500 BLK, 52	BK-M18B-52	100	75
Mountain-500 SVR, 40	BK-M18S-40	100	75
Mountain-500 SVR, 42	BK-M18S-42	100	75
Mountain-500 SVR, 44	BK-M18S-44	100	75
Mountain-500 SVR, 48	BK-M18S-48	100	75
Mountain-500 SVR, 52	BK-M18S-52	100	75
Mountain-500 BLK, 40	BK-M18B-40	100	75
Mountain-500 BLK, 42	BK-M18B-42	100	75
Mountain-500 BLK, 44	BK-M18B-44	100	75
Road-550-W YLO, 38	BK-R64Y-38	100	75
Road-550-W YLO, 40	BK-R64Y-40	100	75
Road-550-W YLO, 42	BK-R64Y-42	100	75
Road-550-W YLO, 44	BK-R64Y-44	100	75
Road-550-W YLO, 48	BK-R64Y-48	100	75

Name	Mountain-500 BLK
Code	BK-M18B-48
SafetyStockLevel	100
ReorderPoint	75
DaysToManufacture	4
StandardCost	294.57
InHouseManufacture	Y
FinishedGoodIndicator	Y
DiscontinuedItemInd	

FIGURE 8-6 Arrangement of members in a derived hierarchy.

Collection Management

You can organize a subset of entity members as a collection in MDS. A new feature is the ability to assign a weight to each collection member within the user interface, as shown in Figure 8-7. You use MDS only to store the weight for use by subscribing systems that use the weight to apportion values across the collection members. Accordingly, the subscription view includes the weight column.



Name	Code	Description	Owner
Bike to work week - US	USBIKE2WRK		

1-1 of 1 members

Name	Code	Type	Weight
Road-550-W YLO, 38	BK-R64Y-38	Leaf	0
Road-550-W YLO, 40	BK-R64Y-40	Leaf	0
Road-550-W YLO, 42	BK-R64Y-42	Leaf	0
Road-550-W YLO, 44	BK-R64Y-44	Leaf	0
Road-550-W YLO, 48	BK-R64Y-48	Leaf	0
Half-Finger Gloves, S	GL-H102-S	Leaf	0
Half-Finger Gloves, M	GL-H102-M	Leaf	0
Half-Finger Gloves, L	GL-H102-L	Leaf	0
Short-Sleeve Classic Jersey, S	SJ-0194-S	Leaf	0
Short-Sleeve Classic Jersey, M	SJ-0194-M	Leaf	0
Short-Sleeve Classic Jersey, L	SJ-0194-L	Leaf	0
Short-Sleeve Classic Jersey, XL	SJ-0194-X	Leaf	0

12 members in collection

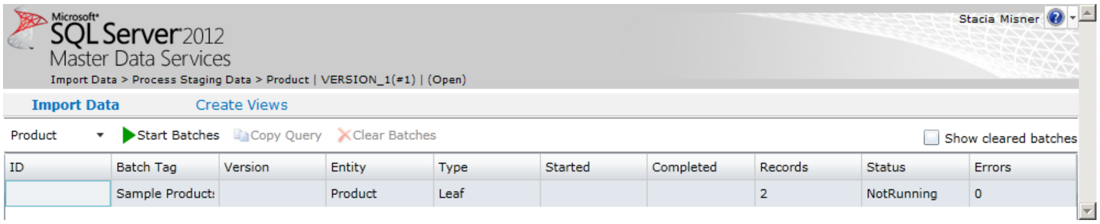
FIGURE 8-7 New collection management interface.

Integration Management

When you want to automate aspects of the master data management process, MDS now has a new, high-performance staging process. One benefit of this new staging process is the ability to load members and attribute values at one time, rather than separately in batches. To do this, you load data into the following tables as applicable, where name is the name of the staging table for the entity:

- stg.name_Leaf Use this table to stage additions, updates, or deletions for leaf members and their attributes.
- stg.name_Consolidated Use this table to stage additions, updates, or deletions for consolidated members and their attributes.
- stg.name_Relationship Assign members in an explicit hierarchy.

There are two ways to start the staging process after you load data into the staging tables—by using the Integration Management functional area or executing stored procedures. In the Integration Management functional area, you select the model in drop-down list, and click the Start Batches button, which you can see in Figure 8-8. The data processes in batches, and you can watch the status change from QueuedToRun to Running to Completed.



ID	Batch Tag	Version	Entity	Type	Started	Completed	Records	Status	Errors
	Sample Product:		Product	Leaf			2	NotRunning	0

FIGURE 8-8 Staging batch in the Integration Management functional area.

If the staging process produces errors, you see the number of errors appear in the grid, but you cannot view them in the Integration Management functional area. Instead, you can click Copy Query button to copy a SQL query that you can paste into a query window in SQL Server Management Studio. The query looks similar to this

```
SELECT * from [stg].[viw_Product_MemberErrorDetails] WHERE Batch_ID = 1
```

This view includes an ErrorDescription column describing the reason for flagging the staged record as an error. It also includes AttributeName and AttributeValue columns to indicate which attribute and value are the cause of the error.

If you use a stored procedure to execute the staging process, you use one of the following stored procedures, where name corresponds to the staging table:

- stg.udp_name_Leaf
- stg.udp_name_Consolidated
- stg.udp_name_Relationship

Each of these stored procedures takes the following parameters:

- VersionName Provide the version name of the model, such as VERSION_1. The collation setting of the SQL Server database determines whether the value for this parameter is case-sensitive.
- LogFile Use a value of 1 to log transactions during staging, or a value of 0 if you do not want to log transactions.
- BatchTag Provide a string of 50 characters or less to identify the batch in the staging table. This tag displays in the batch grid in the Integration Management functional area.

For example, to load leaf members and log transactions for the batch, execute the following code in SQL Server Management Studio:

```
EXEC [stg].[udp_name_Leaf] @VersionName = N'VERSION_1', @LogFlag = 1, @BatchTag = N'batch1'

GO
```

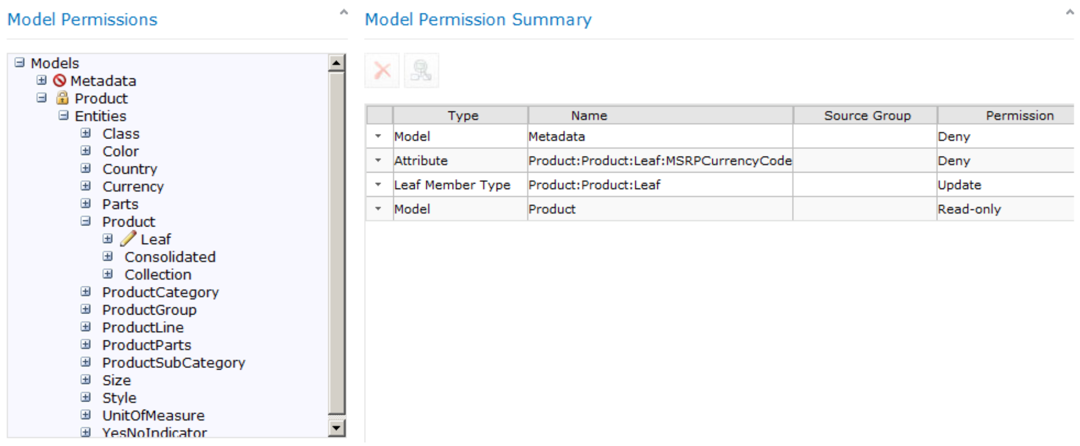
Note Transaction logging during staging is optional. You can enable logging only if you use stored procedures for staging. Logging does not occur if you launch the staging process from the Integration Management functional area.

Important No validation occurs during the staging process. You must validate manually in the Version Management functional area or use the mdm.udpValidateModel stored procedure. Refer to *http://msdn.microsoft.com/en-us/library/hh231023(SQL.110).aspx* for more information about this stored procedure.

You can continue to use the staging tables and stored procedure introduced for the SQL Server 2008 R2 MDS staging process if you like. One reason that you might choose to do this is to manage collections because the new staging process in SQL Server 2012 does not support collections. Therefore, you must use the previous staging process to create or delete collections, add members to or remove them from collections, or reactivate members and collections.

User and Group Permissions

Just as in the previous version of MDS, you assign permissions by functional area and by model object. When you assign Read-only or Update permissions for a model to a user or group on the Models tab of the Manage User page, the permission also applies to lower level objects. For example, when you grant a user or group the Update permission to the Product model, as shown in Figure 8-9, the users can also add, change, or delete members for any entity in the model and can change any attribute value. You must explicitly change permissions on selected entities to Read-only when you want to give users the ability to view but not change entity members, and to Deny when you do not want them to see the entity members. You can further refine security by setting permissions on attribute objects (below the Leaf, Consolidate, or Collection nodes) to control which attribute values a user can see or change.

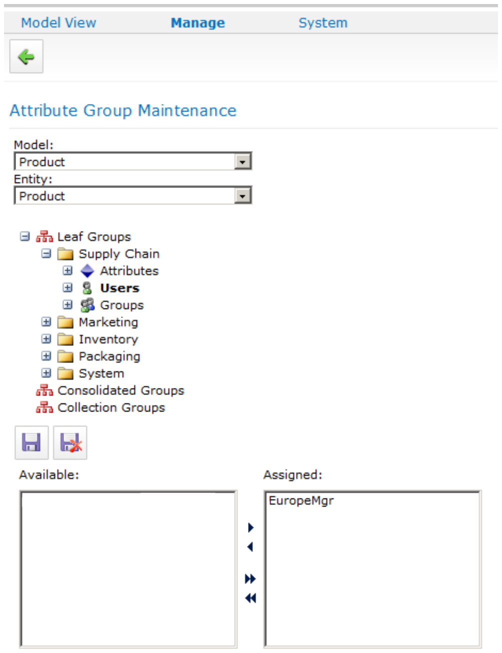


Type	Name	Source Group	Permission
Model	Metadata		Deny
Attribute	Product:Product:Leaf:MSRPCurrencyCode		Deny
Leaf Member Type	Product:Product:Leaf		Update
Model	Product		Read-only

FIGURE 8-9 Model permissions object tree and summary.

In SQL Server 2008 MDS, the model permissions object tree also includes nodes for derived hierarchies, explicit hierarchies, and attribute groups, but these nodes are no longer available in SQL Server 2012 MDS. Instead, derived hierarchies inherit permissions from the model and explicit hierarchies inherit permissions from the associated entity. In both cases, you can override these default permissions on the Hierarchy Members tab of the Manage User page to manage which entity members that users can see or change.

For attribute group permissions, you now use Attribute Group Maintenance page in the System Administration functional area to assign permissions to users or groups, as shown in Figure 8-10. Users or groups appearing in the Assigned list have Update permissions only. You can no longer assign Read-only permissions for attribute groups.



Model: Product
Entity: Product

EuropeMgr

FIGURE 8-10 Users and Groups security for attribute groups.

Model Deployment

A new high-performance command-line tool is now available for deploying packages. If you use the model deployment wizard in the web application, it deploys only the model structure. As an alternative, you can use the MDSModelDeploy tool to create and deploy a package with model objects only or a package with both model objects and data. You find this tool in the Program Files\Microsoft SQL Server\110\Master Data Services\Configuration folder.

Note You cannot reuse packages that you created using SQL Server 2008 MDS. You can deploy a SQL Server 2012 package only to SQL Server 2012 MDS instance.

The executable for this tool uses the following syntax:

```
MDSModelDeploy <commands> [ <options> ]
```

You can use the following commands with this tool:

- listservices View a list of all service instances.
- listmodels View a list of all models.
- listversions View a list of all versions for a specified model.
- createpackage Create a package for a specified model.
- deployclone Create a duplicate of a specified model, retaining names and identifiers. The model cannot exist in the target service instance.
- deploynew Create a new model. MDS creates new identifiers for all model objects.
- deployupdate Deploy a model and update the model version. This option requires you to use a package with a model having the same names and identifiers as the target model.
- help View usage, options, and examples of a command. For example, type the following command to learn how to use the *deploynew* command: MDSModelDeploy help deploynew.

To help you learn how to work with MDS, several sample packages containing models and data are available. To deploy the Product package to the MDS1 Web service instance, type the following command in the Command Prompt window:

```
MDSModelDeploy deploynew -package ..\Samples\Packages\product_en.pkg -model Product -service MDS1
```

Note To execute commands by using this utility, you must have permissions to access the System Administration functional area and you must open the Command Prompt window as an administrator.

During deployment, MDS first creates the model objects, and then creates the business rules and subscription views. MDS populates the model with master data as the final step. If any of these steps fail during deployment of a new or cloned model, MDS deletes the model. If you are updating a model, MDS retains the changes from the previous steps that completed successfully unless the failure occurs in the final step. In that case, MDS updates master data members where possible rather than failing the entire step and rolling back.

Note After you deploy a model, you must manually update user-defined metadata, file attributes, and user and group permissions.

MDS Add-In for Excel

The most extensive addition to MDS in SQL Server 2012 is the new user interface option for enabling data stewards and administrators to manage master data inside Microsoft Excel. Data stewards can retrieve data and make changes using the familiar environment of Excel after installing the MDS Add-in for Excel. Administrators can also use this add-in to create new model objects such as entities and load data into MDS.

Installation of the MDS Add-In

By default, the home page of Master Data Manager includes a link to the download page for the MDS Add-in on the Microsoft web site. On the download page, you choose the language and version (32-bit or 64-bit) that matches your Excel installation. You open the MSI file that downloads to start the setup wizard, and then follow the prompts to accept the license agreement and confirm the installation. When the installation completes, you can open Excel to view the new Master tab in the ribbon, as shown in Figure 8-11.

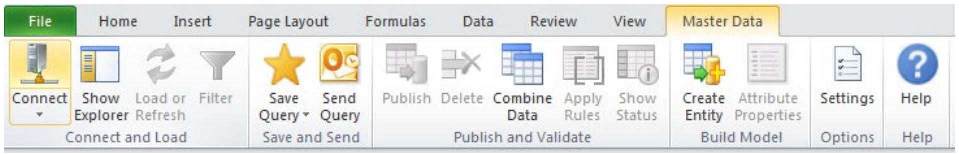


FIGURE 8-11 Master Data tab in Excel ribbon.

Note The add-in works with either Excel 2007 or Excel 2010.

Master Data Management

The MDS add-in supports the primary tasks that you need to perform for master data management. After connecting to MDS, you can load data from MDS into a worksheet to use for reference or to make additions or changes in bulk. You can also apply business rules and correct validation issues, check for duplicates using Data Quality Services integration, and then publish the modified data back to MDS.

Connections

Before you can load MDS data into a worksheet, you must create a connection to the MDS database. If you open a worksheet into which you previously loaded data, the MDS add-in automatically connects to MDS when you refresh the data or publish the data. To create a connection in the MDS Add-in for Excel, follow these steps:

1. On the Master Data tab of the ribbon, click the arrow under the Connect button, and click Manage Connections.

2. In the Manage Connections dialog box, click the Create A New Connection link.

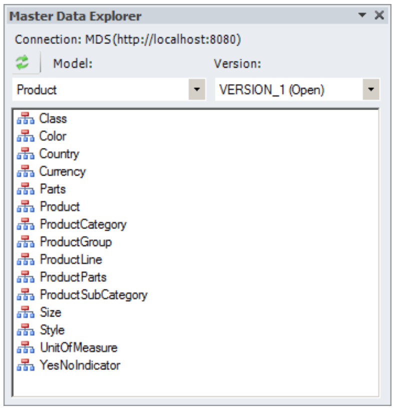
3. In the Add New Connection dialog box, type a description for the connection. This description displays when you click the arrow under the Connect button.

4. In the MDS Server Address box, type the URL that you use to open the Master Data Manager web application, such as http://myserver/mds, and click the OK button. The connection displays in the Existing Connections section of the Manage Connections dialog box.

5. Click the Test button to test the connection, and then click the OK button to close the dialog box that confirms the connection or displays an error.

6. Click the Connect button.

7. In the Master Data Explorer pane, select a model and version from the respective drop-down lists, as shown below.



Data Retrieval

Before you load data from MDS into a worksheet, you can filter the data. Even if you do not filter the data, there are some limitations to the volume of data that you can load. Periodically, you can update the data in the worksheet to retrieve the latest updates from MDS.

Filter data   Rather than load all entity members from MDS into a spreadsheet, which can be a time-consuming task, you can select attributes and apply filters to minimize the amount of data that you retrieve from MDS. You can choose to focus on selected attributes to reduce the number of columns to retrieve. Another option is to use filter criteria to eliminate members.

To filter and retrieve leaf data from MDS, follow these steps:

1. In the Master Data Explorer pane, select the entity that you want to load into the spreadsheet.

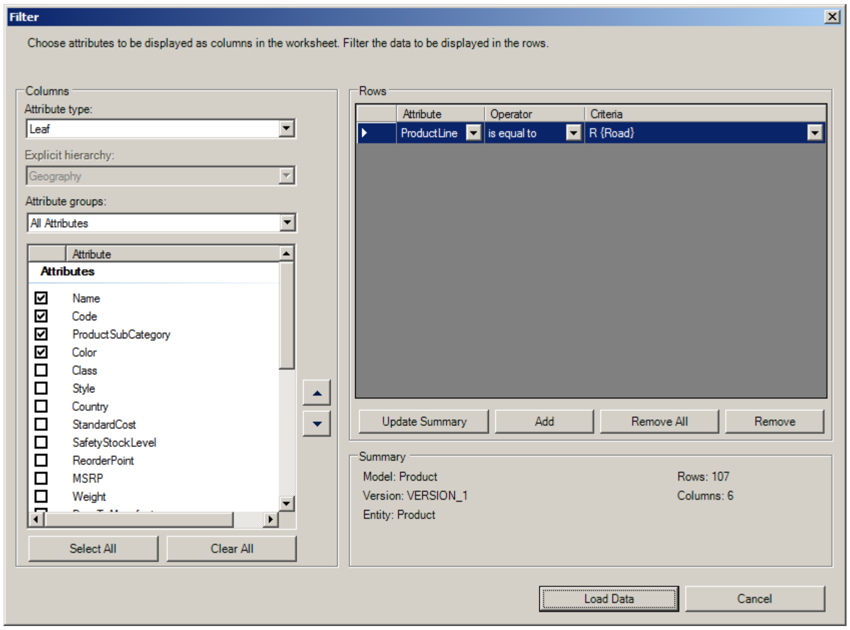
2. On the Master Data tab of the ribbon, click the Filter button.

3. In the Filter dialog box, select the columns to load by selecting an attribute type, an explicit hi-

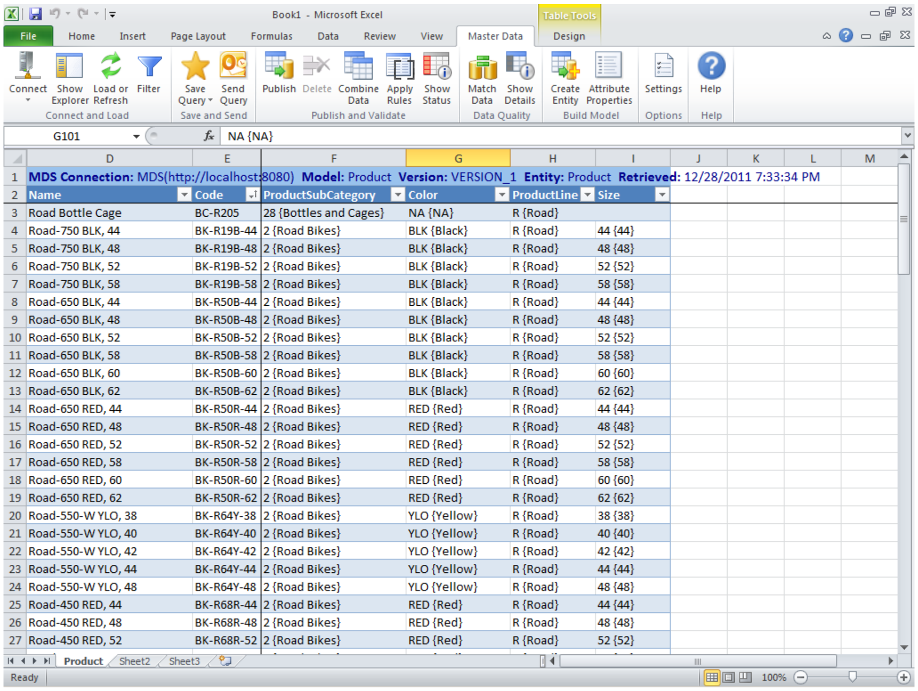
erarchy (if you select the Consolidated attribute type), an attribute group, and individual attributes.

> **Tip** You can change the order of attributes by using the Up and Down arrows to the right of the attribute list to move each selected attribute.

4. Next, select the rows to load by clicking the Add button, and then selecting an attribute, a filter operator, and filter criteria. You can repeat this step to continue adding filter criteria.
5. Click the Update Summary button to view the number of rows and columns resulting from your filter selections, as shown below.



6. In the Filter dialog box, click the Load button. The data loads into the current spreadsheet, as shown below.



G101 NA {NA}

MDS Connection: MDS(http://localhost:8080) **Model:** Product **Version:** VERSION_1 **Entity:** Product **Retrieved:** 12/28/2011 7:33:34 PM

	D	E	F	G	H	I
2	Name	Code	ProductSubCategory	Color	ProductLine	Size
3	Road Bottle Cage	BC-R205	28 {Bottles and Cages}	NA {NA}	R {Road}	
4	Road-750 BLK, 44	BK-R19B-44	2 {Road Bikes}	BLK {Black}	R {Road}	44 {44}
5	Road-750 BLK, 48	BK-R19B-48	2 {Road Bikes}	BLK {Black}	R {Road}	48 {48}
6	Road-750 BLK, 52	BK-R19B-52	2 {Road Bikes}	BLK {Black}	R {Road}	52 {52}
7	Road-750 BLK, 58	BK-R19B-58	2 {Road Bikes}	BLK {Black}	R {Road}	58 {58}
8	Road-650 BLK, 44	BK-R50B-44	2 {Road Bikes}	BLK {Black}	R {Road}	44 {44}
9	Road-650 BLK, 48	BK-R50B-48	2 {Road Bikes}	BLK {Black}	R {Road}	48 {48}
10	Road-650 BLK, 52	BK-R50B-52	2 {Road Bikes}	BLK {Black}	R {Road}	52 {52}
11	Road-650 BLK, 58	BK-R50B-58	2 {Road Bikes}	BLK {Black}	R {Road}	58 {58}
12	Road-650 BLK, 60	BK-R50B-60	2 {Road Bikes}	BLK {Black}	R {Road}	60 {60}
13	Road-650 BLK, 62	BK-R50B-62	2 {Road Bikes}	BLK {Black}	R {Road}	62 {62}
14	Road-650 RED, 44	BK-R50R-44	2 {Road Bikes}	RED {Red}	R {Road}	44 {44}
15	Road-650 RED, 48	BK-R50R-48	2 {Road Bikes}	RED {Red}	R {Road}	48 {48}
16	Road-650 RED, 52	BK-R50R-52	2 {Road Bikes}	RED {Red}	R {Road}	52 {52}
17	Road-650 RED, 58	BK-R50R-58	2 {Road Bikes}	RED {Red}	R {Road}	58 {58}
18	Road-650 RED, 60	BK-R50R-60	2 {Road Bikes}	RED {Red}	R {Road}	60 {60}
19	Road-650 RED, 62	BK-R50R-62	2 {Road Bikes}	RED {Red}	R {Road}	62 {62}
20	Road-550-W YLO, 38	BK-R64Y-38	2 {Road Bikes}	YLO {Yellow}	R {Road}	38 {38}
21	Road-550-W YLO, 40	BK-R64Y-40	2 {Road Bikes}	YLO {Yellow}	R {Road}	40 {40}
22	Road-550-W YLO, 42	BK-R64Y-42	2 {Road Bikes}	YLO {Yellow}	R {Road}	42 {42}
23	Road-550-W YLO, 44	BK-R64Y-44	2 {Road Bikes}	YLO {Yellow}	R {Road}	44 {44}
24	Road-550-W YLO, 48	BK-R64Y-48	2 {Road Bikes}	YLO {Yellow}	R {Road}	48 {48}
25	Road-450 RED, 44	BK-R68R-44	2 {Road Bikes}	RED {Red}	R {Road}	44 {44}
26	Road-450 RED, 48	BK-R68R-48	2 {Road Bikes}	RED {Red}	R {Road}	48 {48}
27	Road-450 RED, 52	BK-R68R-52	2 {Road Bikes}	RED {Red}	R {Road}	52 {52}

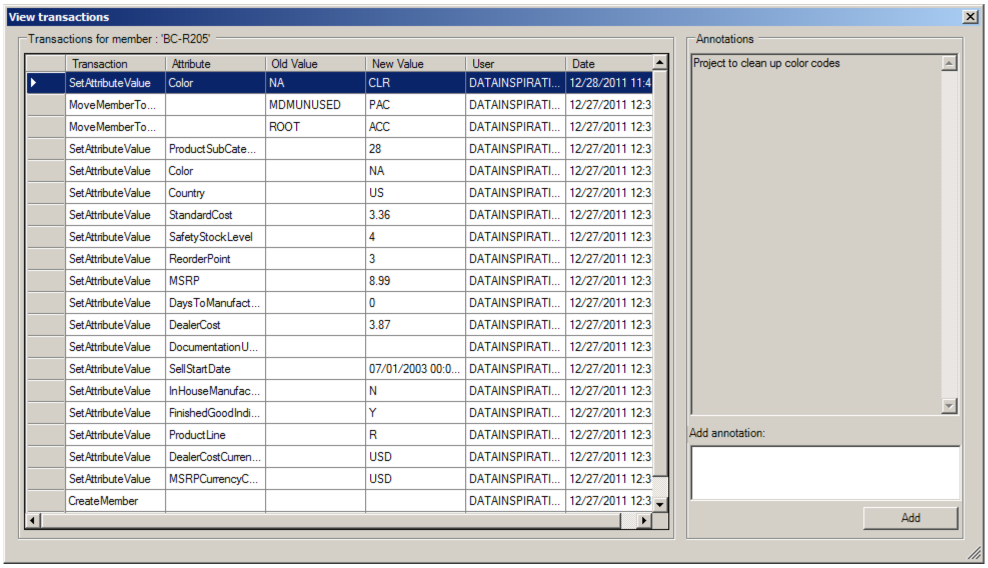
Load data Filtering data before you loading is optional. You can load all members for an entity by clicking the Load And Refresh button in the ribbon. A warning displays if there are more tan 100,000 rows or more than 100 columns, but you can increase or decrease these values or disable the warning by clicking the Settings button on the ribbon, and changing the properties on the Data page of the Settings dialog box. Regardless, if an entity is very large, the add-in automatically restricts the retrieval of data to the first one million members. Also, if a column is a domain-based attribute, then the add-in retrieves only the first 1,000 values.

Refresh data After you load MDS data into a worksheet, you can update the same worksheet by adding columns of data from sources other than MDS or columns containing formulas. When you want to refresh the MDS data without losing the data you added, click the Load And Refresh button in the ribbon.

The refresh process modifies the contents of the worksheet. Deleted members disappear, and New members appear at the bottom of the table with green highlighting. Attribute values update to match the value stored in MDS, but the cell does not change color to identify a new value.

> **Warning** If you add new members or change attribute values, you must publish these changes before refreshing the data. Otherwise, you lose your work. Cell comments on MDS data are deleted, and non-MDS data in rows below MDS data might be replaced if the refresh process adds new members to the worksheet.

Review transactions and annotations You can review transactions for any member by right-clicking the member's row and selecting View Transactions in the context menu, as shown in Figure 8-12. To view an annotation or to add an annotation for a transaction, select the transaction row in the View Transactions dialog box.



Transaction	Attribute	Old Value	New Value	User	Date
SetAttributeValue	Color	NA	CLR	DATAINSPIRATI...	12/28/2011 11:4
MoveMemberTo...		MDMUNUSED	PAC	DATAINSPIRATI...	12/27/2011 12:3
MoveMemberTo...		ROOT	ACC	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	ProductSubCate...		28	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	Color		NA	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	Country		US	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	StandardCost		3.36	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	SafetyStockLevel		4	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	ReorderPoint		3	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	MSRP		8.99	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	DaysToManufact...		0	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	DealerCost		3.87	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	DocumentationU...			DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	SellStartDate		07/01/2003 00:0...	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	InHouseManufac...		N	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	FinishedGoodIndi...		Y	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	ProductLine		R	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	DealerCostCurren...		USD	DATAINSPIRATI...	12/27/2011 12:3
SetAttributeValue	MSRPCurrencyC...		USD	DATAINSPIRATI...	12/27/2011 12:3
CreateMember				DATAINSPIRATI...	12/27/2011 12:3

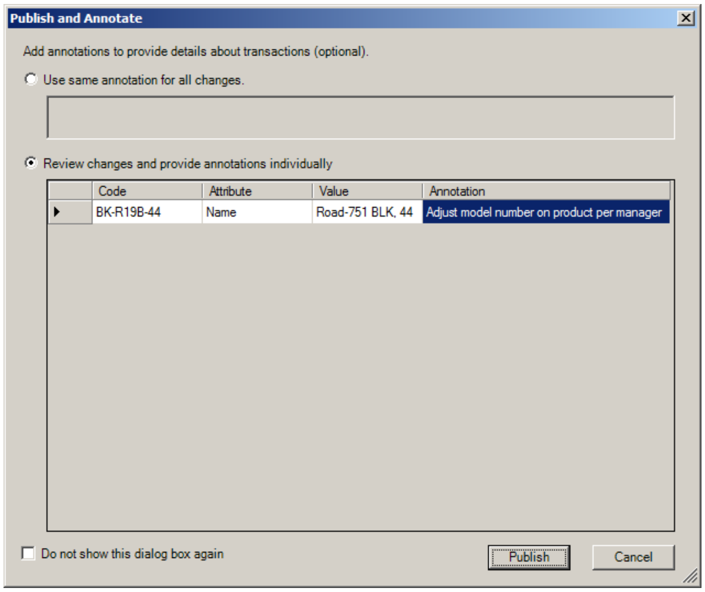
Project to clean up color codes

FIGURE 8-12 Transactions and annotations for a member.

Data Publication

If you make changes to the MDS data in the worksheet, such as altering attribute values, adding new members, or deleting members, you can publish your changes to MDS to make it available to other users. Each change that you make saves to MDS as a transaction, which you have the option to annotate to document the reason for the change. An exception is a deletion, which you cannot annotate although the deletion does generate a transaction.

When you click the Publish button, the Publish And Annotate dialog box displays (unless you disable it in Settings). You can provide a single annotation for all changes or separate annotations for each change, as shown in Figure 8-13. An annotation must be 500 characters or less.


Publish and Annotate
Add annotations to provide details about transactions (optional).
Use same annotation for all changes.
Review changes and provide annotations individually
Do not show this dialog box again
Publish
Cancel

Code	Attribute	Value	Annotation
BK-R19B-44	Name	Road-751 BLK, 44	Adjust model number on product per manager

FIGURE 8-13 Addition of annotation for published data change.

Warning Cell comments on MDS data are deleted during the publication process. Also, a change to the Code value for a member does not save as a transaction, and renders all previous transactions for that member inaccessible.

During the publication process, MDS validates your changes. First, MDS applies business rules to the data. Second, MDS confirms the validity of attribute values, including the length and data type. If a member passes validation, the MDS database updates with the change. Otherwise, the invalid data displays in the worksheet with red highlighting appears and the description of the error appears in the $InputStatus$ column. You can apply business rules prior to publishing your changes by using the Apply Rules button in the ribbon.

Model Building Tasks

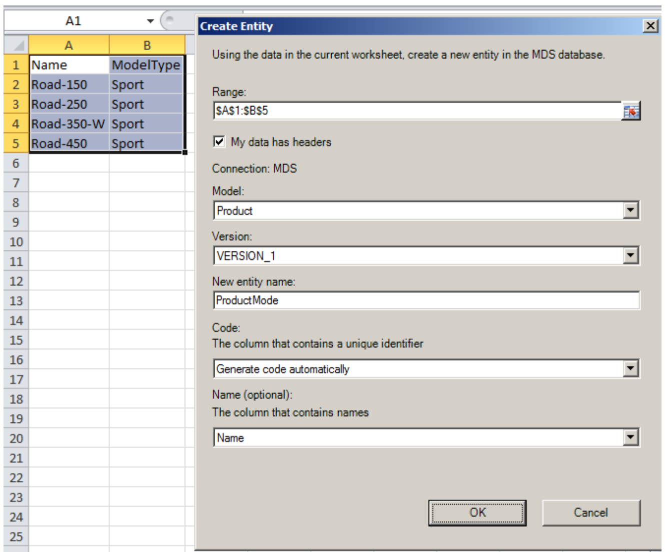
Use of the MDS add-in for Excel is not limited to data stewards. If you are an administrator, you can also use the add-in to create entities and add attributes. However, you must first create a model by using Master Data Manager, and then you can continue add entities to the model by using the add-in.

Entities and Attributes

Before you add an entity to MDS, you create data in a worksheet. The data must include a header row, and at least one row of data. Each row should include at least a Name column. If you include a Code column, the column values must be unique for each row. You can add other columns to create attributes for the entity, but you do not need to provide values for them. If you do, you can use text, numeric, or data values, but you cannot use formulas or time values.

To create a new entity in MDS, follow these steps:

1. Select all cells in the header and data rows to load into the new entity.
2. Click the Create Entity button in the ribbon.
3. In the Create Entity dialog box, ensure the range includes only the data you want to load and do not clear the My Data Has Headers check box.
4. Select a model and version from the respective drop-down lists, and provide a name in the New Entity Name box.
5. In the Code drop-down list, select the column that contains unique values for entity members or select the Generate Code Automatically option.
6. In the Name drop-down list, select the column that contains member names, as shown below, and then click OK. The add-in creates the new entity in the MDS database and validates the data.



	A	B
1	Name	ModelType
2	Road-150	Sport
3	Road-250	Sport
4	Road-350-W	Sport
5	Road-450	Sport

Create Entity

Using the data in the current worksheet, create a new entity in the MDS database.

Range: $A$1:$B$5

[✓] My data has headers

Connection: MDS

Model: Product

Version: VERSION_1

New entity name: ProductMode

Code: The column that contains a unique identifier: Generate code automatically

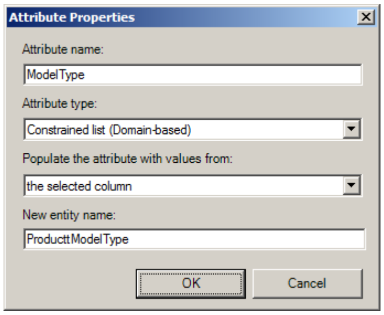
Name (optional): The column that contains names: Name

Note You might need to correct the data type or length of an attribute after creating the entity. To do this, click any cell in the attribute's column, and click the Attribute Properties button in the ribbon. You can make changes as necessary in the Attribute Properties dialog box. However, you cannot change the data type or length of the Name or Code columns.

Domain-based Attributes

If you want to restrict column values of an existing entity to a specific set of values, you can create a domain-based attribute from values in a worksheet or an existing entity. To create a domain-based attribute, follow these steps:

1. Load the entity into a worksheet, and click a cell in the column that you want to change to a domain-based attribute.
2. Click the Attribute Properties button in the ribbon.
3. In the Attribute Properties dialog box, select Constrained List (Domain-based) in the Attribute Type drop-down list.
4. Select an option from the Populate The Attribute With Values From drop-down list. You can choose The Selected Column to create a new entity based on the values in the selected column, as shown below, or you can choose an entity to use values from the selected entity.



Attribute name: ModelType
Attribute type: Constrained list (Domain-based)
Populate the attribute with values from: the selected column
New entity name: ProducttModelType

5. Click OK. The column now allows you to select from a list of values. You can change the available values in the list by loading the entity on which the attribute is based into a separate worksheet, making changes by adding new members or updating values, and then publishing the changes back to MDS.

Shortcut Query Files

You can easily load frequently accessed data by using a shortcut query file. This file contains information about the connection to the MDS database, the model and version containing the MDS data, the entity to load, filters to apply, and the column order. After loading MDS data into a worksheet, you create a shortcut query file by clicking the Save Query button in the ribbon, and selecting Save As Query in the menu. When you want to use it later, you open an empty worksheet, click the Save Query button, and select the shortcut query file from the list that displays.

You can also use the shortcut query file as a way to share up-to-date MDS data with other users without emailing the worksheet. Instead, you can email the shortcut query file as long as you have Outlook 2010 or later installed on your computer. First, load MDS data into a worksheet and then click the Send Query button in the ribbon to create an email message with the shortcut query file as an attachment. As long as the recipient of the email message has the add-in installed, he or she can double-click the file to open it.

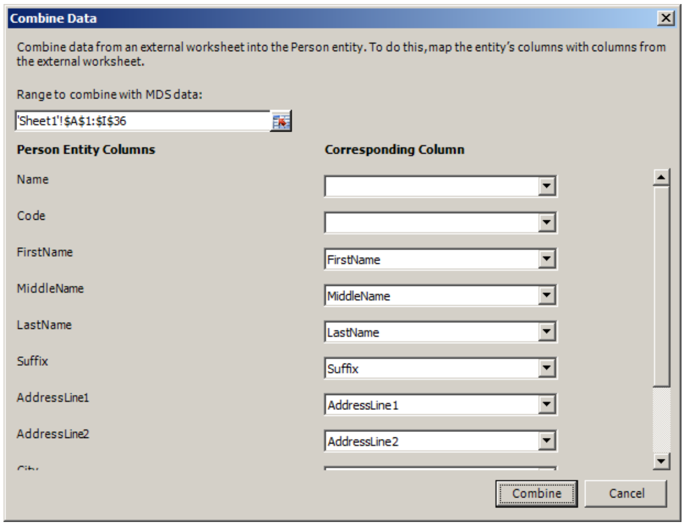
Data Quality Matching

Before you add new members to an entity using the add-in, you can prepare data in a worksheet and combine it with MDS data for comparison. Then you use DQS to identify duplicates. The matching process adds detail columns to show matching scores which you can use to decide which data to publish to MDS.

As we explained in Chapter 7, "Data Quality Services," you must enable Data Quality Services (DQS) integration in the Master Data Services Configuration Manager and create a matching policy in a knowledge base. In addition, both the MDS database and the DQS_MAIN database must exist in the same SQL Server instance.

The first step in the data quality matching process is to combine data from two worksheets into a single worksheet. The first worksheet must contain data that you load from MDS. The second worksheet must contain data with a header row and one or more detail rows. To combine data, follow these steps:

1. On the first worksheet, click the Combine Data button in the ribbon.
2. In the Combine Data dialog box, click the icon next to the Range To Combine With MDS Data text box.
3. Click the second worksheet and highlight the header and detail rows to combine with MDS data.
4. In the Combine Data dialog box, click the icon to the right of the Range To Combine With MDS Data box.
5. Navigate to the second worksheet and highlight the header row and detail rows, and then click the icon next to the range in the collapsed Combine Data dialog box.
6. In the expanded Combine Data dialog box, in the Corresponding Column drop-down list, select a column from the second worksheet that corresponds to the entity column that displays to its left, as shown below.


Combine Data

Combine data from an external worksheet into the Person entity. To do this, map the entity's columns with columns from the external worksheet.

Range to combine with MDS data:

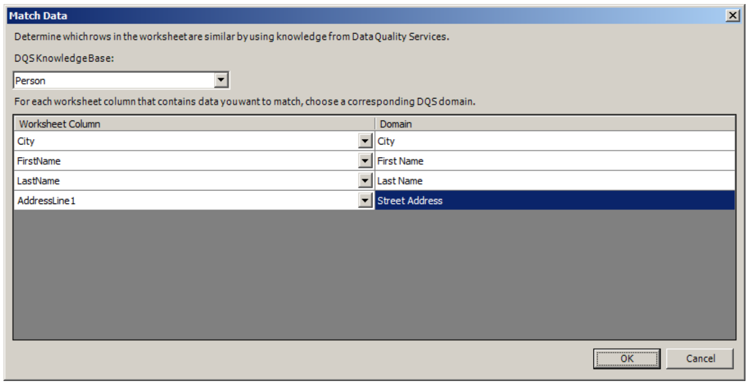
'Sheet1'!$A$1:$I$36

Person Entity Columns	Corresponding Column
Name	
Code	
FirstName	FirstName
MiddleName	MiddleName
LastName	LastName
Suffix	Suffix
AddressLine1	AddressLine1
AddressLine2	AddressLine2

7. Click the Combine button. The rows from the second worksheet display in the first worksheet below the existing rows, and the SOURCE column displays whether the row data comes from MDS or from an external source, as shown below.

8. Click the Match Data button in the ribbon.

9. In the Match Data dialog box, select a knowledge base from the DQS Knowledge Base drop-down list, and map worksheet columns to each domain listed in the dialog box.


Match Data

Determine which rows in the worksheet are similar by using knowledge from Data Quality Services.

DQS KnowledgeBase:

Person

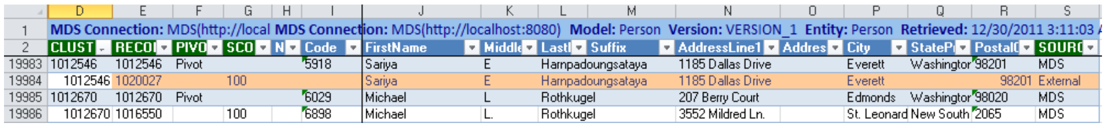
For each worksheet column that contains data you want to match, choose a corresponding DQS domain.

Worksheet Column	Domain
City	City
FirstName	First Name
LastName	Last Name
AddressLine1	Street Address

Note Rather than use a custom knowledge base as shown above, you can use the default knowledge base, DQS Data. In that case, you add a row to the dialog box for each column that you want to use for matching and assign a weight value. The sum of weight values for all rows must equal 100.

10. Click OK, and then click the Show Details button in the ribbon to view columns containing matching details. The SCORE column indicates the similarity between the pivot record (indicat-

ed by Pivot in the PIVOT_MARK column) and the matching record. You can use this information to eliminate records from the worksheet before publishing your changes and additions to MDS.



MDS Connection: MDS(http://local MDS Connection: MDS(http://localhost:8080) Model: Person Version: VERSION_1 Entity: Person Retrieved: 12/30/2011 3:11:03

	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
2	CLUST	RECOI	PIVO	SCO	N	Code	FirstName	Middle	LastI	Suffix	AddressLine1	Addres	City	StateP	Postal(	SOUR(
19983	1012546	1012546	Pivot			5918	Sariya	E	Harnpadoungsataya		1185 Dallas Drive		Everett	Washingtor	98201	MDS
19984	1012546	1020027		100			Sariya	E	Harnpadoungsataya		1185 Dallas Drive		Everett		98201	External
19985	1012670	1012670	Pivot			6029	Michael	L	Rothkugel		207 Berry Court		Edmonds	Washingtor	98020	MDS
19986	1012670	1016550		100		6898	Michael	L.	Rothkugel		3552 Mildred Ln.		St. Leonard	New South	2065	MDS

Miscellaneous Changes

Thus far in this chapter, our focus has been the new features available in MDS. However, there are some more additions and changes to review. SQL Server 2012 offers some new features for SharePoint integration, and retains some features from the SQL Server 2008 R2 version that are still available, but deprecated. There are also features that are discontinued. In this section, we review these feature changes and describe alternatives where applicable.

SharePoint Integration

There are two ways that you can integrate MDS with SharePoint. First, when you add the Master Data Manager website to a SharePoint page, you can add &hosted=true as a query parameter to reduce the amount of required display space. This query parameter removes the header, the menu bar, and the padding at the bottom of the page. Second, you can save shortcut query files to a SharePoint document library to provide lists of reference data to other users.

Metadata

The Metadata model continues to display in Master Data Manager, but is deprecated. Microsoft recommends that you do not use it because it will be removed in a future release of SQL Server. You cannot create versions of the Metadata model and users cannot view metadata in the Explorer functional area.

Bulk Updates and Export

Making changes to master data one record at a time can be a tedious process. In the previous version of MDS, you can update an attribute value for multiple members at the same time, but this capability is no longer available in SQL Server 2012. Instead, you can use the staging process to load the new values into the stg.name_Leaf table as we describe in the "Integration Management" section of this chapter. As an alternative, you can use the MDS add-in to load the entity into an Excel worksheet (described in the "Master Data Management" section of this chapter), update the attribute values in bulk using copy and paste, and then publish the results to MDS.

The purpose of storing master data in MDS is to have access to this data for other purposes. You use the Export to Excel button on the Member Information page when using the previous version of MDS, but this button is not available in SQL Server 2012. When you require MDS data in Excel, you use the MDS add-in to load entity members from MDS into a worksheet as we describe in the "Data Retrieval" section of this chapter.

Transactions

MDS uses transactions to log every change that users make to master data. In the previous version, users review transactions in the Explorer functional area and optionally reverse their own transactions to restore a prior value. Now only administrators can revert transactions in the Version Management functional area.

MDS allows you to annotate transactions. In SQL Server 2008 R2, MDS stores annotations as transactions and allows you to delete them by reverting the transaction. However, annotations are now permanent in SQL Server 2012. Although you continue to associate an annotation with a transaction, MDS stores the annotation separately and does not allow you to delete it.

PowerShell

In the previous version of MDS, you can use PowerShell cmdlets for administration of MDS. One cmdlet allows you to create the database, another cmdlet allows you to configure settings for MDS, and other cmdlets allow you to retrieve information about your MDS environment. No cmdlets are available in the current version of MDS.