User Guide

DATA MANAGEMENT

for Microsoft® System Center

PROVANCE

This document is provided for informational purposes only and Provance makes no warranties, either express or implied, in this document. Information in this document, including external references, is subject to change without notice. The entire risk of the use or the results from the use of this document remains with the user. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into an electronic retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Provance Technologies, Inc.

Provance Technologies, Inc. may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Provance, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Provance Data Management for System Center Service Manager 2012 R2

© 2014 Provance Technologies, Inc. All rights reserved.
Provance Technologies Inc.
490 Saint-Joseph Boulevard, 2nd Floor
Gatineau, Quebec
CANADA
J8Y 3Y7

Provance® and the Provance logo are either trademarks or registered trademarks of Provance Technologies Inc.
Microsoft, SQL Server, Windows, Windows Server, and Active Directory are either registered trademarks or trademarks of Microsoft.

All other trademarks are property of their respective owners.

Contents

Getting started 1-1
Understanding the terminology 1-2
How data import works 1-3
Comparing the features of import tools 1-5
About mapping data 1-6
Property types and actions for them 1-7
When the source differs from the destination 1-10
How to import data 1-11
Permissions required 1-12
Importing a csv file 1-12
Prerequisites for a csv file 1-12
To import data from a csv file: 1-13
Importing SQL Server data 1-13
Prerequisites for a SQL Server import 1-13
To import data from a SQL Server query: 1-14
Importing through LDAP 1-15
To import data from an Active Directory domain or OU: 1-15
Importing through PowerShell 1-16
To run the PowerShell cmdlet: 1-16
To load the PowerShell cmdlet: 1-17
Parameters used by the cmdlet 1-18
Sample cmdlet calls 1-20
Importing through a template 1-21
Creating and configuring objects 1-22
To create and configure an object: 1-22
Configuring general properties 1-22
Configuring map properties 1-25
Configuring Criteria 1-26
Creating relationships between objects 1-27

Configuring general properties 1-28
Configuring map properties 1-31
How to create a relationship 1-32
How to remove (delete) a relationship 1-32
Completing the import wizard 1-34
Cancelling an import 1-35
Error handling options 1-35
Using import templates 1-36
Saving import templates as XML 1-37
Loading import templates 1-37
Editing import templates 1-38
PowerShell import template parameters 1-39
PowerShell sample calls 1-40

Getting started

Data Management is a data import management pack for Microsoft System Center 2012 - Service Manager. Use its wizards to migrate data from other systems or sources into Service Manager, and control how the data gets created. Your data can be converted and imported into the Configuration Management Database (CMDB) as objects, such as configuration items (CIs). Depending on your criteria, the management pack may either update existing objects with new data, or create new objects. If the source data contains relationships, then they are maintained within the new objects. Any related objects can also be created as part of the import. For example, the creation of a hardware asset with a company as the supplier of the asset results in a hardware asset being created as well as a relationship to the supplier. If the source contains enough data about the supplier, then it could also be created.

A data map is used to define the objects and properties that are updated or created and how they relate to your source data. It can also be used to change the data during the import. If the object has a Status property, and the source data has incorrect or missing values for status, then you could replace status with a default value.

It is also possible to export data and then reimport the data after it has been modified. For example, you may export software licenses, add upgrade information to them, then reimport those licenses. Use the Export Data wizard, provided with the IT Asset Management Pack, to create a csv file of the licenses. Add upgrade information to the file, then use the Data Management Import wizard to reimport the licenses.

Understanding the terminology

The following terms are used throughout this document to describe the functionality of Data Management:

Term	Description
class	A set of properties that are related to a destination object. For example, the hardware asset class defines the properties of a hardware asset, which include a model, a manufacturer, a life cycle status, and a location. The management pack imports data into classes and their properties. A configuration item is defined by its class.
configuration item	Any object that you manage for the purpose of delivering a service. An object can have relationships to other objects. An object can have children as well as a parent. Some examples of configuration items include hardware assets, users, contracts and software titles.
criteria	Criteria are used during object mapping to determine if a property is going to be updated or created.
csv file	A file with comma separated values that is used to import data into Service Manager.
header	The first row in your source file or database, which describes the contents of each column in the remaining rows.
field	A single value or cell in the source that is used to populate one or more properties.
object	An object and a CI are used interchangeably in this document. See **configuration item**.
object instance	An actual instance of an object, such as a **T510 laptop computer** or the company entitled **Lenovo**.
projection	A projection contains one or more classes and their relationships.
property	A property defines a specific attribute of a class. For example, the properties of a contract include the contract manager, the status, the publisher, and the dates. Data is imported into the properties of a class.

Term	Description
query	A query returns rows or records from an Active Directory Lightweight Directory Access Protocol or SQL Server database.
record	A record is a row in a csv or a row returned by a query. A row consists of a series of fields that correspond to the properties of an object. A single record in the source is used to create or update one object.

How data import works

Import Step	Description
1. Determine the Import Type	Using Data Management there are several ways to import data into Service Manager: from a csv file, from a database and from an LDAP source. Determine which method suits your needs then provide the connection details to the wizard.
2. Review Validation Results	The source data is verified. See "Importing a csv file" on page 12 and "Importing SQL Server data" on page 13 for more information about what is required.
3. Map Your Data	A data map transforms source data into objects in the CMDB or it updates existing objects. Create a map with the import wizard and determine: • which object you wish to create or update • which properties to import and which properties to ignore. For example, if you have no content for the Refresh Date, then you would not import anything into that property. Your content may contain some or all of the properties of the object, but you choose which ones to import and which ones not to import. Properties that are required, identified with a red asterisk, must have data in them. Required properties are not essential when *updating* existing objects.

Import Step	Description
3. Map Your Data (continued)	• what changes to make to the content as it is being imported. For example, change the value of Notes during the import so that all the computers created have a value that is different from your source content operation. There are a variety of ways you can change the content during an import, such as concatenate or substring. For more details on how you can change the content, see "Property types and actions for them" on page 7 and "About mapping data" on page 6. • what relationships to create during the import. Your source data may contain information about multiple objects, which means you may need to create relationships between them during the import. You can either use values in your source data for the objects in the relationship, or you can enter new values.
4. Review Import Results	The source data is either added as new objects to Service Manager or as updates to existing objects. If errors occur during the import they are summarized as part of the results. The type of error handling you chose in the beginning determines how the import behaves when an error occurs. For more details about error handling, see "Error handling options" on page 35.

Comparing the features of import tools

The features of Data Management and csv import (provided by Service Manager) are identified here so you can decide which tool best suits your import requirements.

Feature	Data Management Import	Csv Import
Create new objects	✓	✓
Update existing objects based on certain criteria	✓	
Create and update relationships between objects	✓	
Create an object and update it in the same process	✓	✓†
Create duplicate objects	✓	
Split one object into several different objects (using relationships)	✓	
Change a property as it is being imported	✓	
Create objects having different values for a certain property	✓	✓
Create objects having the same value for a certain property	✓	
Create templates for mapping source data to destination objects and properties	✓	
Supports multiple formats for the data source	✓	

†You must know the unique ID of each object.

About mapping data

When you map data during an import, you define which object is created or updated, what content is imported, and what changes, if any, are made to the source data. This "data map" encompasses everything needed, including the destination object, the properties of that object, and what action to perform on the source data as it is being transformed into a property. The entire data map can be saved as an XML template and shared between multiple systems through a template import.

How you map depends on the type of property. For example, a property that is a list is treated differently from a property that is a text field.

In summary, a data map permits you to:

- copy the contents of your source file or database exactly as-is
- substitute some of the source content with new values that you manually input
- substitute some of the source content with predefined list values recognized by Service Manager
- choose to ignore a particular property, i.e., not copy any values into it. This is useful if you are updating an object and you are not changing all of its properties.
- manipulate the source content during the import, such as concatenating several values together
- create duplicate objects from the source content
- split the source content, i.e., create different objects based on the existence of relationships in the content

More details on how to perform some of these actions with the data map can be found in "Property types and actions for them" on page 7.

Property types and actions for them

Once you choose the destination object, you need to determine what properties to populate and what actions to perform on them. Will the property be copied directly from the source? Will the property be ignored (not updated or left empty)? Will the property be replaced with a value that is different from the source? Each property, because of its nature, will permit different kinds of actions on it. All property types are listed in Table 1, along with the actions that are available to them when mapping source data.

NOTE: *The property types and actions described here are also applicable when creating relationships between objects.*

TABLE 1. Property Types and their Actions

Property Type	Action	Description
Boolean	None	Do not import a value for the property. Leave a property as-is if you are updating an existing object, or if you have no content for the property
	Value	The property requires a Boolean value, which must be entered as true/false.
Date	None	See Boolean property for description.
	Copy	Select the column from the source that contains the date and choose the format used in the source. The formats supported for the custom date are described by Microsoft as .NET Framework 1.1 Custom DateTime format strings. In general the format specifiers are: d for day, M for month, y for year, and m for minute. For example, enter dd/MM/yy for dates formatted in your source as 06/03/11. Please visit http://msdn.microsoft.com for more information on this format.
	Value	Choose a new date with the date selection tool. This action ignores your source data.
Decimal	None	See Boolean property for description.

TABLE 1. Property Types and their Actions (Continued)

Property Type	Action	Description
Decimal (cont'd)	Copy	Use the contents of the source file or database for the property. Select the column from your source which corresponds to the property being imported. A decimal value can only have three decimal places. If it has more than two places, it is rounded up to two places. Only use your source content if it is valid for the property. If your content is not valid, then choose Value to manually input a correct value for the property.
	Value	Enter a new value for the property. This action ignores the source data. A decimal value normally has only two decimal places. If it has more than two places, it is rounded up to two places. Each object created or updated will contain the same value for the property. Use Value when your source has no value for the property, or when you do not wish to use its value.
Integer	None	See Boolean property for description.
	Copy	See Decimal property for description, but consider that an integer can not have any decimal places.
	Value	See Decimal property for description.
List	None	See Boolean property for description.
	List Copy	Copies the value from your source content as-is. Ensure that the values in the source are valid for the list. If they are not, use List Lookup to map your source list values to their correct predefined list values.

TABLE 1. Property Types and their Actions (Continued)

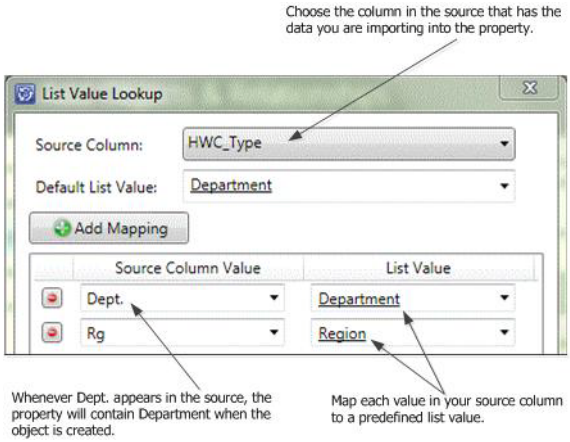
Property Type	Action	Description
List (cont'd)	List Lookup	Choose List Lookup in either of these situations: when the source content has values that are different from the predefined ones for a given property; or when you have multiple values to map against a property. In the first situation, if your source contains 1 and you map 1 to the predefined value retired, then all the destination objects will have a property of retired whenever the source contains 1. In the second situation, your source may contain the values 1, 2, 3, and 4, which need to be equated to a specific life cycle status. Use the List Lookup option to equate 1 to installed, 2 to retired, 3 to disposed, and 4 to shipped. The objects created will contain life cycle statuses according to the mapping you defined.
	List Value	Ignore the contents of the source file or database. Instead, select a predefined value for the property. Each object created or updated will contain the same value for the property. Choose List Value when the source has no value for the property, or when you do not wish to use its value.
String	None	See Boolean property for description.
	Concatenate	String multiple values together to form a single value for the property. Select one or more columns from your source to use in the string and combine them with your own text values. For example, the Asset Name property could be formed by concatenating two columns and a text string: Asset Tag + " - " + Model. The Asset Name would then look something like: 10074589 - Lenovo ThinkPad 520.
	Copy	See Decimal property for description.

TABLE 1. Property Types and their Actions (Continued)

Property Type	Action	Description
String (cont'd)	Split	Use split to extract a string from the source using the position of its segment and a delimiter. For example, if the source contained the value sn-xyz-001, you could extract any of the three segments using the hyphen delimiter to distinguish between segments. In this case, entering the delimiter as - and the position as 3, would yield a result of 001. Changing the position to 2, would yield the value of xyz.
	Substring	Use substring to extract a string from the source using character start and end positions. For example, if the source contained the value Model T450 Lenovo ThinkPad, you could extract a portion of that text to use as the value. If the start position was 7 and the end position was 26, then the result would be T450 Lenovo ThinkPad.
	Value	See Decimal property for description.

When the source differs from the destination

Because you can change the source data as its being imported, initially it does not need to be valid for the destination object. For example, when you are importing data into a list, if the values in your source data are valid for that list, then data mapping is simplified. However, if your data has consistent values for a list, but they are not identical to the predefined values of the list, then you need to transform the data as it is being imported. You must associate each predefined value with a value from the source (see Figure). In this situation it is important that your source be consistent in its use of the values. This way you can expect consistent results when you are mapping a data source to the properties of an object. For example, if you are importing data into the Type property of an organization, one predefined value for Type is Department. If your data contained Dept., or even an obscure value such as 2, then you could always transform either of these to the value Department. Use the List Lookup action to transform list values.



NOTE: *When your source data exceeds the maximum length of a property, it is truncated. The trailing characters outside the permitted length are removed during the import. The data does not appear truncated on the form, even when it is saved.*

How to import data

By planning your import jobs, you can reduce the number of reimport attempts that you have to make. Choose a method for importing your data based on the following questions: Is the data in csv format? Is the data in a SQL Server database? Does the data originate from an LDAP (Lightweight Directory Access Protocol) source, such as Active Directory? Do you expect imports to occur on a scheduled basis? Scheduled imports are performed with the PowerShell cmdlet.

Familiarize yourself with the data being imported so that you can make the right decisions when mapping your data to properties. Are there any missing columns that are required? Are the headers missing? You will need to either add missing fields or skip them during the import. Do you need to change the data to make it valid? Are you going to create new records, update existing records, or do both?

Permissions required

Any import into Service Manager should be performed by someone who belongs to the Service Manager Administrators role. Ensure that your account has the Administrators role before performing an import. Alternatively, if it does not have this role and you attempt an import, a Run As Account (Data Management Administration Account) is automatically substituted. The Run As Account may perform an import, providing it too is configured with a user who has the Administrators role. Refer to the Deployment Guide for more details and instructions on using a Run As Account.

Importing a csv file

A csv file is one method of importing data using the Data Management Pack. Unlike the csv file used by SCCM's import csv wizard, a csv file used by Data Management does not need to be accompanied by an XML file.

Prerequisites for a csv file

The prerequisites for importing data from a csv file are as follows:

- Close the file if it is open in another application
- Ensure the file uses a comma as the delimiter
- Ensure the file has headers. Headers are used when mapping source content to properties. The header is the title for the column of data and is what you use throughout the wizard when selecting source data. If your source file does not have headers, then the first row of the file is used, which means you will not be able to import the first row. Similarly, the headers should correspond to the data so that when you are mapping the data you will know what you are mapping. For example, if you are importing a hardware asset, then one of the headers would be Life Cycle Status. Likewise, the content of that column should have appropriate states for life cycle. They need not be identical to the states required, but they need to be relevant to life cycle so that when you are mapping your data to the hardware asset you will know how to map it to the predefined states.

A preview of the csv file's first five records is available during the import, so you can ensure that it has been set up properly.

To import data from a csv file:

1. Go to the Data Management workspace > Import view.
2. In the Tasks pane, click **Import CSV File**. The Import Wizard opens.
3. Choose a Template Definition. Even if you do not wish to create a new template, you can choose not to save the template at the end of the wizard.
4. Select an **Error Handling** option to determine whether the wizard should proceed or not when it encounters an error. For more details on error handling, see "Error handling options" on page 35.
5. Click **Next**. The Select File page opens.
6. Click **Browse** to find the file to import.
7. Click **Next**. The Source Preview page opens. The preview lets you determine whether the source file and its layout are as expected. Click **Previous** if you need to select another file.
8. Click **Next** to begin mapping the data to an object and its properties.
9. Continue with "Creating and configuring objects" on page 22.

Importing SQL Server data

The Data Management Pack can import data through a query to a Microsoft SQL Server database.

Prerequisites for a SQL Server import

The prerequisites for importing data from a Microsoft SQL Server database are as follows:

- Supports Microsoft SQL Server, 2003 or higher.
- Know the name of your SQL Server Database. While the wizard returns a list of SQL servers on your network, it may not find every server.
- The management pack uses Windows authentication for the database connection. It does not connect with the credentials you used to log into the Service Manager console. Instead it connects with the credentials you used to log into the computer (either the management server or your local computer if you are using the remote console). Your Windows account must have permission to execute queries against the database.

WARNING: There are no restrictions on the type of query that you can execute on a database. All queries are passed to the server untouched, even those that include delete actions.

To import data from a SQL Server query:

1. Go to the Data Management workspace > Import view.
2. In the Tasks pane, click **Import SQL Server Data**. The Import Wizard opens.
3. Choose a Template Definition. Even if you do not wish to create a new template, you can choose not to save the template at the end of the wizard.
4. Select an **Error Handling** option to determine whether the wizard should proceed or not when it encounters an error. For more details on error handling, see "Error handling options" on page 35.
5. Click **Next**. The Enter Database Information & Query page opens.
6. Select the **Database server name** from the list of database servers found on the network. If a database server is new, it may not appear in the list. If your database server does not appear on the list, enter the name manually.
7. Select the **Database name** from the list of databases found on the server or enter the name manually.
8. Click **Test Connection** to ensure that the management pack can connect to the database.
9. Enter the SQL Server query in the **Query Definition** field. The SQL query is saved with the template if you choose to save it.
10. Click **Next**. The Source Preview page opens. The preview lets you determine whether the query and connection details return the results you expect. Click **Previous** if you need to change the database or your query.
11. Click **Next** to map the data retrieved from the database to an object and its properties.
12. Continue with "Creating and configuring objects" on page 22.

Importing through LDAP

You can import data that resides in an Active Directory domain or an organization unit (OU). For example, you could import users, printers or computers. As with other import methods, this data can be mapped to objects in Service Manager.

To import data from an Active Directory domain or OU:

1. Go to the Data Management workspace > Import view.
2. In the Tasks pane, click **Import LDAP Data**. The Import wizard opens at the Choose Template & Error Handling page.
3. Choose a Template Definition. Even if you do not wish to create a new template, you can choose not to save the template at the end of the wizard.
4. Select an **Error Handling** option to determine whether the wizard should proceed or not when it encounters an error. For more details on error handling, see "Error handling options" on page 35.
5. Click **Next**. The Select Domain/OU/Object Type page opens.
6. Select the domain or Organizational Unit that the wizard will connect to.
7. Select the type of object that you wish to import.
8. Under Credentials, select the name of the account that will be used to import the LDAP data.
9. It may be necessary to enter credentials before you can import these objects. Click **Details..** or enter the User Name and Password when prompted.
10. Click **Next**. The Source Preview page opens. The preview lets you determine whether the results you expect are returned. Click **Previous** if you need to change the connection information or the data being retrieved.
11. Click **Next** to map the data retrieved from the directory to an object and its properties.
12. Continue with "Creating and configuring objects" on page 22.

Importing through PowerShell

A PowerShell cmdlet (import - servicemanagerdata) is included with Data Management. The cmdlet allows you to import data from a command line into Service Manager objects. Using PowerShell 2.0 you can schedule data imports into Service Manager from other systems. You must understand how to use PowerShell and have an existing import template stored in Data Management that can be used to map the import data.

The form of import chosen — csv or SQL Server — is determined by the parameters included in the command line call (parameter set). The default parameter set is for a csv import. If no parameter set is specified, PowerShell requests values for the mandatory csv import parameters.

To run the PowerShell cmdlet:

1. On the Service Manager management server where you installed the Data Management pack, click the Start menu and search for the Provance folder.
2. Open the Provance folder and click **Provance Data Management Shell**. A command prompt opens with the cmdlet loaded.
3. Enter the import-servicemanagerdata command and its parameters to run the cmdlet. For more details, see "Sample cmdlet calls" on page 20.

NOTE: *If you wish to run the cmdlet from a command prompt without directly accessing the management server, it needs to be loaded first, see "To load the PowerShell cmdlet:" on page 17.*

To load the PowerShell cmdlet:

If you do not run the cmdlet from the Start menu, it needs to be loaded before you can use it. When loaded in this fashion the cmdlet does not persist between sessions; it must be re-loaded for use in every new session.

- Place the module folder \Program Files\Microsoft System Center\Service Manager 2012\Provance\Provance.ServiceManager.Integration.Cmdlets, within one of the PowerShell module paths and run Import-Module Provance.ServiceManager.Integration.Cmdlets. The environment variable psmodulepath contains the paths for PowerShell.

  OR

- Place the module folder Provance.ServiceManager.Integration.Cmdlets where you wish, take note of the path from the root directory, and run Import-Module "{your path}\Provance.ServiceManager.Integration.Cmdlets

Parameters used by the cmdlet

The following table contains a list of the parameters used by the cmdlet. It includes those specific to importing data, as well as some PowerShell general parameters that have been implemented. Parameters can also be placed into a separate source, such as a csv file, that can be piped into the cmdlet. Some examples of using these parameters are provided in "Sample cmdlet calls" on page 20.

Parameter	Type	Description
Name	import-specific	(mandatory) The name of the existing template (in Data Management) to be used during the import. Unnamed parameter, position 1. **NOTE:** *This parameter does not accept values piped to it from a file.*
OnErrorAction	import-specific	The action to take when an error occurs while processing data. The values are Abort and Skip. If an invalid value is used, the script uses the default value Abort. (This does not override or in any way relate to the PowerShell default parameter -ErrorAction). Unnamed parameter, position 2. **NOTE:** *This parameter does not accept values piped to it from a file.*
LiteralPath	csv import	(mandatory) The full path, from the root directory, to the source csv file, including the file name.
SqlServer	SQL Server import	(mandatory) The server where the source SQL database is installed. The server must be accessible, with the current user credentials, from the computer where you are running the script.
SqlDatabase	SQL Server import	(mandatory) The name of the source SQL database. The database must be accessible from the computer where you are running the script, with the current user credentials.
Sqlquery	SQL Server import	(mandatory) The query that retrieves your source data from the database. If the query includes escape characters, enclose them in single quotation marks. Single quotation marks indicate that Windows PowerShell should not interpret the characters as an escape sequence.

Parameter	Type	Description
Verbose	PowerShell	Triggers the display of the WriteVerbose messages included in the cmdlet. The progress of the cmdlet is displayed as follows: start time, end time, records read, entities created, and entities updated.
Debug	PowerShell	Triggers the display of the WriteDebug messages included in the cmdlet. It provides you with more information about the values returned from service calls.
ErrorAction	PowerShell	Controls how PowerShell handles WriteError messages from the cmdlet. It can have any of the following values: **Continue** - PowerShell notifies you when an error occurs and continues running the cmdlet. **Inquire** - PowerShell pauses the cmdlet and asks you how it should proceed. **SilentlyContinue** - PowerShell continues running the cmdlet without notifying you of an error. **Stop** - PowerShell treats the error like a TerminatingError and stops processing the cmdlet. **NOTE:** *This parameter does not accept values piped to it from a file.*
ErrorVariable	PowerShell	Provides a user defined variable in which to store the WriteError messages, rather than the default $error variable. If the variable name is prefixed with a +, then the messages accumulate, otherwise the variable only holds the most recent error message. The syntax is as follows: `-ErrorVariable myErr` or `-ErrorVariable +myErr`, where myErr is the user variable.

Sample cmdlet calls

In the following samples, the template name is 'psimport'.

- A basic call to the cmdlet, explicitly specifying the parameter names:

  import-servicemanagerdata -name psimport -literalpath

  "c:\Users\username\documents\test5.csv"

- A basic call to the cmdlet using positional call for template name:

  import-servicemanagerdata psimport -literalpath

  "c:\Users\username\documents\test3.csv"

- A call to the cmdlet with verbose messages turned on:

  import-servicemanagerdata -verbose -name psimport -literalpath

  "c:\Users\username\documents\test5.csv"

- A call to the cmdlet with user controlled error handling, where the user variable is 'myerr':

  import-servicemanagerdata -verbose psimport -literalpath

  "c:\Users\username\documents\test3.csv" -erroraction Inquire -errorvariable +myerr

- Piping the parameters in a file to the cmdlet:

  Import-csv "C:\SqlQuery.csv" | import-servicemanagerdata -name sqlimport

  -onerroraction skip

Importing through a template

An import template is created or modified during the process of importing data. A template can be saved separately to facilitate repeated imports, and then used to run an import job. Ensure that the path to the source data referred to in the template is valid before you begin. For more details about templates, see "Using import templates" on page 36.

To run an import job from a template:

1. Go to the Data Management workspace > All Import Templates view.
2. Select your template in the list.
3. In the Tasks pane, click **Run Job From Template**. A message appears indicating that the import has begun.
4. To see the status of the import, go to the **All Import Jobs** view and look up the template used for the import.
5. To see if there are any errors associated with the job, select the job and click **View Details**. The Processing Details tab contains a summary of the results.

Creating and configuring objects

Each object you create will have its own set of properties to configure. Properties determine what type, or class, of object is created, how its unique properties map to your source data, and even whether a new object instance is created or an existing instance is updated. If you are updating objects, you must enter criteria for finding instances to update. Once all your object classes are identified in the import wizard—either through a Create, an Update, or as a Reference—then you can add relationships between them.

To create and configure an object:

1. On the Add Objects & Relationship page, click **Add Object**. The Add Destination Object wizard opens.
2. Use each page of the wizard to configure the object: General Properties, Map Properties, and Criteria.

Configuring general properties

To configure the properties on the General page of the Add Destination Object wizard:

1. Select the **Destination Object Class**. This is the object (class) that will be created in the CMDB. Choose an object class that corresponds to the source data; for example, if you are importing software license contractual data into the IT Asset Management Pack, choose the correct agreement, such as Software License Agreement.
2. Enter a **Destination Object Name**. The Name is only used to help you identify objects within the wizard. It does not appear in the object instances that are created after the import.

3. Choose one of the **Object Processing Options** to determine whether you wish the objects to be created, updated, or both depending on certain criteria. The criteria must be defined on the Criteria page.

Option	Mapping	Criteria	Description
Create only	Mandatory	Optional	Creates all instances of the object. If you are certain that the instances do not already exist in the database, criteria are unnecessary and could slow the import down. If there is a chance that the source data could result in duplicate instances, define criteria to eliminate them.
Create or update	Mandatory	Mandatory	Updates all instances of the object that meet the criteria. Creates all instances of the object that do not meet the criteria. The criteria are therefore used to filter on the instances that you wish to update.
Update only	Mandatory	Mandatory	Updates all instances of the object that meet the criteria. Ignores all instances of the object that do not meet the criteria.
Reference Object	None	None	Use only when removing a relationship. This option indicates that any item can be removed from the relationship. In a one to one relationship, the instance of the reference object is removed. In a one to many relationship, all instances of the reference object are removed. Mapping and criteria are not permitted with this option.

Option	Mapping	Criteria	Description
Reference Instance	None	Mandatory	Use when creating, updating or removing a relationship. It is designed to identify a specific instance of an object in a relationship. No changes are performed on the instance itself; however, the instance must already exist or be created during the import before you use this option. Mapping is not permitted, but criteria are required to identify the instance. For example, if you wish to create the **Assigned to** relationship on a hardware asset, then you would need to identify the specific user on the object. The criteria for the Reference Instance would include Alison Kraus to ensure that she becomes the **Assigned to** in the relationship.
Delete		Mandatory	Deletes all object instances that meet the criteria. Instances deleted using this processing option are actually moved to the Deleted Items folder and must be permanently removed from there. Relationships are also removed from deleted instances. If you decide to restore an instance from the Deleted Items folder, its relationships are returned intact.

NOTE: *If you select either the **Update** or the **Create or Update** option and you do not define any Criteria, then nothing will be updated.*

4. (Optional) If you do not want object instances to be created, based on the appearance of certain values in your source data, select the **Source Column** having those values.
5. Add one or more values from the source column.

Configuring map properties

Your source content can be mapped to all, some or even none of the properties from a destination object. You may choose not to map any property when updating objects, or when creating relationships between objects.

To configure the properties on the Mapping page:

1. Go to the Mapping page. All the properties of the class you chose on the General page are displayed in the **Destination** column.
2. For each property in the Destination column, choose an **Action** (see Figure 1). The Action identifies how to map the source to the destination. All of the actions for mapping are described in "Property types and actions for them" on page 7.

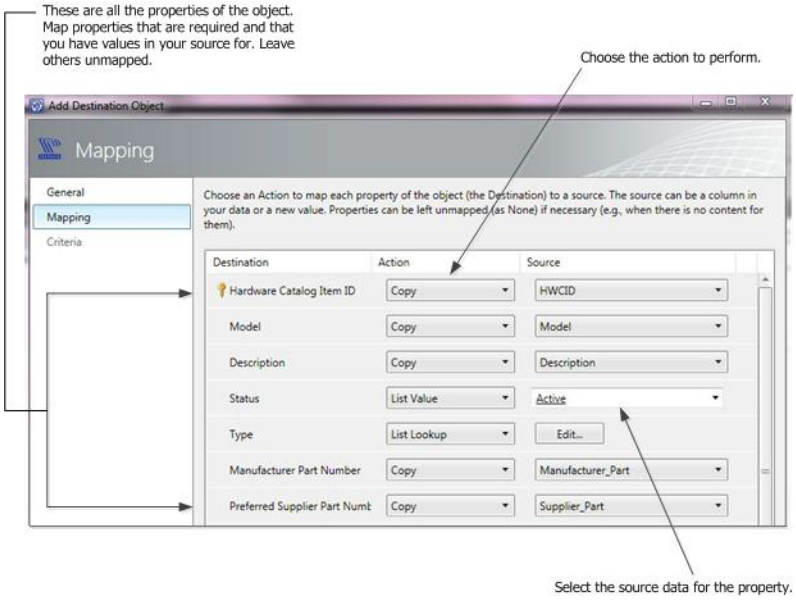


Figure 1: **Mapping source data to an object's properties**

3. In the Source column, select an item or enter a value based on the Action chosen in step 2.

4. If you are updating existing records, proceed to the Criteria page, otherwise click **OK** to complete the configuration of the object or relationship.

NOTE: *When importing hardware assets, the Asset Name field on the asset form is entitled "Display Name" on the Mapping page.*

Configuring Criteria

The Criteria page provides search criteria during an import. It can also identify a particular instance of an object when working with relationships, e.g., the T510 laptop computer whose serial number is 11178430.

During an import, instances that meet the criteria are updated (when the **Update Only** Object Processing option is chosen). If you have chosen the **Create or Update** Object Processing option on the General page, then instances that do not meet the criteria are instead created. In this case, you may use criteria to prevent duplicate instances from being created.

In a relationship, an object instance that meets the criteria is either added to the relationship or removed from the relationship. No changes are performed on the instance itself. If no instance meets the criteria, then nothing is added or removed from the relationship.

To configure the options on the Criteria page:

1. Import the source data as described in "To import data from a csv file:" on page 13 or "To import data from a SQL Server query:" on page 14.
2. Map the source data to the destination objects as described in "Configuring map properties" on page 25.
3. Go to the **Criteria** page, select a property and click **Add**. The property is added to the criteria field. Select as many properties as necessary to narrow the scope of the search.
4. From the drop down adjacent to each property, select the corresponding column in the source data. These properties and their matching columns define which object instances are updated or created and which instances are added to a relationship or removed from one, depending on the Object Processing Options selected on the General page.
5. Click **OK**. The object is now configured.

The next step is to create relationships between your objects. See "Creating relationships between objects" on page 27. If you are ready to create the objects, see "Completing the import wizard" on page 34.

Creating relationships between objects

You can establish relationships between objects that you are creating or updating from your source. However, before you create relationships, all objects in the relationship must first be defined, see "Creating and configuring objects" on page 22. You can select the objects for the relationship in any order; the import wizard determines which object should be the parent and which should be the child.

NOTE: *Refer to the Provance Class Definitions document to understand the number of objects possible in any given relationship.*

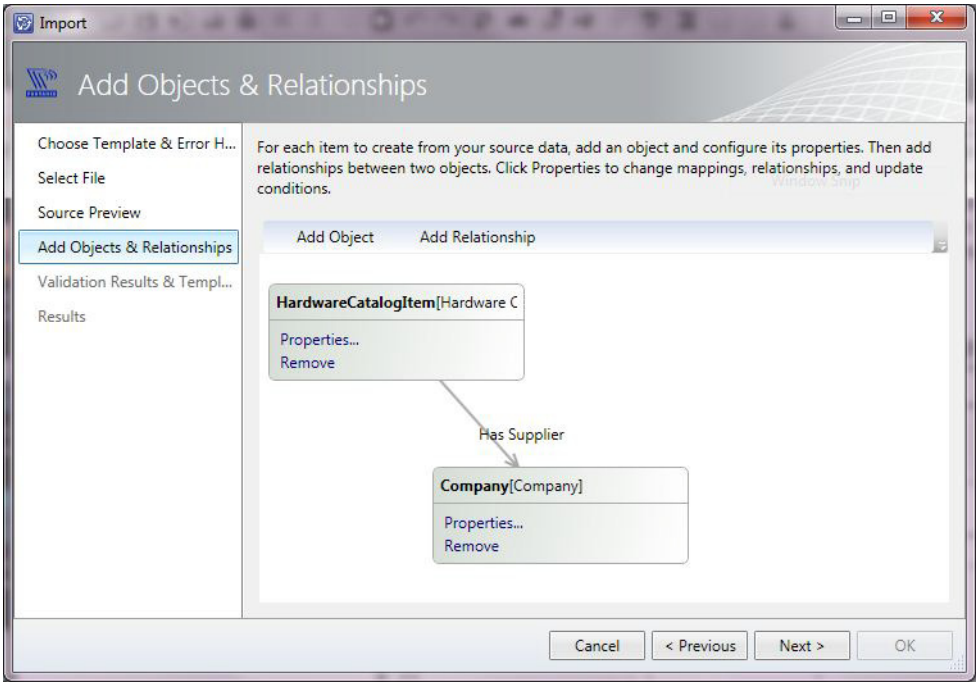
To create a relationship between objects in your data:

>> On the Add Objects and Relationships page, click **Add Relationship**. The Add Relationship form opens. Configure the relationship as described in "Configuring general properties" on page 28, "Configuring map properties" on page 31, and "How to create a relationship" on page 32.

Configuring general properties

1. On the Add Relationship form, select an Object in the relationship.
2. Select a second object in the relationship. Select the objects in any order.
3. Select the type of relationship by choosing a class. The available classes changes depending on the objects selected. Choose Provance classes over other classes as they are more likely to represent an accurate relationship for the objects.

4. Enter a name for the relationship. This helps you identify it while working in the wizard, as with the following relationship between a hardware catalog item and the supplier that it is related to:



5. Choose a Relationship Processing Option to determine how the relationship should be processed:

Create Only	Create a relationship between the two objects. The objects must exist and be identified with the Object Processing Options of Create, Update or Reference Instance. In a one to one relationship, the create only functions when there is no existing relationship. For example, if a hardware asset already has an Assigned to, then no additional Assigned to relationship would be permitted because it can only have one. In a one to many relationship, any new relationship is created, but only if that relationship did not previously exist. For example, there could be multiple cost events added to a hardware asset through a Create Only, but if one of those cost events previously existed then it would not be added again to the asset. For more details, see "How to create a relationship" on page 32.
Create or Update	Create a relationship between the two objects or change an existing relationship. The objects must exist and be identified with the Object Processing Options of Create, Update, or Reference Instance. In order to perform a create, omit the criteria.When creating a one to one relationship, any existing relationship is replaced. In all other situations, a new relationship is created. For example, if a hardware asset already has an **Assigned to**, a different user would replace the existing assignment. Updates are only performed on relationships with properties. When a relationship has properties, they are visible on the Mapping page of the wizard. You must enter criteria for it to find the relationship. If the relationship matches the criteria, then its properties are updated. If a match is not found then the relationship itself is created. For more details, see "How to create a relationship" on page 32.

Update Only	An Update Only is used when you wish to modify the properties of a relationship, such as the **type** of Software Manager on a software title. Updates are only performed when the source and target objects are found through the Criteria defined for them (Destination Object wizard). If the source and target objects in the relationship cannot be found, then nothing is updated. An update can be performed in a one to one relationship or in a one to many relationship. The properties of the relationship are defined on the Criteria page of the Relationship wizard. **NOTE:** When there is no existing relationship, no update is performed because it is considered a create. The relationship must have properties associated with it in order for the update to proceed. The objects must be defined and identified with the Object Processing Options of Create, Update, or Reference Instance.
Delete	Remove all relationships of the type chosen when used in conjunction with the Object Processing Option: Reference Object. Remove only the relationship that matches the criteria defined with the Object Processing Option: Reference Instance. The objects themselves are retained. For more details, see "How to remove (delete) a relationship" on page 32.

Configuring map properties

Sometimes relationships have properties associated with them. When there are properties associated with a relationship, then you may map them to data in your source file. When you use **Update Only** as the Relationship Processing option, the properties of a relationship are considered as the update.

NOTE: *When there are no properties associated with the relationship, the Mapping page has no content.*

1. If there are properties associated with the relationship, map them to your data. Follow the steps for mapping properties of an object, described under "Configuring map properties" on page 25.
2. If there are no properties listed, click **OK** to save the relationship.
3. Add additional relationships as required.

How to create a relationship

Are you concerned about overwriting an existing relationship? In a one to one relationship you may not wish to overwrite the existing relationship. The best way to discover whether there is one or many objects in a relationship is to attempt to add more than one through the form; if you can add more than one, then it is a one to many relationship. If you cannot add more than one, then it is a one to one relationship.

Yes I am concerned: You may not wish to add a relationship when one already exists or when a specific object instance is in the relationship, e.g., the supplier Adobe should not be removed. In either case, use the **Reference Instance** Object Processing Option. Reference Instance lets you add criteria for finding the object instance. The advantage of using Reference Instance over the Update option, which could also be used, is that there is no update needed to the second object—you are simply relating them. On the relationship, choose **Create or Update** as the Relationship Processing Option.

Example: You wish to assign Wil Smith as the user to a hardware asset, but only if there is no one already assigned. There can only be one assignment on a hardware asset. You would add Wil Smith as the Reference Instance criteria for the user object in the relationship.

No I am not concerned:In a one to many relationship it may not matter to you if another relationship is added. In a one to one relationship, it also may not matter to you if the existing relationship is overwritten. Choose **Reference Instance** as the Object Processing Option and choose **Create or Update** as the Relationship Processing Option.

How to remove (delete) a relationship

Are you concerned about removing the wrong object from the relationship? If you wish to remove a specific item from the relationship, then you need to add criteria to search for an instance of the object. If you are not concerned about what is removed, then you can remove any item from the relationship.

Deleting an instance of an object (actually marked as pending delete in the console) removes any relationships it has with other objects.

Yes I am concerned:Use the **Delete** Relationship Processing Option and the **Reference Instance** Object Processing Option so you can add criteria for finding a specific instance of the object. In this case the relationship may be one to many or you may only wish to remove the relationship when it meets specific criteria.

Example: You only wish to remove the assignment from a hardware asset if the assignment is to Angelica Hudson. The criteria for the second object would be Angelica Hudson.

No I am not concerned:Use the **Delete** Relationship Processing Option and the **Reference Object** Object Processing Option because you do not need to select any search criteria. Reference Object removes all instances of objects from the relationship, even if it is a one to many relationship.

Examples: You wish to remove an assignment from a hardware asset and it does not matter who the assignment is to. You wish to remove **all** cost events from a hardware asset.

Completing the import wizard

You have created objects and mapped them to your source data. You have created relationships between objects. Now it is time to complete the import by saving the data map as a template and processing all of the instructions you gave during the configuration.

1. When you have added all of the objects and relationships, click **Next**. The Validation Results & Template Creation page opens. The page contains results from a validation performed on your source file. If there are no errors to report, then nothing is shown. The kinds of errors that can be reported include the following: added, deleted or renamed column; too many or too few columns on a line; and mapping issues that need attention.

   NOTES: *If you receive the error message "A template cannot be created without at least one mapping in one object," the most common reason is that you have not mapped some of the values in the template. Check the data map of each object to determine where the values have been missed (usually identified for you with an asterisk). This can happen when you have transferred a template between systems and a value does not exist or the GUIDs do not match between the systems.*

   If you receive the error message "Template contains invalid definition", the source file is inconsistent with the template chosen. Inconsistencies include missing columns and renamed columns. Review the object's properties for items in red which are considered inconsistent with the template's definition.

2. To save the data map as a template, click **Save Template**. Only Service Manager Administrators and Authors can save templates. Enter a Name and a Description that will help you choose it from a list of other templates.
3. To continue with the import, click **Import**. The Results page opens with a summary of the import process. The **Records processed** field tells you how many records were actually processed before any errors occurred. It does not tell you how many records were created. The All Import Jobs view contains a list of objects created and objects updated by each job.
4. To cancel the import before it begins, click **Close**. If you have selected the Save Template option and given it a name, the template is still saved.
5. To cancel the import while it is in progress, click **Cancel**. For more details on the cancel operation, see "Cancelling an import" on page 35.

Cancelling an import

When you click **Cancel** while the import is in progress, processing on the current record completes before the import is cancelled. Records imported prior to the cancel are committed, i.e., they are not deleted or rolled back.

Error handling options

The options for error handling are as follows:

Error option	Description
Import records until an error occurs, then abort	The import stops on the first error and does not continue. The records processed before the error occurred are imported, while all the remaining records are not imported. However, there may be situations where no records are imported because processing was not complete when the error occurred. Even with this option selected, if an error occurs while committing data to the database, objects created from the current row in your data may be saved to the database. The remainder of the file will be aborted. In this situation, the actual number of objects created or updated will be less than reported in the Import Jobs view.
Skip records with errors and continue import	All records without errors are imported; records with errors are skipped.

Using import templates

A data map describes how the columns in your source data are mapped to the properties of the object. A data map can be saved as an XML template to facilitate repetitive import tasks, such as imports every month from a financial system. A template is created or modified during the process of importing data. A template created and saved during a csv import can be reused in a SQL Server import, and vice versa, as long as the headers match the data being imported. A template also includes details about the source, such as the SQL Server connection, the SQL query entered, the source CSV filename and LDAP parameters. Even if you modify a template for csv that was originally intended for a SQL Server import, the template will remember its last SQL Server connection and query details. Since a template contains all the information required to perform an import, you can run an import directly from any template (All Import Templates view).

Only individuals with Service Manager roles of Author and Administrator can save templates during the import process.

One use for templates is to create different objects from the same data. Even though your source data may contain columns that are applicable to multiple objects, such as a hardware asset, a cost center and a location, use a template to create one type of object from that data. For example, one template can create a cost center object from your data. Another template could create a hardware asset. Both templates would create their objects from the same source data.

NOTES:

The Last modified time in the All templates view is not the server time. It is the current GMT.

If you modify a csv after creating a template from it, such as changing a header or removing a column, an error is generated during the next import of the csv. You can correct the error by remapping the affected property of the object you are creating. In the case of a renamed header, you would simply remap the property to the new header name.

Saving import templates as XML

Save an import template as XML so that it can be reused in another Service Manager console.

To save an import template for use with another console:

1. Create a template while importing data (as part of the Import wizard).
2. Go to the Data Management workspace > Import > View All Templates.
3. Select the template to save as XML.
4. Click **Save Template as XML**. The Save Template as XML form opens.
5. Click the **Browse** button to enter a filename and a path for the template.
6. Click **OK** to save the template. The template can be loaded into another console using the Load Import Template task.

Loading import templates

An import template, from version 1.2 or later, that is saved as XML can be loaded into another Service Manager console and used to import data. You can import a single template or multiple templates either through the task or the PowerShell cmdlet (Import-ProvanceDMPTemplate). Although the cmdlet is designed to import a single template, you can import a directory of files using the native piping ability of PowerShell (see "PowerShell sample calls" on page 40).

When they have identical names, an import template replaces an existing template.

To load import templates using the task:

1. Go to the Data Management workspace > Import view.
2. In the Tasks pane, click **Load Import Template**. The Load Import Template form opens.
3. Click the **Add Files** button to find and add one or more XML files to the list. When you load a template file that has the same name as an existing file a warning is given. You can choose to continue with the import or remove the duplicate file from the list.

4. Enter a Display Name and a Description. The name and description should provide enough information about the template so you can choose it easily in a list.
5. Click **OK** to load the template file or files into the console. Loaded templates are found in the **View All Templates** view.

To load an import template using a PowerShell script:

1. On the Service Manager management server where you installed the Data Management pack, click the **Start** menu and search for the Provance folder.
2. Open the Provance folder and click **Provance Data Management Shell**. A command prompt opens with the cmdlet loaded.
3. Enter the import command and its parameters to run the cmdlet, such as Import-ProvanceDMPTemplate -LiteralPath "c:\mypath\myfolder\MyTemplateDefinition.xml" -Name MyImportedTemplate -OnErrorAction skip. For more details, see "PowerShell sample calls" on page 40 and "PowerShell import template parameters" on page 39.

NOTE: *If you wish to run the cmdlet from a command prompt without directly accessing the management server, it needs to be loaded first, see "To load the PowerShell cmdlet:" on page 17.*

Editing import templates

A template is edited during the process of importing source data. When you change your CSV filename, SQL Server database connection information and query, and most importantly the property mappings between your source and the objects to create, you are changing the template. Saving all these changes to the existing template updates it for the next import.You do not need to import data in order to create or modify a template through an import wizard. When you reach the end of an import wizard, simply select **Save Template** and click **Close**.

PowerShell import template parameters

The following table contains a list of the parameters used by the cmdlet. It includes those specific to loading import templates. For a description of PowerShell general parameters that have been implemented, see parameters whose type is "PowerShell" in the table of "Parameters used by the cmdlet" on page 18. Parameters can also be placed into a separate source, such as a csv file, that can be piped into the cmdlet. Some examples of using import parameters are provided in "PowerShell sample calls" on page 40.

Parameter	Position	Description
-LiteralPath	0	(Mandatory) The full path from the root directory to the source template definition file, including the file name.
-Name	1	(Optional) The name to use when creating a new template, or to search for when updating an existing template. If the name is not supplied, the file name will be used instead.
-OnErrorAction	2	(Optional) The action to take if the import process encounters an error. The values that are understood include **Abort** and **Skip**. If an invalid value is used, or if a value is not provided, the cmdlet uses the default value of **Abort**. **Note:** This does not override or in any way relate to the PowerShell default parameter -ErrorAction.

PowerShell sample calls

- The following basic call to the cmdlet imports a single template definition file, explicitly specifying the optional parameters Name and OnErrorAction:

  ```
  Import-ProvanceDMPTemplate -LiteralPath "c:\mypath\myfolder\MyTemplateDefini-
  tion.xml" -Name MyImportedTemplate -OnErrorAction skip
  ```

- An advanced command set that attempts to import all the xml files in a specified folder path:

  ```
  Get-ChildItem -Path "C:\Users\myfolder\Documents\Exported Templates\*" -Include *.xml
  | Select-Object -ExpandProperty FullName | Import-ProvanceDMPTemplate -OnErrorAc-
  tion skip
  ```

- An advanced command set that attempts to import all the xml files in a specified folder path and explicitly sets the optional Name parameter to the filename. Although the filename to name conversion is the default action of the cmdlet, it shows the possibility of setting explicit names:

  ```
  Get-ChildItem -Path "C:\Users\myfolder\Documents\Exported Templates\*" -Include *.xml
  | % {Import-ProvanceDMPTemplate -LiteralPath $($_.FullName) -Name $($_.BaseName) -
  OnErrorAction skip}
  ```