Data Importer For SharePoint (DIFS)
Documentation

Ref	DIFS
Date	12/10/2016
Version	2.3

Document Control Sheet

Revision History			
Date	Change	Version	
12/06/2012	First release	1.0	
03/10/2016	Updated for Office 365 & SharePoint 2016	2.0	
06/10/2016	Added support for reporting to console for Powershell scripting. Added support for managed meta data tagging using CSOM – 2013/2016/Online	2.2	
12/10/2016	Added support for multi-value managed meta data columns. Add example for document set creation. Added support for installation on 32bit Windows.	2.3	

Referenced Sources

Table of Contents

1 Introduction

1.1 What is DIFS?

DIFs is an application for Microsoft Windows which allows you to import data and documents into SharePoint lists and document libraries.

1.2 Why DIFS?

There are a lot of migration tools available that import data into SharePoint and so why develop DIFS?

Many of the available migration tools for SharePoint have a high cost, restrictive licencing model and can be quite limited in their flexibility / ability to address real life scenarios that arise in import / migration projects.

DIFS is free, flexible and leverages other available technologies. It can import from file systems and 100's of OleDB providers. You may use tools like Excel and SQL to cleanse data prior to import and by clever definition of your data sources, SQL statements and configuration files you can achieve complex migration scenarios.

1.3 How does DIFS work?

1. You install DIFS on your PC.
2. You create a data source to import from such as an Excel spreadsheet or a SQL database table.
3. You create an XML based configuration file that tells DIFS how to carry out the import (such as the destination library, credentials and data mappings).
4. You run the import to transfer the data.

1.4 Used DIFS? Worked Great?

Please post your stories, experiences, and let us know of any improvements.

2 Key Features

Key Features	
SharePoint Versions	SharePoint Server SharePoint Foundation 2010,2013 and 2016 SharePoint Online
Authentication	Supports forms based authentication
Meta data	Allows existing file meta data to be imported
Import Control	Imports can be paused, resumed and cancelled. Import progress is reported to the use interface.
Exception handling	Exceptions can be saved for correction and reprocessing.
Save settings	Import settings can be saved and retained for future use.
Uses Client Object Model	You may run the software on the server or a client PC.

3 Installation

3.1 DIFS

Run the setup.exe or MSI that you downloaded.

3.2 OleDB Data Source

You will need to ensure that you have the correct 64bit OleDB provider for DIFS installed for the source that you wish to access.

3.2.1 Excel

Try

https://www.microsoft.com/en-gb/download/details.aspx?id=13255

or

https://www.google.co.uk/#q=excel+64+bit+oledb+

4 Examples

4.1 Import from an Excel Source into a SharePoint Online List

4.1.1 Overview

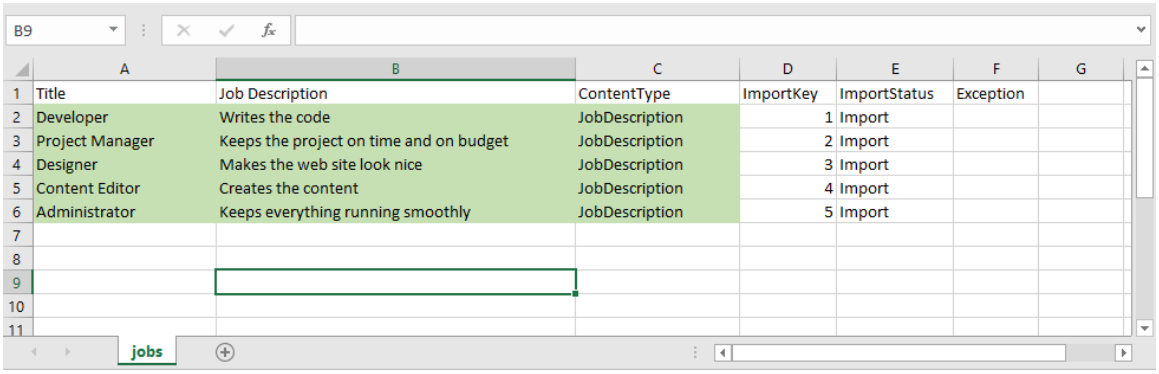
Import list items from an Excel spreadsheet source into a SharePoint Online (Office 365) list.

4.1.2 The Source

The source is an XLSX

Columns define the title, description, content type and control the import.

The worksheet is called “Jobs”.



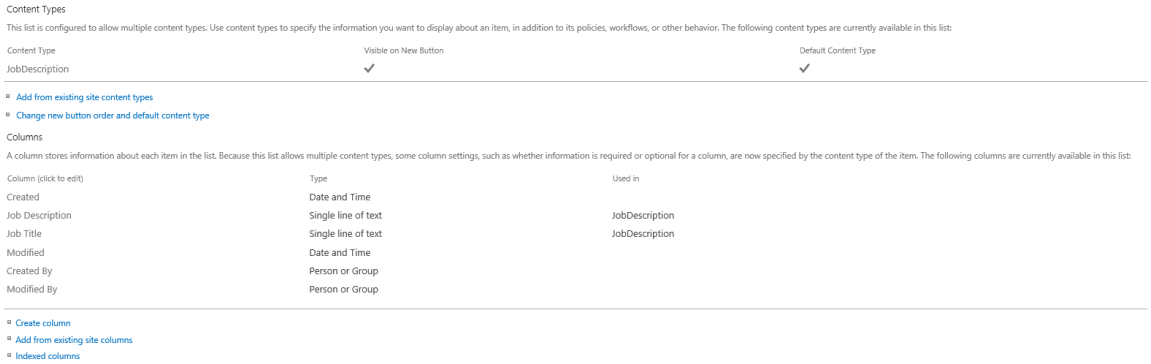
	A	B	C	D	E	F	G
1	Title	Job Description	ContentType	ImportKey	ImportStatus	Exception	
2	Developer	Writes the code	JobDescription	1	Import		
3	Project Manager	Keeps the project on time and on budget	JobDescription	2	Import		
4	Designer	Makes the web site look nice	JobDescription	3	Import		
5	Content Editor	Creates the content	JobDescription	4	Import		
6	Administrator	Keeps everything running smoothly	JobDescription	5	Import		
7							
8							
9							
10							
11							

4.1.3 The Configuration

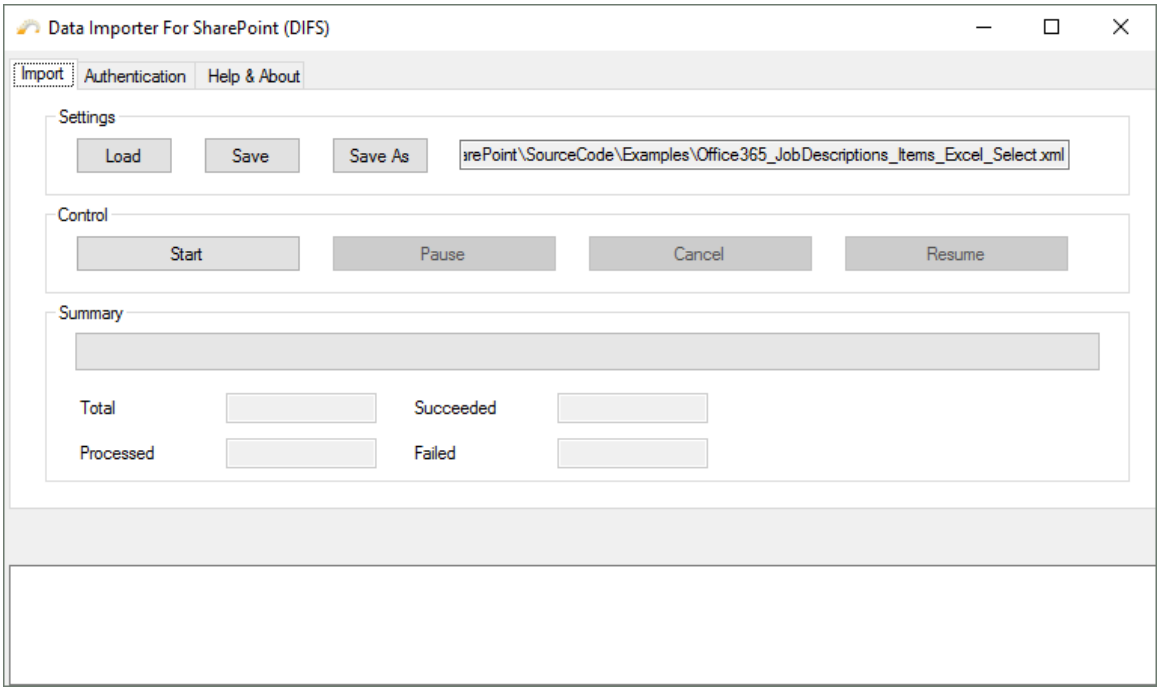
```
<?xml version="1.0" encoding="utf-8"?>
<DataSetImportSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Source>
    <!-- The source used is OLEDbSelect meaning that DIFS expected to run a SQL select statement against an OleDB data
source.  OleDB select will ignore columns like importstatus-->
    <SourceDataSetType>OLEDbSelect</SourceDataSetType>
    <!-- The Connection string uses a microsoft provider to access on Excel spreadsheet-->
    <OleDbSourceDataSetSettings>
      <ConnectionString>Provider=Microsoft.ACE.OLEDB.12.0;Data Source=\\vmware-host\Shared
Folders\Projects\ProductDevelopment\ImportForSharePoint\WorkingArea\Examples\JobDescriptions.xlsx;Extended Properties="Excel
12.0 Xml;HDR=YES;IMEX=0";</ConnectionString>
    </OleDbSourceDataSetSettings>
    <OleDbTableSourceDataSetSettings />
    <!-- The select statement will get all the jobs from the jobs worksheet -->
    <OleDbSelectSourceDataSetSettings>
      <SelectStatement>select * from [jobs$]</SelectStatement>
    </OleDbSelectSourceDataSetSettings>
  </Source>
  <Destination>
    <!-- The authentication type is Office365 i.e. online cloud sharepoint, you can load the configuration file to DIFS to
enter and save the credentials in encrypted format -->
    <AuthenticationSettings>
      <AuthenticationType>Office365</AuthenticationType>
      <domain />
      <username></username>
      <encryptedpassed></encryptedpassed>
    </AuthenticationSettings>
    <DestinationItemSettings>
      <!-- The destination you are tell DIFS to make is an item as opposed to a File or Folder -->
      <DestinationItemType>Item</DestinationItemType>
      <!-- If the item already exists then overwrite it -->
      <ItemExistsBehaviour>Overwrite</ItemExistsBehaviour>
      <!-- Map data from your source into SharePoint fields-->
```

```
      <ImportMappings>
        <!-- Map title to title assuming that it is a string -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Title</DestinationField>
          <SourceColumn>Title</SourceColumn>
        </ImportMapping>
        <!-- Map job description to job description assuming that it is a string -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Job Description</DestinationField>
          <SourceColumn>Job Description</SourceColumn>
        </ImportMapping>
      </ImportMappings>
    </DestinationItemSettings>
    <!-- Tell DIFS exactly where the list is -->
    <DestinationListSettings>
      <DestinationWebUrlRelative>/sites/SPImportHelper</DestinationWebUrlRelative>
      <DestinationFolderUrlRelative>/sites/SPImportHelper/Lists/Items</DestinationFolderUrlRelative>
      <DestinationServerUrl>https://company.sharepoint.com</DestinationServerUrl>
      <DestinationListName>Items</DestinationListName>
    </DestinationListSettings>
    <!-- Tell DIFS about the source you importing from-->
    <SourceColumns>
      <!-- The column in the source which contains the full path of the file being imported.  The value entered here is
ignored unless DestinationItemType is Document -->
      <SourceFileNameAndPath>FullName</SourceFileNameAndPath>
      <!-- The column in the source the value of which matches the content type in sharepoint to set on the item.  If you
are not using content types on the destination list you can enter an OOTB SharePoint content type such as Item, Document,
Folder -->
      <ContentType>ContentType</ContentType>
      <!-- The column in the source (if application) that contains the subfolder path to import to -->
      <DestinationSubFolder>DestinationSubDirectories</DestinationSubFolder>
      <!-- The column in the source which contains the destination file name. The value entered here is ignored unless
DestinationItemType is Document -->
      <DestinationFileName>DestinationFileName</DestinationFileName>
    </SourceColumns>
  </Destination>
</DataSetImportSettings>
```

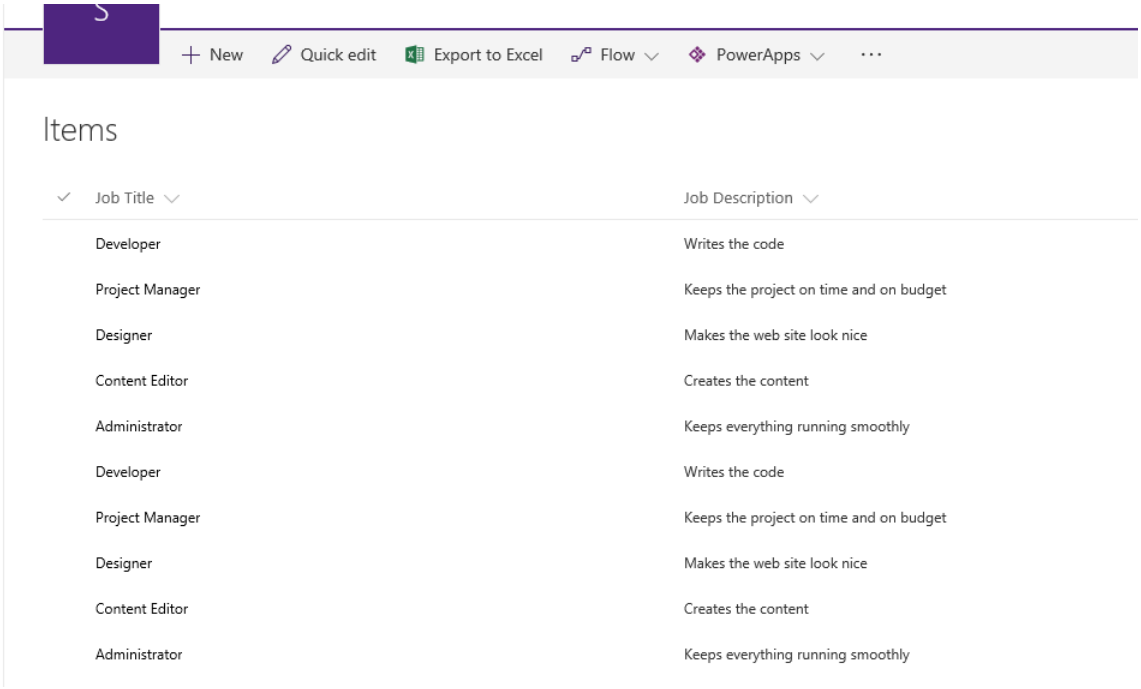
4.1.4 The Destination



4.1.5 The Execution



4.1.6 The Result



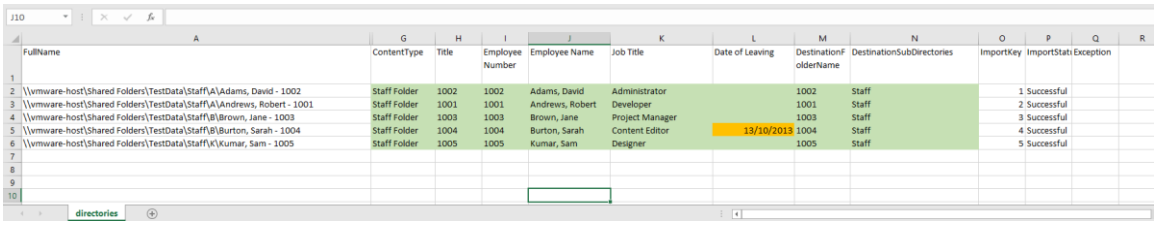
4.2 Create Folders from an Excel Source in a SharePoint Online Library adding meta data

4.2.1 Overview

Create a folder structure suitable for staff records.

Set meta title on the created folders including lookup fields and managed meta data.

4.2.2 The Source



	A	G	H	I	J	K	L	M	N	O	P	Q	R
1	FullName	ContentType	Title	Employee Number	Employee Name	Job Title	Date of Leaving	DestinationFolderName	DestinationSubDirectories	ImportKey	ImportStati	Exception	
2	\\vmware-host\Shared Folders\TestData\Staff\A\Adams, David - 1002	Staff Folder	1002	1002	Adams, David	Administrator		1002	Staff	1	Successful		
3	\\vmware-host\Shared Folders\TestData\Staff\A\Andrews, Robert - 1001	Staff Folder	1001	1001	Andrews, Robert	Developer		1001	Staff	2	Successful		
4	\\vmware-host\Shared Folders\TestData\Staff\B\Brown, Jane - 1003	Staff Folder	1003	1003	Brown, Jane	Project Manager		1003	Staff	3	Successful		
5	\\vmware-host\Shared Folders\TestData\Staff\B\Burton, Sarah - 1004	Staff Folder	1004	1004	Burton, Sarah	Content Editor	13/10/2013	1004	Staff	4	Successful		
6	\\vmware-host\Shared Folders\TestData\Staff\K\Kumar, Sam - 1005	Staff Folder	1005	1005	Kumar, Sam	Designer		1005	Staff	5	Successful		
7													
8													
9													
10													

directories

4.2.3 The Configuration

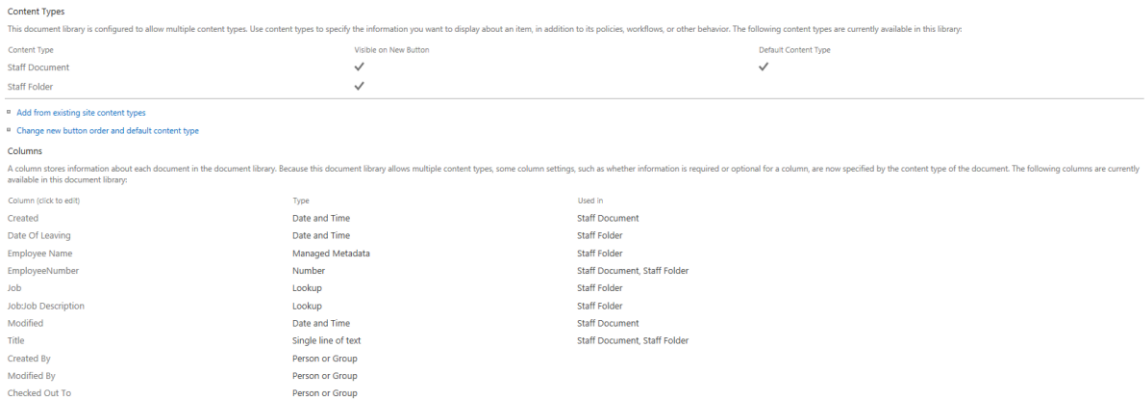
```
<?xml version="1.0" encoding="utf-8"?>
<DataSetImportSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Source>
    <!-- The source used is OLEDbSelect meaning that DIFS expected to run a SQL select statement against an OleDB data
source.  OleDB select will ignore columns like importstatus-->
    <SourceDataSetType>OLEDbSelect</SourceDataSetType>
    <OleDbSourceDataSetSettings>
      <!-- The Connection string uses a microsoft provider to access on Excel spreadsheet-->
      <ConnectionString>Provider=Microsoft.ACE.OLEDB.12.0;Data Source=\\vmware-host\Shared
Folders\Projects\ProductDevelopment\ImportForSharePoint\WorkingArea\Examples\StaffFolders.xlsx;Extended Properties="Excel
12.0 Xml;HDR=YES;IMEX=0";</ConnectionString>
    </OleDbSourceDataSetSettings>
    <OleDbSelectSourceDataSetSettings>
      <!-- The select statement will get all the directories from the directories worksheet -->
      <SelectStatement>select * from [directories$]</SelectStatement>
    </OleDbSelectSourceDataSetSettings>
  </Source>
  <Destination>
    <!-- The authentication type is Office365 i.e. online cloud sharepoint, you can load the configuration file to DIFS to
enter and save the credentials in encrypted format -->
    <AuthenticationSettings>
      <AuthenticationType>Office365</AuthenticationType>
      <domain />
      <username></username>
      <encryptedpassed></encryptedpassed>
    </AuthenticationSettings>
    <DestinationItemSettings>
      <!-- The DestinationItemType you are telling DIFS to make is a Folder as opposed to an Item or Document -->
      <DestinationItemType>Folder</DestinationItemType>
      <!-- If the item already exists then overwrite it -->
      <ItemExistsBehaviour>Overwrite</ItemExistsBehaviour>
      <!-- Map data from your source into SharePoint fields-->
      <ImportMappings>
```

```
        <!-- Map title to title assuming that it is a string -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Title</DestinationField>
          <SourceColumn>Title</SourceColumn>
        </ImportMapping>
        <!-- Map Employee Number to EmployeeNumber without conversion.  In this instance this works because both are numeric
fields-->
        <ImportMapping xsi:type="ImportMapping_Native">
          <DestinationField>EmployeeNumber</DestinationField>
          <SourceColumn>Employee Number</SourceColumn>
        </ImportMapping>
        <!-- Map Employee Name to Employee Name as ManagedMetaDataAutoAdd.  In this instance the value in the spreadsheet is
created as a managed meta data term and then the folder tagged with it -->
        <ImportMapping xsi:type="ImportMapping_ManagedMetaDataAutoAdd">
          <DestinationField>Employee Name</DestinationField>
          <SourceColumn>Employee Name</SourceColumn>
        </ImportMapping>
        <!-- Map Date of Leaving to DateOfLeaving converting from a string date.  In this instance the string is assumed to
be in UK date format-->
        <ImportMapping xsi:type="ImportMapping_DateTimeFromString">
          <DestinationField>DateOfLeaving</DestinationField>
          <SourceColumn>Date of Leaving</SourceColumn>
                    <!-- Refer to DateTime.ParseExact on MSDN for more information on ConversionMask (format) and culture
(iformat provider)-->
                    <ConversionMask>dd/MM/yyyy hh:mm:ss</ConversionMask>
                    <Culture>en-GB</Culture>
        </ImportMapping>
        <!-- Map Job Title to Job assuming that job is a look up field and the value of the column job title matches an
entry in it.-->
        <ImportMapping xsi:type="ImportMapping_Lookup">
                    <!-- This is the field of type lookup.-->
          <DestinationField>Job</DestinationField>
          <SourceColumn>Job Title</SourceColumn>
                    <!-- This is the list that contains the lookup values -->
                    <LookupListTitle>Items</LookupListTitle>
                    <!-- This is the field in the lookup list the value of which matches the value in the source column-->
                    <LookupFieldInternalName>Title</LookupFieldInternalName>
```

```
                    <!-- This is the CAML value type Text,Number,DateTime,Guid,MultiChoice,Lookup for the field on in the
lookup list that matches the value specified in the source column -->
                    <LookupFieldCAMLType>Text</LookupFieldCAMLType>
        </ImportMapping>
      </ImportMappings>
    </DestinationItemSettings>
    <!-- Tell DIFS exactly where the destination list or library is -->
    <DestinationListSettings>
      <DestinationWebUrlRelative>/sites/SPImportHelper</DestinationWebUrlRelative>
      <DestinationFolderUrlRelative>/sites/SPImportHelper/Docs</DestinationFolderUrlRelative>
      <DestinationServerUrl>https://company.sharepoint.com</DestinationServerUrl>
      <DestinationListName>Docs</DestinationListName>
    </DestinationListSettings>
    <!-- Tell DIFS about the source you importing from-->
    <SourceColumns>
      <!-- The column in the source which contains the full path of the file being imported.  The value entered here is
ignored unless DestinationItemType is Document -->
      <SourceFileNameAndPath>FullName</SourceFileNameAndPath>
      <!-- The column in the source the value of which matches the content type in sharepoint to set on the item.  If you
are not using content types on the destination list you can enter an OOTB SharePoint content type such as Item, Document,
Folder -->
      <ContentType>ContentType</ContentType>
      <!-- The column in the source (if applicable) that contains the subfolder path (If any) to import to -->
      <!-- This could be, for example, "Staff" or "Staff/Managers" or "Staff/Managers/Retired".  The folder must already
exist and can be created using DIFS -->
      <DestinationSubFolder>DestinationSubDirectories</DestinationSubFolder>
      <!-- The column in the source which contains the destination name. The value entered here is ignored unless
DestinationItemType is Document or Folder -->
      <DestinationFileName>DestinationFolderName</DestinationFileName>
    </SourceColumns>
  </Destination>
</DataSetImportSettings>
```

4.2.4 The Destination

4.2.4.1 Library



Content Types

This document library is configured to allow multiple content types. Use content types to specify the information you want to display about an item, in addition to its policies, workflows, or other behavior. The following content types are currently available in this library:

Content Type	Visible on New Button	Default Content Type
Staff Document	✓	✓
Staff Folder	✓	

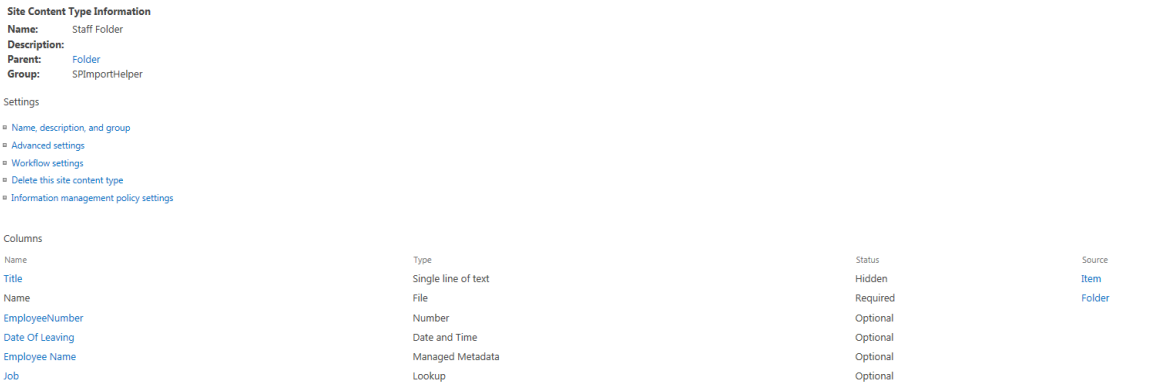
- Add from existing site content types
- Change new button order and default content type

Columns

A column stores information about each document in the document library. Because this document library allows multiple content types, some column settings, such as whether information is required or optional for a column, are now specified by the content type of the document. The following columns are currently available in this document library:

Column (click to edit)	Type	Used in
Created	Date and Time	Staff Document
Date Of Leaving	Date and Time	Staff Folder
Employee Name	Managed Metadata	Staff Folder
EmployeeNumber	Number	Staff Document, Staff Folder
Job	Lookup	Staff Folder
JobJob Description	Lookup	Staff Folder
Modified	Date and Time	Staff Document
Title	Single line of text	Staff Document, Staff Folder
Created By	Person or Group	
Modified By	Person or Group	
Checked Out To	Person or Group	

4.2.4.2 Content Type



Site Content Type Information

Name: Staff Folder
Description:
Parent: Folder
Group: SPImportHelper

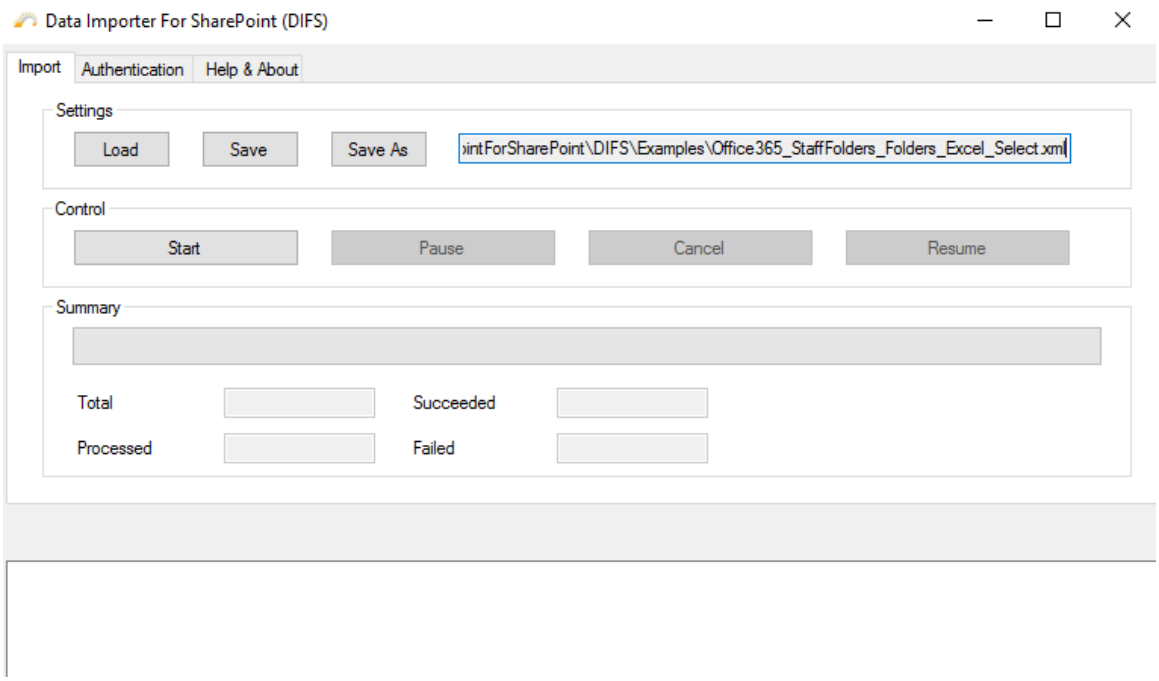
Settings

- Name, description, and group
- Advanced settings
- Workflow settings
- Delete this site content type
- Information management policy settings

Columns

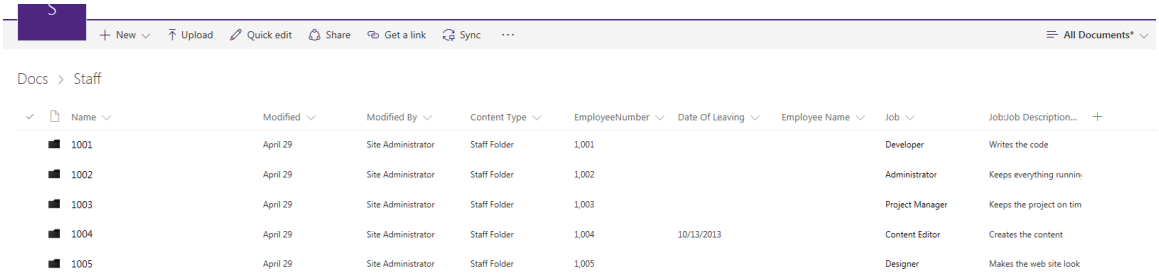
Name	Type	Status	Source
Title	Single line of text	Hidden	Item
Name	File	Required	Folder
EmployeeNumber	Number	Optional	
Date Of Leaving	Date and Time	Optional	
Employee Name	Managed Metadata	Optional	
Job	Lookup	Optional	

4.2.5 The Execution



4.2.6 The Result

Folders have been created and meta data fields have been set.



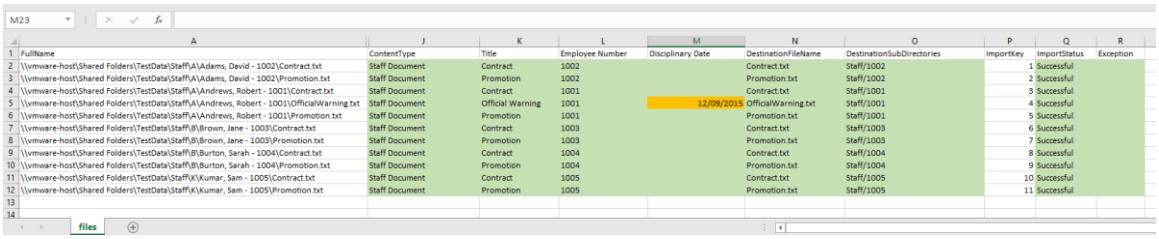
4.3 Import Documents from an Excel Source into a SharePoint Online Library

4.3.1 Overview

An Excel spreadsheet contains a list of documents, the path to each document and associated meta data.

DIFS imports the meta data from the spreadsheet and the associated document into a document library.

4.3.2 The Source



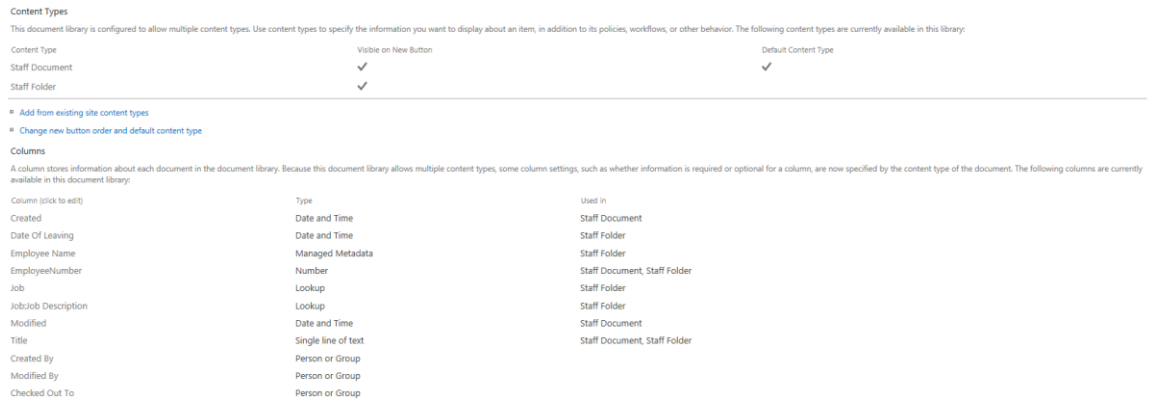
	A	J	K	L	M	N	O	P	Q	R
1	FullName	ContentType	Title	Employee Number	Disciplinary Date	DestinationFileName	DestinationSubDirectories	ImportKey	ImportStatus	Exception
2	\\vmware-host\Shared Folders\TestData\Staff\A\Adams, David - 1002\Contract.txt	Staff Document	Contract	1002		Contract.txt	Staff/1002	1	Successful	
3	\\vmware-host\Shared Folders\TestData\Staff\A\Adams, David - 1002\Promotion.txt	Staff Document	Promotion	1002		Promotion.txt	Staff/1002	2	Successful	
4	\\vmware-host\Shared Folders\TestData\Staff\A\Andrews, Robert - 1001\Contract.txt	Staff Document	Contract	1001		Contract.txt	Staff/1001	3	Successful	
5	\\vmware-host\Shared Folders\TestData\Staff\A\Andrews, Robert - 1001\OfficialWarning.txt	Staff Document	Official Warning	1001	12/09/2015	OfficialWarning.txt	Staff/1001	4	Successful	
6	\\vmware-host\Shared Folders\TestData\Staff\A\Andrews, Robert - 1001\Promotion.txt	Staff Document	Promotion	1001		Promotion.txt	Staff/1001	5	Successful	
7	\\vmware-host\Shared Folders\TestData\Staff\B\Brown, Jane - 1003\Contract.txt	Staff Document	Contract	1003		Contract.txt	Staff/1003	6	Successful	
8	\\vmware-host\Shared Folders\TestData\Staff\B\Brown, Jane - 1003\Promotion.txt	Staff Document	Promotion	1003		Promotion.txt	Staff/1003	7	Successful	
9	\\vmware-host\Shared Folders\TestData\Staff\B\Burton, Sarah - 1004\Contract.txt	Staff Document	Contract	1004		Contract.txt	Staff/1004	8	Successful	
10	\\vmware-host\Shared Folders\TestData\Staff\B\Burton, Sarah - 1004\Promotion.txt	Staff Document	Promotion	1004		Promotion.txt	Staff/1004	9	Successful	
11	\\vmware-host\Shared Folders\TestData\Staff\K\Kumar, Sam - 1005\Contract.txt	Staff Document	Contract	1005		Contract.txt	Staff/1005	10	Successful	
12	\\vmware-host\Shared Folders\TestData\Staff\K\Kumar, Sam - 1005\Promotion.txt	Staff Document	Promotion	1005		Promotion.txt	Staff/1005	11	Successful	
13										
14										

4.3.3 The Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<DataSetImportSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Source>
    <!-- The source used is OLEDbSelect meaning that DIFS expected to run a SQL select statement against an OleDB data
source.  OleDB select will ignore columns like importstatus-->
    <SourceDataSetType>OLEDbSelect</SourceDataSetType>
    <OleDbSourceDataSetSettings>
      <!-- The Connection string uses a microsoft provider to access an Excel spreadsheet-->
      <ConnectionString>Provider=Microsoft.ACE.OLEDB.12.0;Data Source=\\vmware-host\Shared
Folders\Projects\ProductDevelopment\ImportForSharePoint\WorkingArea\Examples\StaffDocuments.xlsx;Extended Properties="Excel
12.0 Xml;HDR=YES;IMEX=0";</ConnectionString>
    </OleDbSourceDataSetSettings>
    <OleDbSelectSourceDataSetSettings>
      <!-- The select statement will get all the documents from the files worksheet -->
      <SelectStatement>select * from [files$]</SelectStatement>
    </OleDbSelectSourceDataSetSettings>
  </Source>
  <Destination>
    <!-- The authentication type is Office365 i.e. online cloud sharepoint, you can load the configuration file to DIFS to
enter and save the credentials in encrypted format -->
    <AuthenticationSettings>
      <AuthenticationType>Office365</AuthenticationType>
      <domain />
      <username></username>
      <encryptedpassed></encryptedpassed>
    </AuthenticationSettings>
    <DestinationItemSettings>
      <!-- The DestinationItemType you are telling DIFS to make is a Document as opposed to an Item or Folder -->
      <DestinationItemType>Document</DestinationItemType>
      <!-- If the item already exists then overwrite it -->
      <ItemExistsBehaviour>Overwrite</ItemExistsBehaviour>
      <!-- Map data from your source into SharePoint fields-->
      <ImportMappings>
```

```
        <!-- Map title to title assuming that it is a string -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Title</DestinationField>
          <SourceColumn>Title</SourceColumn>
        </ImportMapping>
        <!-- Map Employee Number to EmployeeNumber -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>EmployeeNumber</DestinationField>
          <SourceColumn>Employee Number</SourceColumn>
        </ImportMapping>
      </ImportMappings>
    </DestinationItemSettings>
    <!-- Tell DIFS exactly where the destination list or library is -->
    <DestinationListSettings>
      <DestinationWebUrlRelative>/sites/SPImportHelper</DestinationWebUrlRelative>
      <DestinationFolderUrlRelative>/sites/SPImportHelper/Docs</DestinationFolderUrlRelative>
      <DestinationServerUrl>https://company.sharepoint.com</DestinationServerUrl>
      <DestinationListName>Docs</DestinationListName>
    </DestinationListSettings>
    <!-- Tell DIFS about the source you importing from-->
    <SourceColumns>
      <!-- The column in the source which contains the full path of the file being imported.  The value entered here is
ignored unless DestinationItemType is Document -->
      <SourceFileNameAndPath>FullName</SourceFileNameAndPath>
      <!-- The column in the source the value of which matches the content type in sharepoint to set on the item.  If you
are not using content types on the destination list you can enter an OOTB SharePoint content type such as Item, Document,
Folder -->
      <ContentType>ContentType</ContentType>
      <!-- The column in the source (if applicable) that contains the subfolder path (If any) to import to -->
      <!-- This could be, for example, "Staff" or "Staff/Bob Smith" or "Staff/1001".  The folder must already exist and can
be created using DIFS -->
      <DestinationSubFolder>DestinationSubDirectories</DestinationSubFolder>
      <!-- The column in the source which contains the destination name. The value entered here is ignored unless
DestinationItemType is Document or Folder -->
      <DestinationFileName>DestinationFileName</DestinationFileName>
    </SourceColumns>
  </Destination>
</DataSetImportSettings>
```

4.3.4 The Destination



Content Types

This document library is configured to allow multiple content types. Use content types to specify the information you want to display about an item, in addition to its policies, workflows, or other behavior. The following content types are currently available in this library:

Content Type	Visible on New Button	Default Content Type
Staff Document	✓	✓
Staff Folder	✓	

Add from existing site content types

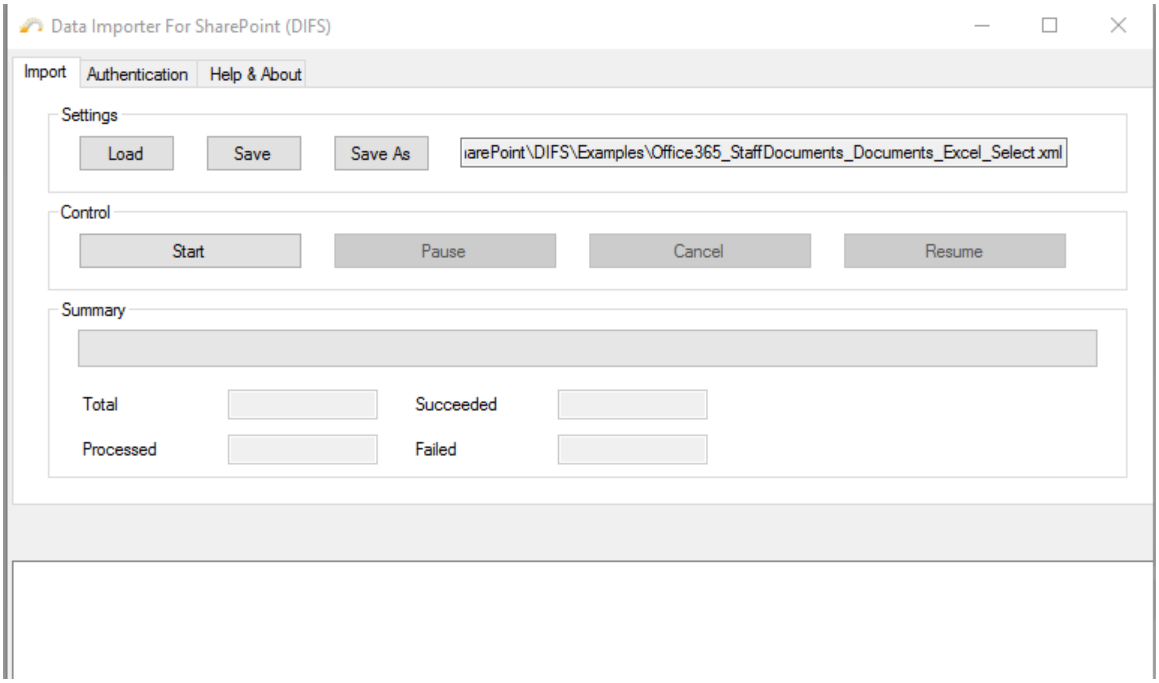
Change new button order and default content type

Columns

A column stores information about each document in the document library. Because this document library allows multiple content types, some column settings, such as whether information is required or optional for a column, are now specified by the content type of the document. The following columns are currently available in this document library:

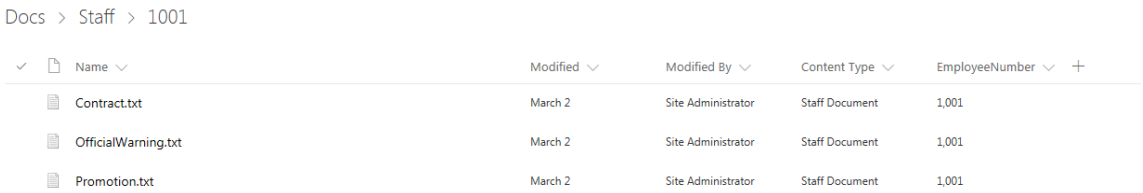
Column (click to edit)	Type	Used in
Created	Date and Time	Staff Document
Date Of Leaving	Date and Time	Staff Folder
Employee Name	Managed Metadata	Staff Folder
EmployeeNumber	Number	Staff Document, Staff Folder
Job	Lookup	Staff Folder
JobJob Description	Lookup	Staff Folder
Modified	Date and Time	Staff Document
Title	Single line of text	Staff Document, Staff Folder
Created By	Person or Group	
Modified By	Person or Group	
Checked Out To	Person or Group	

4.3.5 The Execution



4.3.6 The Result

The staff documents are all in the correct folders.



4.4 Create Document Sets from an Excel Source in SharePoint Online Library

4.4.1 Overview

Create a document set for each employee for their yearly review.

4.4.2 The Source

Content Type	Description	Title	Employee Number	Employee Name	DestinationFolderName	DestinationSubDirectories
Staff Document Set	All documents for Adams, David Yearly Staff Appraisal	Adams, David Yearly Staff Appraisal	1002	Adams, David	1002	Appraisals
Staff Document Set	All documents for Andrews, Robert Yearly Staff Appraisal	Andrews, Robert Yearly Staff Appraisal	1001	Andrews, Robert	1001	Appraisals
Staff Document Set	All documents for Brown, Jane Yearly Staff Appraisal	Brown, Jane Yearly Staff Appraisal	1003	Brown, Jane	1003	Appraisals
Staff Document Set	All documents for Burton, Sarah Yearly Staff Appraisal	Burton, Sarah Yearly Staff Appraisal	1004	Burton, Sarah	1004	Appraisals
Staff Document Set	All documents for Kumar, Sam Yearly Staff Appraisal	Kumar, Sam Yearly Staff Appraisal	1005	Kumar, Sam	1005	Appraisals

4.4.3 The Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<DataSetImportSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Source>
    <SourceDataSetType>OLEDbSelect</SourceDataSetType>
    <OleDbSourceDataSetSettings>
      <ConnectionString>Provider=Microsoft.ACE.OLEDB.12.0;Data Source=\\vmware-host\Shared
Folders\Projects\ProductDevelopment\ImportForSharePoint\WorkingArea\Examples\StaffDocumentSets.xlsx;Extended
Properties="Excel 12.0 Xml;HDR=YES;IMEX=0";</ConnectionString>
    </OleDbSourceDataSetSettings>
    <OleDbTableSourceDataSetSettings />
    <OleDbSelectSourceDataSetSettings>
      <SelectStatement>select * from [directories$]</SelectStatement>
    </OleDbSelectSourceDataSetSettings>
  </Source>
  <Destination>
    <AuthenticationSettings>
      <AuthenticationType>Office365</AuthenticationType>
      <domain />
      <username></username>
      <encryptedpassed></encryptedpassed>
    </AuthenticationSettings>
    <DestinationItemSettings>
      <!-- Document sets can be created in the same way as folders-->
      <DestinationItemType>Folder</DestinationItemType>
      <ItemExistsBehaviour>Overwrite</ItemExistsBehaviour>
      <ImportMappings>
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Title</DestinationField>
          <SourceColumn>Title</SourceColumn>
        </ImportMapping>
```

```
        <!-- Each document set has a description field.  Since there are multiple description fields in SharePoint we will
use the internal name-->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>DocumentSetDescription</DestinationField>
          <SourceColumn>Description</SourceColumn>
        </ImportMapping>
        <ImportMapping xsi:type="ImportMapping_Native">
          <DestinationField>EmployeeNumber</DestinationField>
          <SourceColumn>Employee Number</SourceColumn>
        </ImportMapping>
        <ImportMapping xsi:type="ImportMapping_ManagedMetaDataCSOM">
          <DestinationField>Employee Name</DestinationField>
          <SourceColumn>Employee Name</SourceColumn>
        </ImportMapping>
      </ImportMappings>
    </DestinationItemSettings>
    <DestinationListSettings>
      <DestinationWebUrlRelative>/sites/SPImportHelper</DestinationWebUrlRelative>
      <DestinationFolderUrlRelative>/sites/SPImportHelper/Docs</DestinationFolderUrlRelative>
      <DestinationServerUrl>https://company.sharepoint.com</DestinationServerUrl>
      <DestinationListName>Docs</DestinationListName>
    </DestinationListSettings>
    <SourceColumns>
      <SourceFileNameAndPath>FullName</SourceFileNameAndPath>
      <ContentType>ContentType</ContentType>
      <DestinationSubFolder>DestinationSubDirectories</DestinationSubFolder>
      <DestinationFileName>DestinationFolderName</DestinationFileName>
    </SourceColumns>
  </Destination>
</DataSetImportSettings>
```

4.4.4 The Destination

4.4.4.1 Content Type

A document set content type.

Site Content Type Information

Name: Staff Document Set
Description:
Parent: Document Set
Group: SPImportHelper

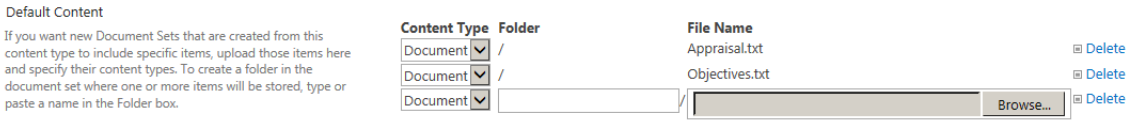
Settings

- Name, description, and group
- Advanced settings
- Workflow settings
- Delete this site content type
- Document Set settings
- Information management policy settings

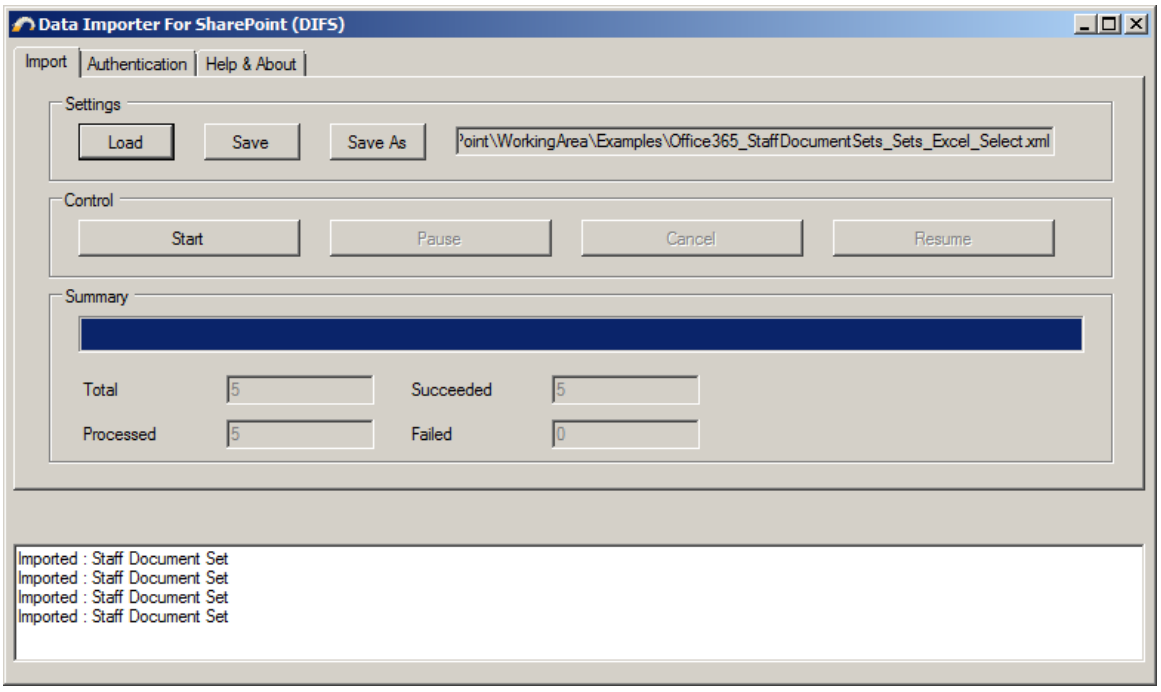
Columns

Name	Type
Title	Single line of text
Name	File
Description	Multiple lines of text
Employee Name	Managed Metadata
EmployeeNumber	Number

Which deploys some standard documents.

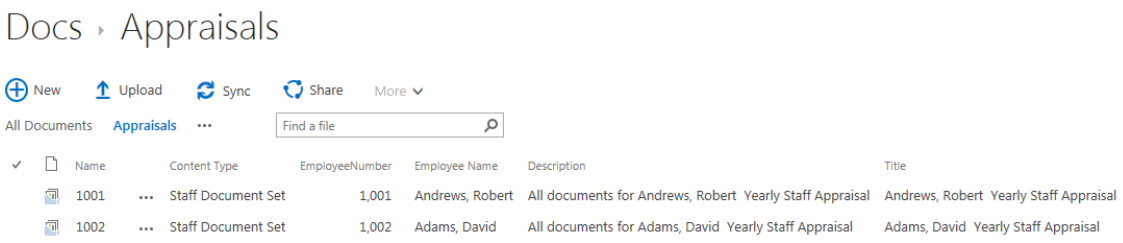


4.4.5 The Execution

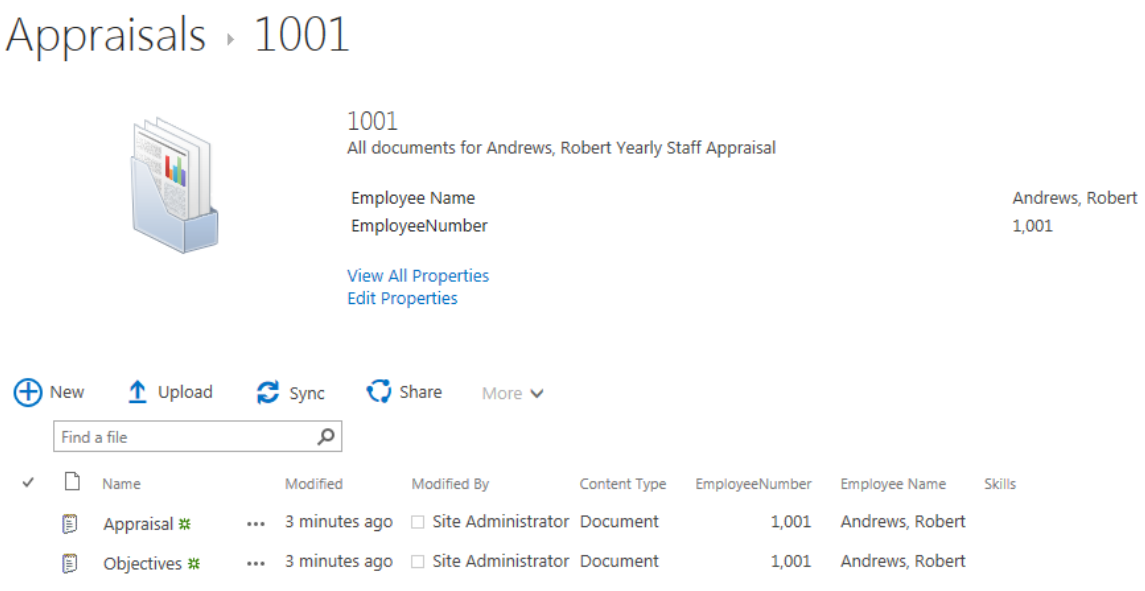


4.4.6 The Result

The document sets are created.



Each document set auto-populates

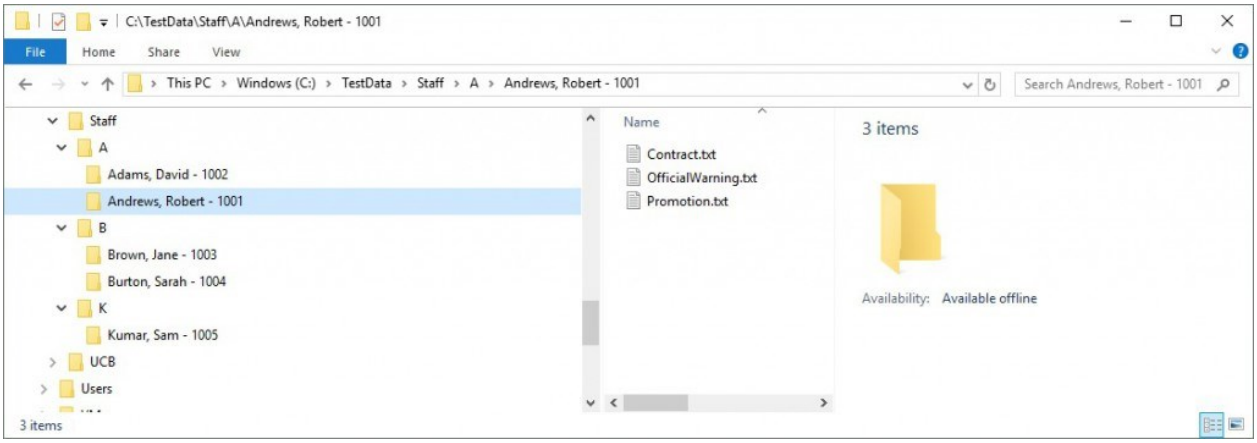


4.5 File System Migration – Staff Records

4.5.1 Scenario

Our theoretical scenario mirrors what is so often encountered when dealing with file shares.

We have a basic file structure. At the outer nodes the files are stored. The "meta data" is inferred by the location of the file in this structure as shown below.



The file share is the only data source. There is no Staff Database. If there was then this might be handled differently with that database providing some of the data.

The example has you have now seen is for some Staff Records. Such records need to be stored in a compliant manner and retention scheduling is key to ensure that we retain only the correct records for each staff member. The requirement in this post is vastly simplified in comparison to most Staff Record scenarios but it serves to illustrate the concepts very well.

4.5.2 Why Migrate

Be clear on why you are migrating the data into SharePoint before you start the migration process. You may need to design the migration process to ensure that the desired benefits are achieved.

In our scenario the key drivers are;

- Compliance – Specifically data retention scheduling.
- Efficiency – Consolidation into SharePoint.
- Efficiency – Ease of use.
- Efficiency – Process automation.

4.5.3 Design and Implement the Destination

Before you execute migration you need to have a destination to migrate into.

This is key for two reasons;

- Migrating a live file share which is being updated is harder to manage.
- Until you have designed and implemented the destination you won't necessarily know how to define the import sources, in this example the two Excel spreadsheets.

For our scenario we have implemented;

- A record centre – /sites/Staff/Records.
- A record library – Records
- A top level folder "Staff"
- A content type for each staff folder "Staff Folder" which has a date of leaving field and retention action set from that date.
- A content type for each staff record "Staff Record" which has an employee number field.
- A content type for each disciplinary record "Staff Disciplinary Record" which has an Disciplinary Date and retention action set from that date because these records are kept for a shorter period of time.

It is sometimes useful to catalogue the file shares as part of this design process. This will give you an insight into the scenarios that your destination will need to cope with. The file shares can then be re-catalogued for migration at a later date.

4.5.4 Catalogue the File Share

To catalogue the file share we can use a PowerShell script.

You can run the script either from a PC as a user with access to the file share OR from the server hosting the file share. The script below will audit both the files in the file share and the directories, each to separate CSV files.

```
Function Audit-File($file)
{
   Write-Host "Auditing: " $file.FullName;
```

```
    $file | add-member -name "Owner" -membertype noteproperty -value (get-acl $_.fullname).owner;
    $file | Add-Member -Name "Action" -MemberType NoteProperty -Value "Copy";

    return $file
}

Function Audit-Files($source, $destination)
{

    Get-ChildItem -Recurse $source |  ?{-not $_.PSIsContainer} | ForEach-Object {Audit-File $_} | Sort-Object fullname | Select
FullName,CreationTime,LastWriteTime,Length,Owner,BaseName,Name,Extension,Action,ContentType,Title,Meta1,Meta2,De
stinationFileName,DestinationWebUrl,DestinationLibraryName,DestinationSubDirectories | Export-Csv -Force -
NoTypeInformation $destination


}

Function Audit-Directory($folder)
{
    Write-Host "Auditing: " $folder.FullName;

    $folder | add-member -name "Owner" -membertype noteproperty -value (get-acl $_.fullname).owner;
    $folder | Add-Member -Name "Files" -MemberType NoteProperty -Value ($_.GetFiles().Count).ToString();
    $folder | Add-Member -Name "Directories" -MemberType NoteProperty -Value ($_.GetDirectories().Count).ToString();
    $folder | Add-Member -Name "Action" -MemberType NoteProperty -Value "Create";

    return $folder;
}

Function Audit-Directories($source, $destination)
{

    Get-ChildItem -Recurse $source |  ?{$_.PSIsContainer} | ForEach-Object {Audit-Directory $_} | Sort-Object fullname |
Select
FullName,CreationTime,LastWriteTime,Owner,Files,Directories,Action,ContentType,Title,Meta1,Meta2,DestinationFileName,
DestinationWebUrl,DestinationLibraryName,DestinationSubDirectories | Export-Csv -Force -NoTypeInformation
$destination


}

Audit-Files "\\vmware-host\Shared Folders\TestData\Staff\" "\\vmware-host\Shared Folders\TestData\files.csv"
Audit-Directories "\\vmware-host\Shared Folders\TestData\Staff\" \\vmware-host\Shared Folders\TestData\directories.csv
```

The CSV files will contain the basic information that is available from the file system.

Tip: Try and use UNC paths instead of mapped drive letters when cataloguing file shares.

If you are unsure how to execute PowerShell scripts then pop “how to execute a powershell script” into your favourite search engine.

4.5.5 Prepare the Excel Spreadsheets

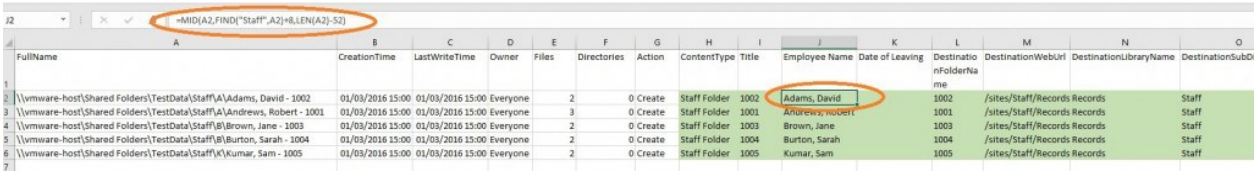
The CSV files can be imported to Excel and turned into a spreadsheet.

This will enable us to automate the population of meta data that is so important to the success of migration projects

4.5.5.1 Folders

In Excel we can easily populate some extra columns (shown here in green).

Here we are going to create an import source which will create a folder of content type "Staff Folder" for each staff member. This is going to have the Date of Leaving (which will drive retention schedules) and the staff name as meta data. The employee name and the employee number will be extracted from the folder name from the file system using an Excel formula.

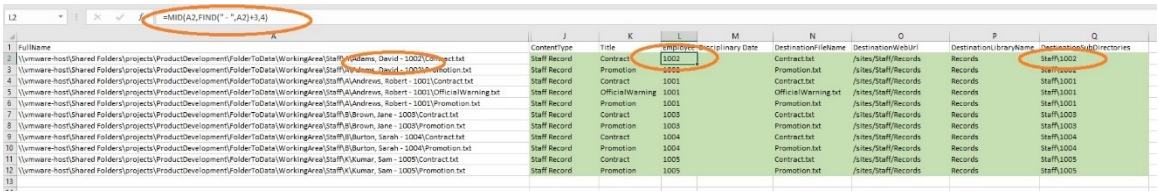


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	FullName	CreationTime	LastWriteTime	Owner	Files	Directories	Action	ContentType	Title	Employee Name	Date of Leaving	DestinationFolderName	DestinationWebUrl	DestinationLibraryName	DestinationSubDi
2	\\vmware-host\Shared Folders\TestData\Staff\A\Adams, David - 1002	01/03/2016 15:00	01/03/2016 15:00	Everyone	2	0	Create	Staff Folder	1002	Adams, David		1002	/sites/Staff/Records	Records	Staff
3	\\vmware-host\Shared Folders\TestData\Staff\A\Andrews, Robert - 1001	01/03/2016 15:00	01/03/2016 15:00	Everyone	3	0	Create	Staff Folder	1001	Andrews, Robert		1001	/sites/Staff/Records	Records	Staff
4	\\vmware-host\Shared Folders\TestData\Staff\B\Brown, Jane - 1003	01/03/2016 15:00	01/03/2016 15:00	Everyone	2	0	Create	Staff Folder	1003	Brown, Jane		1003	/sites/Staff/Records	Records	Staff
5	\\vmware-host\Shared Folders\TestData\Staff\B\Burton, Sarah - 1004	01/03/2016 15:00	01/03/2016 15:00	Everyone	2	0	Create	Staff Folder	1004	Burton, Sarah		1004	/sites/Staff/Records	Records	Staff
6	\\vmware-host\Shared Folders\TestData\Staff\K\Kumar, Sam - 1005	01/03/2016 15:00	01/03/2016 15:00	Everyone	2	0	Create	Staff Folder	1005	Kumar, Sam		1005	/sites/Staff/Records	Records	Staff
7															

4.5.5.2 Files

In Excel we can easily populate some extra columns shown in green.

Here we are going to create an import source which will import all of the files for the Staff. The employee number is extracted from the folder name which will tell us destination folder and the set some meta data against each document. This is extracted from the folder path.



	A	J	K	L	M	N	O	P	Q
1	FullName	ContentType	Title	Employee	Disciplinary Date	DestinationFileName	DestinationWebUrl	DestinationLibraryName	DestinationSubDirectories
2	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\A\Adams, David - 1002\Contract.txt	Staff Record	Contract	1002		Contract.txt	/sites/Staff/Records	Records	Staff\1002
3	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\A\Adams, David - 1002\Promotion.txt	Staff Record	Promotion	1002		Promotion.txt	/sites/Staff/Records	Records	Staff\1002
4	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\A\Andrews, Robert - 1001\Contract.txt	Staff Record	Contract	1001		Contract.txt	/sites/Staff/Records	Records	Staff\1001
5	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\A\Andrews, Robert - 1001\OfficialWarning.txt	Staff Record	OfficialWarning	1001		OfficialWarning.txt	/sites/Staff/Records	Records	Staff\1001
6	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\A\Andrews, Robert - 1001\Promotion.txt	Staff Record	Promotion	1001		Promotion.txt	/sites/Staff/Records	Records	Staff\1001
7	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\B\Brown, Jane - 1003\Contract.txt	Staff Record	Contract	1003		Contract.txt	/sites/Staff/Records	Records	Staff\1003
8	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\B\Brown, Jane - 1003\Promotion.txt	Staff Record	Promotion	1003		Promotion.txt	/sites/Staff/Records	Records	Staff\1003
9	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\B\Burton, Sarah - 1004\Contract.txt	Staff Record	Contract	1004		Contract.txt	/sites/Staff/Records	Records	Staff\1004
10	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\B\Burton, Sarah - 1004\Promotion.txt	Staff Record	Promotion	1004		Promotion.txt	/sites/Staff/Records	Records	Staff\1004
11	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\K\Kumar, Sam - 1005\Contract.txt	Staff Record	Contract	1005		Contract.txt	/sites/Staff/Records	Records	Staff\1005
12	\\vmware-host\Shared Folders\projects\ProductDevelopment\FolderToData\WorkingArea\Staff\K\Kumar, Sam - 1005\Promotion.txt	Staff Record	Promotion	1005		Promotion.txt	/sites/Staff/Records	Records	Staff\1005
13									

4.5.6 Get User Input

One of the strengths of Excel is that most users will have skills in using it.

What this enables us to do is use it to capture from the user base any additional data and permit the users to generally cleanse the data.

Tip: Give the users some guidance notes on how the Excel spreadsheets should be completed.

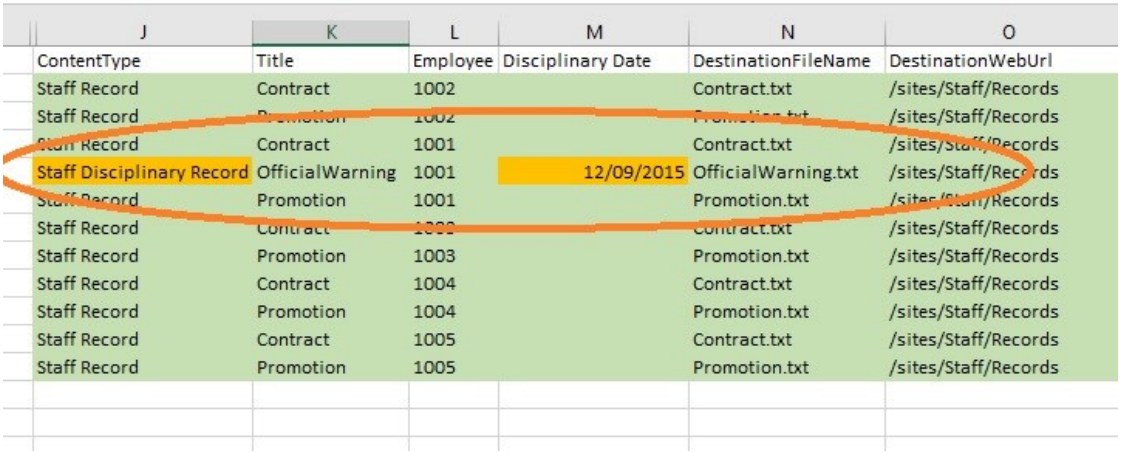
4.5.6.1 Folders

Here the user has completed the date of leaving field which in turn will allow SharePoint to manage the retention schedule accordingly.

G	H	I	J	K	L	M	N
Action	ContentType	Title	Employee Name	Date of Leaving	DestinationFolderName	DestinationWebUrl	DestinationLibraryName
Create	Staff Folder	1002	Adams, David		1002	/sites/Staff/Records	Records
Create	Staff Folder	1001	Andrews, Robert		1001	/sites/Staff/Records	Records
Create	Staff Folder	1003	Brown, Jane		3	/sites/Staff/Records	Records
Create	Staff Folder	1004	Burton, Sarah	13/10/2013	1004	/sites/Staff/Records	Records
Create	Staff Folder	1005	Kumar, Sam		1005	/sites/Staff/Records	Records

4.5.6.2 Files

Here the user has spotted that one file is a disciplinary document. They have therefore changed the disciplinary date and content type accordingly.



J	K	L	M	N	O
ContentType	Title	Employee	Disciplinary Date	DestinationFileName	DestinationWebUrl
Staff Record	Contract	1002		Contract.txt	/sites/Staff/Records
Staff Record	[illegible]	1002		[illegible]	/sites/Staff/Records
[illegible] Record	Contract	1001		Contract.txt	/sites/[illegible]/Records
Staff Disciplinary Record	OfficialWarning	1001	12/09/2015	OfficialWarning.txt	/sites/Staff/Rec[illegible]rds
[illegible] Record	Promotion	1001		Promotion.txt	/sites/[illegible]/Records
Staff Record	[illegible]	[illegible]		[illegible]	/sites/Staff/Records
Staff Record	Promotion	1003		Promotion.txt	/sites/Staff/Records
Staff Record	Contract	1004		Contract.txt	/sites/Staff/Records
Staff Record	Promotion	1004		Promotion.txt	/sites/Staff/Records
Staff Record	Contract	1005		Contract.txt	/sites/Staff/Records
Staff Record	Promotion	1005		Promotion.txt	/sites/Staff/Records

4.5.7 Preparation Complete

Once preparation is complete you should have a set of Excel spreadsheets.

This should be double checked and quality controlled before you commence the migration process but the core work is done.

You can now create the folder structure in SharePoint - Create Folders from an Excel Source in a SharePoint Online Library adding meta data.

You can then import the documents into that folder structure - Import Documents from an Excel Source into a SharePoint Online Library,

4.6 Import from / Migrate Legacy Document Management Systems

4.6.1 The Source

4.6.1.1 Introduction

Most document management systems are built to the same basic architecture.

The meta data is stored in a database – typically SQL or Oracle.

The documents themselves are stored on file server(s).

With a bit of work to decode the schema it is frequently possible to perform a migration directly into SharePoint using DIFS.

This section works through an example for a document management system (DMS) with a very simple schema used for storing shipping contracts.

The same approach will work for more complex schemas as such as OpenText LiveLink.

Some DM systems require you to access the documents via an API. Talk to the DIFS developers in this instance for assistance.

4.6.1.2 Schema

The DMS has two tables “Documents” and “Versions”.

We can look at those tables.

```
use DMSystem
select top 10
ID,
TITLE,
field10,
field16
from Documents


select top 10
OBJECT_ID,
FilePath
from versions
```

ID	TITLE	field10	field16
1042	Contract Cost Agreement	Shipping	FALKLAND I
1043	Contract Cost Agreement	Shipping	ST HELENA
1045	Contract Cost Agreement	Shipping	ASCENSION
1047	Contract Cost Agreement	Shipping	BAHAMAS
1052	Contract Cost Agreement	Shipping	GHANA
1056	Contract Cost Agreement	Shipping	GAMBIA
1086	Contract Cost Agreement	Shipping	UGANDA
1088	Contract Cost Agreement	Shipping	UGANDA
1089	Contract Cost Agreement	Shipping	ZIMBABWE
1090	Contract Cost Agreement	Shipping	ZIMBABWE

OBJECT_ID	FilePath
124895	\\vmware-host\Shared

	Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
124896	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
124904	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
125003	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
125134	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
125135	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
125137	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
125138	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
125139	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf
125172	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\SourceCode\Examples\Test.pdf

Ok so this is an easy one to work out.

The ID relates the two tables.

The **Documents** table stores the meta data and the **Versions** table stores the pointers to the files.

4.6.1.3 Create an Import Source

Run a SQL command to create a single ImportSource table that contains all the information that DIFS needs to process the Import.

```
select TITLE,filename,field10,field16,FilePath,
'Contracts' as DestinationSubDirectories,
'Contract' as ContentType,
Versions.ID as ImportKey,
CAST('Import' as varchar(255)) as ImportStatus,
CAST(null as varchar(255)) as Exception
into ImportSource
from Documents,Versions
where documents.ID = versions.Object_ID
order by Versions.ID
```

We can have a look at our new table.

```
select top 10 * from ImportSource
```

TITLE	filename	field10	field16	FilePath	DestinationSubDirectories	ContentType	ImportKey	ImportStatus	Exception
Contract Cost Agreement.TIF	000	Ship	FAL	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	10	Import	NULL
Contract Cost Agreement.TIF	000	Ship	FAST	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	11	Import	NULL
Contract Cost Agreement.TIF	000	Ship	FAS	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	12	Import	NULL
Contract Cost Agreement.TIF	000	Ship	BAH	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	13	Import	NULL
Contract Cost Agreement.TIF	000	Ship	GHA	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	14	Import	NULL
Contract Cost Agreement.TIF	000	Ship	FAL	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	15	Import	NULL
Contract Cost Agreement.TIF	000	Ship	FAST	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	16	Import	NULL
Contract Cost Agreement.TIF	000	Ship	FAS	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	17	Import	NULL
Contract Cost Agreement.TIF	000	Ship	BAH	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	18	Import	NULL
Contract Cost Agreement.TIF	000	Ship	GHA	\\vmware-host\Shared Folders\Projects\ProductDevelopment\ImportForSharePoint\Source Code\Examples\Test.pdf	Contracts	Contract	19	Import	NULL

Contra	000	Sh		\\vmware-host\Shared					
ct Cost	000	ip	GA	Folders\Projects\ProductDevelop		Co			N
Agree	31.	pi	MB	ment\ImportForSharePoint\Source	Contrac	ntr	10	Imp	UL
ment	TIF	ng	IA	Code\Examples\Test.pdf	ts	act	46	ort	L
Contra	000	Sh		\\vmware-host\Shared					
ct Cost	000	ip	UG	Folders\Projects\ProductDevelop		Co			N
Agree	52.	pi	AN	ment\ImportForSharePoint\Source	Contrac	ntr	10	Imp	UL
ment	TIF	ng	DA	Code\Examples\Test.pdf	ts	act	73	ort	L
Contra	000	Sh		\\vmware-host\Shared					
ct Cost	000	ip	UG	Folders\Projects\ProductDevelop		Co			N
Agree	53.	pi	AN	ment\ImportForSharePoint\Source	Contrac	ntr	10	Imp	UL
ment	TIF	ng	DA	Code\Examples\Test.pdf	ts	act	75	ort	L
Contra	000	Sh	ZI	\\vmware-host\Shared					
ct Cost	000	ip	MB	Folders\Projects\ProductDevelop		Co			N
Agree	54.	pi	AB	ment\ImportForSharePoint\Source	Contrac	ntr	10	Imp	UL
ment	TIF	ng	WE	Code\Examples\Test.pdf	ts	act	76	ort	L
Contra	000	Sh	ZI	\\vmware-host\Shared					
ct Cost	000	ip	MB	Folders\Projects\ProductDevelop		Co			N
Agree	55.	pi	AB	ment\ImportForSharePoint\Source	Contrac	ntr	10	Imp	UL
ment	TIF	ng	WE	Code\Examples\Test.pdf	ts	act	77	ort	L

4.6.2 The Configuration

```
<?xml version="1.0" encoding="utf-8"?>
<DataSetImportSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Source>
    <!-- The source used is OLEDbTable-->
    <SourceDataSetType>OLEDbTable</SourceDataSetType>
    <!-- The Connection string uses a SQL Server data set-->
    <OleDbSourceDataSetSettings>

<ConnectionString>Provider=SQLNCLI10;Server=127.0.0.1\SharePoint;Database=DMSystem;Trusted_Connection=yes;</ConnectionString
>
    </OleDbSourceDataSetSettings>
    <!-- We will use the ImportSource table-->
    <OleDbTableSourceDataSetSettings>
      <TableName>ImportSource</TableName>
    </OleDbTableSourceDataSetSettings>
  </Source>
  <Destination>
    <!-- The authentication type is current because we are using SharePoint Server On-Premises and running the import as a
user with sufficient permissions -->
    <AuthenticationSettings>
      <AuthenticationType>Current</AuthenticationType>
      <domain />
      <username></username>
      <encryptedpassed></encryptedpassed>
    </AuthenticationSettings>
    <DestinationItemSettings>
      <!-- The DestinationItemType you are telling DIFS to make is an item as opposed to a File or Folder -->
      <DestinationItemType>Document</DestinationItemType>
      <!-- If the item already exists then overwrite it -->
      <ItemExistsBehaviour>Overwrite</ItemExistsBehaviour>
      <!-- Map data from your source into SharePoint fields-->
      <ImportMappings>
        <!-- Map title to title assuming that it is a string -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Title</DestinationField>
```

```
          <SourceColumn>Title</SourceColumn>
        </ImportMapping>
        <!-- Field10 is the contract type -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>ContractType</DestinationField>
          <SourceColumn>Field10</SourceColumn>
        </ImportMapping>
        <!-- Field16 is the country -->
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Country</DestinationField>
          <SourceColumn>Field16</SourceColumn>
        </ImportMapping>
      </ImportMappings>
    </DestinationItemSettings>
    <!-- Tell DIFS exactly where the destination list or library is -->
    <DestinationListSettings>
      <DestinationWebUrlRelative>/sites/SPImportHelper</DestinationWebUrlRelative>
      <DestinationFolderUrlRelative>/sites/SPImportHelper/Docs</DestinationFolderUrlRelative>
      <DestinationServerUrl>http://productdev</DestinationServerUrl>
      <DestinationListName>Docs</DestinationListName>
    </DestinationListSettings>
    <!-- Tell DIFS about the source you importing from-->
    <SourceColumns>
      <!-- The column in the source which contains the full path of the file being imported.  The value entered here is
ignored unless DestinationItemType is Document -->
      <SourceFileNameAndPath>filepath</SourceFileNameAndPath>
      <!-- The column in the source the value of which matches the content type in sharepoint to set on the item.  If you
are not using content types on the destination list you can enter an OOTB SharePoint content type such as Item, Document,
Folder -->
      <ContentType>ContentType</ContentType>
      <!-- The column in the source (if applicable) that contains the subfolder path (If any) to import to -->
      <!-- This could be, for example, "Staff" or "Staff/Managers" or "Staff/Managers/Retired".  The folder must already
exist and can be created using DIFS -->
      <DestinationSubFolder>DestinationSubDirectories</DestinationSubFolder>
      <!-- The column in the source which contains the destination name. The value entered here is ignored unless
DestinationItemType is Document or Folder -->
      <DestinationFileName>filename</DestinationFileName>
    </SourceColumns>
```

```
  </Destination>
</DataSetImportSettings>
```

4.6.4 The Destination

4.6.4.1 Content Type

Site Content Type Information

Name: Contract

Description:

Parent: Document

Group: SPImportHelper

Settings

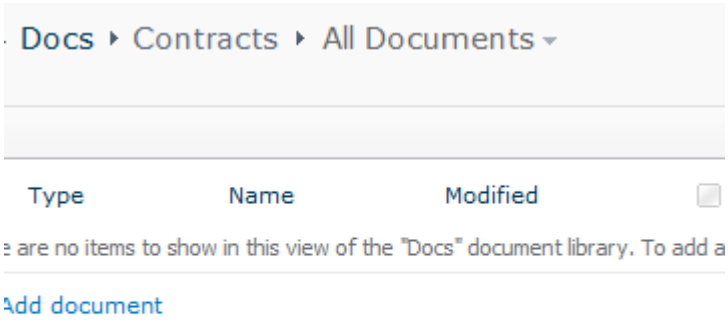
- Name, description, and group
- Advanced settings
- Workflow settings
- Delete this site content type
- Document Information Panel settings
- Information management policy settings

Columns

Name	Type
Name	File
Title	Single line of text
ContractType	Single line of text
Country	Single line of text

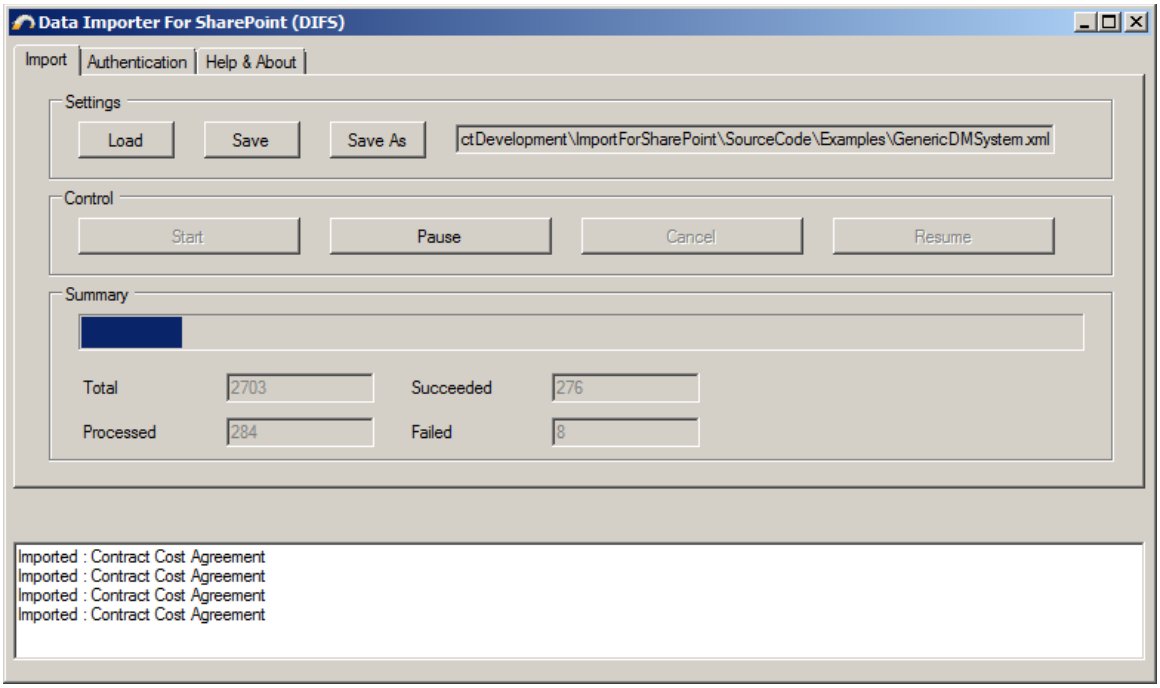
- Add from existing site columns
- Add from new site column
- Column order

4.6.4.2 Library / Folder



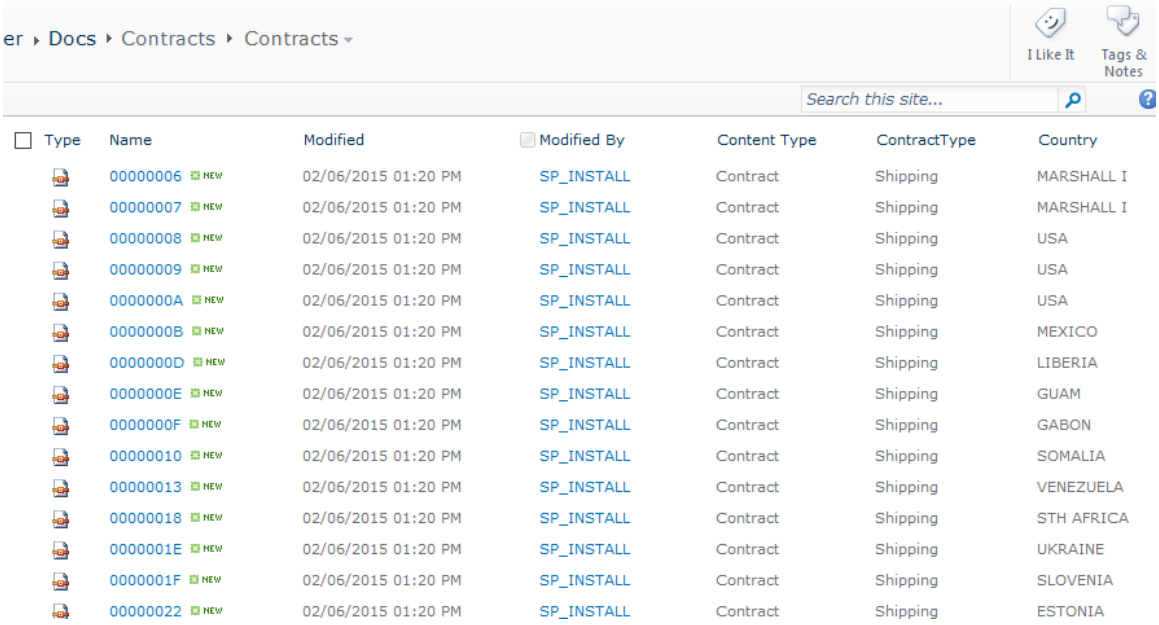
4.6.5 The Execution

DIFS will show progress as files are imported.



4.6.6 The Result

The files are imported into SharePoint / SharePoint online.



Type	Name	Modified	Modified By	Content Type	ContractType	Country
	00000006 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	MARSHALL I
	00000007 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	MARSHALL I
	00000008 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	USA
	00000009 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	USA
	0000000A NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	USA
	0000000B NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	MEXICO
	0000000D NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	LIBERIA
	0000000E NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	GUAM
	0000000F NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	GABON
	00000010 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	SOMALIA
	00000013 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	VENEZUELA
	00000018 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	STH AFRICA
	0000001E NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	UKRAINE
	0000001F NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	SLOVENIA
	00000022 NEW	02/06/2015 01:20 PM	SP_INSTALL	Contract	Shipping	ESTONIA

4.7 Import Documents from an Excel Source into OneDrive for Business

4.7.1 Overview

An Excel spreadsheet contains a list of documents, the path to each document and associated meta data.

DIFS imports the meta data from the spreadsheet and the associated document into OneDrive for Business.

4.7.2 The Source

FullName	ContentType	Title	DestinationFileName
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test1	Test1.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test2	Test2.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test3	Test3.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test4	Test4.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test5	Test5.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test6	Test6.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test7	Test7.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test8	Test8.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test9	Test9.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test10	Test10.pdf
\\vmware-host\Shared Folders\TestData\Generic\Test.pdf	Document	Test11	Test11.pdf

4.7.3 The Configuration

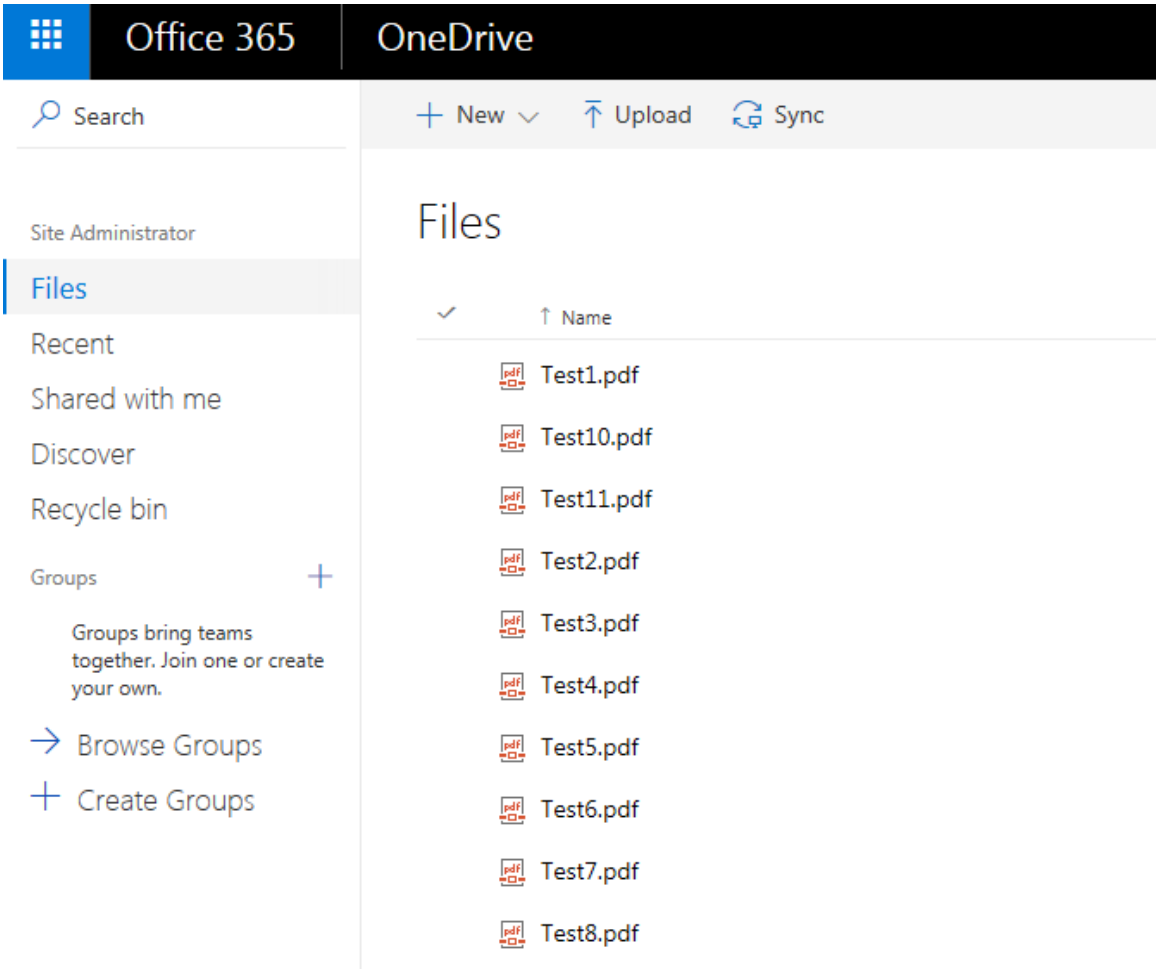
```
<?xml version="1.0" encoding="utf-8"?>
<DataSetImportSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Source>
    <SourceDataSetType>OLEDbSelect</SourceDataSetType>
    <OleDbSourceDataSetSettings>
      <ConnectionString>Provider=Microsoft.ACE.OLEDB.12.0;Data Source=\\vmware-host\Shared
Folders\Projects\ProductDevelopment\ImportForSharePoint\WorkingArea\Examples\OneDrive.xlsx;Extended Properties="Excel 12.0
Xml;HDR=YES;IMEX=0";</ConnectionString>
    </OleDbSourceDataSetSettings>
    <OleDbTableSourceDataSetSettings />
    <OleDbSelectSourceDataSetSettings>
      <SelectStatement>select * from [files$]</SelectStatement>
    </OleDbSelectSourceDataSetSettings>
  </Source>
  <Destination>
    <AuthenticationSettings>
      <AuthenticationType>Office365</AuthenticationType>
      <domain />
      <username>me@company.onmicrosoft.com</username>
      <encryptedpassed> /encryptedpassed>
    </AuthenticationSettings>
    <DestinationItemSettings>
      <DestinationItemType>Document</DestinationItemType>
      <ItemExistsBehaviour>Overwrite</ItemExistsBehaviour>
      <ImportMappings>
        <ImportMapping xsi:type="ImportMapping_String">
          <DestinationField>Title</DestinationField>
          <SourceColumn>Title</SourceColumn>
        </ImportMapping>
      </ImportMappings>
    </DestinationItemSettings>
```

```
    <DestinationListSettings>
      <DestinationWebUrlRelative>/personal/me_company_onmicrosoft_com</DestinationWebUrlRelative>
      <DestinationFolderUrlRelative>/personal/david_company_onmicrosoft_com/Documents</DestinationFolderUrlRelative>
      <DestinationServerUrl>https://company-my.sharepoint.com</DestinationServerUrl>
      <DestinationListName>Documents</DestinationListName>
    </DestinationListSettings>
    <SourceColumns>
      <SourceFileNameAndPath>FullName</SourceFileNameAndPath>
      <ContentType>ContentType</ContentType>
      <DestinationSubFolder>DestinationSubDirectories</DestinationSubFolder>
      <DestinationFileName>DestinationFileName</DestinationFileName>
      <Publish>Publish</Publish>
      <CheckInComment>CheckInComment</CheckInComment>
      <PublishComment>PublishComment</PublishComment>
    </SourceColumns>
  </Destination>
</DataSetImportSettings>
```

4.7.4 The Destination

User's OneDrive for business "Files".

4.7.5 The Result



5 Source Configuration

The Source section of the XML configuration file defines the source to Import from.

.

5.1 SourceDataSetType

Defines how to get the source data set.

5.1.1 OLEDbTable

You can define a data source as being a table using OLEDbTable. This allows each row to be updated after import as successful or otherwise along with anyException data. This is useful for large imports.

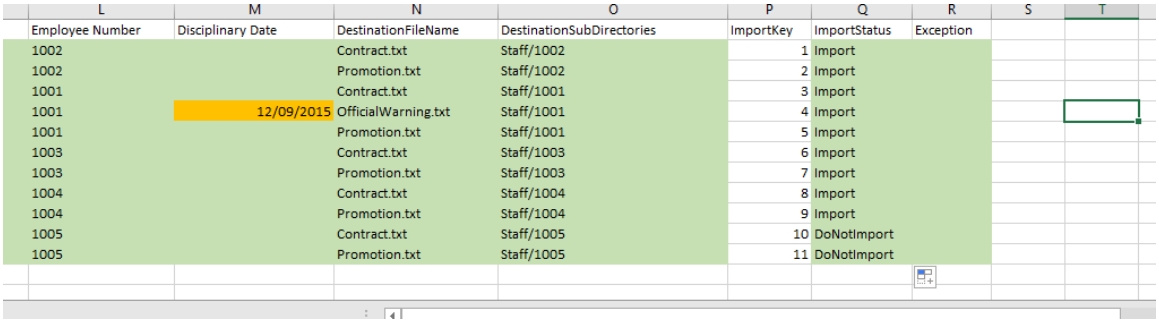
Below the table that we are telling DIFS to use is a worksheet “Clients” in an excel spreadsheet.

```
  <Source>
    <SourceDataSetType>OLEDbTable</SourceDataSetType>
    <OleDbSourceDataSetSettings>
      <ConnectionString>Provider=Microsoft.ACE.OLEDB.12.0;Data Source=\\vmware-
host\Shared Folders\projects\ClientFolders.xlsx;Extended Properties="Excel 12.0
Xml;HDR=YES;IMEX=0";</ConnectionString>
    </OleDbSourceDataSetSettings>
    <OleDbTableSourceDataSetSettings>
      <TableName>clients$</TableName>
    </OleDbTableSourceDataSetSettings>
  </Source>
```

The table must contain columns ImportKey, ImportStatus, Exception.

ImportKey must be unique.

In the example below only the first 10 rows will be considered for import since only they have the importstatus “Import”.



	L	M	N	O	P	Q	R	S	T
	Employee Number	Disciplinary Date	DestinationFileName	DestinationSubDirectories	ImportKey	ImportStatus	Exception		
	1002		Contract.txt	Staff/1002	1	Import			
	1002		Promotion.txt	Staff/1002	2	Import			
	1001		Contract.txt	Staff/1001	3	Import			
	1001	12/09/2015	OfficialWarning.txt	Staff/1001	4	Import			
	1001		Promotion.txt	Staff/1001	5	Import			
	1003		Contract.txt	Staff/1003	6	Import			
	1003		Promotion.txt	Staff/1003	7	Import			
	1004		Contract.txt	Staff/1004	8	Import			
	1004		Promotion.txt	Staff/1004	9	Import			
	1005		Contract.txt	Staff/1005	10	DoNotImport			
	1005		Promotion.txt	Staff/1005	11	DoNotImport			

After you have run the import the spreadsheet will be updated.

N	O	P	Q	R	S	T	U	V
tionFileName	DestinationSubDirectories	ImportKey	ImportStatus	Exception				
t.txt	Staff/1002	1	Successful					
ion.txt	Staff/1002	2	Successful					
t.txt	Staff/1001	3	Successful					
Warning.txt	Staff/1001	4	Successful					
ion.txt	Staff/1001	5	Successful					
t.txt	Staff/1003	6	Successful					
ion.txt	Staff/1003	7	Successful					
t.txt	Staff/1004	8	Exception	The remote server returned an error: (409) Conflict.				
ion.txt	Staff/1004	9	Exception	The remote server returned an error: (409) Conflict.				
t.txt	Staff/1005	10	DoNotImport					
ion.txt	Staff/1005	11	DoNotImport					

You can then correct the causes of the Exception, change the ImportStatus back to Import and re-run.

5.1.2 OLEDbSelect

You can define a data source as simply the results of a select statement using OLEDbSelect.

```
  <Source>
    <!-- The source used is OLEDbSelect meaning that DIFS expected to run a SQL
select statement against an OleDB data source.  OleDB select will ignore columns
like importstatus-->
    <SourceDataSetType>OLEDbSelect</SourceDataSetType>
    <!-- The Connection string uses a microsoft provider to access an Excel
spreadsheet-->
    <OleDbSourceDataSetSettings>
      <ConnectionString>Provider=Microsoft.ACE.OLEDB.12.0;Data Source=\\vmware-
host\Shared
Folders\Projects\ProductDevelopment\ImportForSharePoint\WorkingArea\Examples\JobDe
```

```
scriptions.xlsx;Extended Properties="Excel 12.0
Xml;HDR=YES;IMEX=0";</ConnectionString>
    </OleDbSourceDataSetSettings>
    <OleDbTableSourceDataSetSettings />
    <!-- The select statement will get all the jobs from the jobs worksheet -->
    <OleDbSelectSourceDataSetSettings>
      <SelectStatement>select * from [jobs$]</SelectStatement>
    </OleDbSelectSourceDataSetSettings>
  </Source>
```

5.2 ConnectionString

This is any valid OleDB connection string.

A google of OleDB connection string will give us example connection strings for 100's of different providers.

DIFS is most commonly used with Excel or SQL Server.

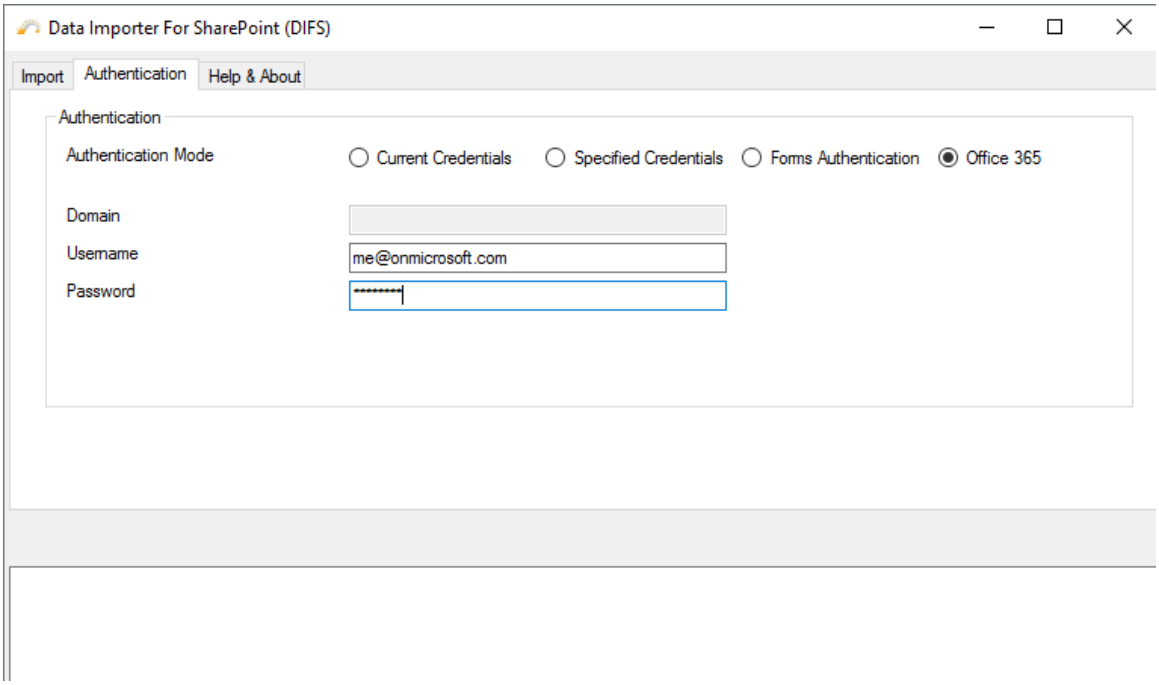
6 Destination Configuration

The Source section of the XML configuration file defines the destination location, authentication, data mapping etc.

6.1 AuthenticationSettings

This tells DIFS how to authenticate to SharePoint.

Since credentials are encrypted the easiest way to configure authentication is from the DIFS user interface – authentication tab.



Once entered you can save the configuration file from the Import tab if required.

6.2 DestinationItemSettings

6.2.1 DestinationItemType

This tells DIFS the base type – Item, Document or Folder – so DIFS knows the basic action to execute.

The content type to set on the item is defined elsewhere.

6.2.2 ItemExistsBehaviour

This setting controls what happens when a file already exists.

As such it also governs how you may import multiple versions. This can be useful when migrating from legacy systems which had version control and where you wish to re-produce that in SharePoint.

The effective that it has is determined by what versioning setting a SharePoint document library has.

The table below shows the files we are uploading – each source is targeted to the same destination TestA.txt.

FullName	ContentType	Title	DestinationFileName
Version1.txt	Document	Version 1	TestA.txt
Version2.txt	Document	Version 2	TestA.txt
Version3.txt	Document	Version 3	TestA.txt
Version4.txt	Document	Version 4	TestA.txt

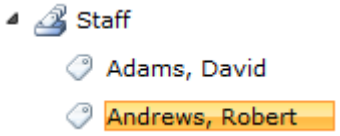
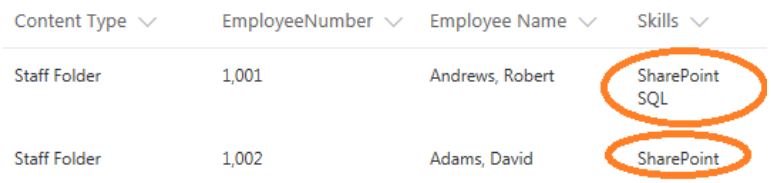
The table below shows the SharePoint setting will determine the end result.

ItemExistsBehaviour	Library	Effect
Overwrite	Major Versions	No. ↓ Modified 4.0 10/6/2016 7:00 AM Title Version 4 3.0 10/6/2016 7:00 AM Title Version 3 2.0 10/6/2016 7:00 AM Title Version 2 1.0 10/6/2016 6:59 AM Title Version 1
Overwrite	Minor Version	No. ↓ Modified 0.4 10/6/2016 7:00 AM Title Version 4 0.3 10/6/2016 7:00 AM Title Version 3 0.2 10/6/2016 7:00 AM Title Version 2 0.1 10/6/2016 6:59 AM Title Version 1
Overwrite	No Versioning	Single version – Version 4
DoNotOverwrite	Major Versions	No. ↓ Modified 1.0 10/6/2016 6:59 AM Title Version 1
DoNotOverwrite	Minor Version	No. ↓ Modified 0.1 10/6/2016 6:59 AM Title Version 1
DoNotOverwrite	No Versioning	Single version – Version 1

6.2.3 ImportMappings

Using ImportMappings allows you to set column / field values on the destination items.

Mapping	Usage
String	Take the string equivalent of SourceColumn and set DestinationField to that value. <ImportMapping xsi:type="ImportMapping_String"> <DestinationField>Title</DestinationField> <SourceColumn>Title</SourceColumn> </ImportMapping>
DateTimeFromString	Take the string value of SourceColumn, convert it to DateTime using the specified culture and format. Set DestinationField to that value. <ImportMapping xsi:type="ImportMapping_DateTimeFromString"> <DestinationField>DateOfLeaving</DestinationField> <SourceColumn>Date of Leaving</SourceColumn> <ConversionMask>dd/MM/yyyy hh:mm:ss</ConversionMask> <Culture>en-GB</Culture> </ImportMapping> <!-- Refer to DateTime.ParseExact on MSDN for more information on ConversionMask (format) and culture (iformat provider)-->
Lookup	Take the value of SourceColumn. Find that value in the LookUpList. Set DestinationField to the id of that item. <ImportMapping xsi:type="ImportMapping_Lookup"> <!-- This is the field of type lookup.--> <DestinationField>Job</DestinationField> <SourceColumn>Job Title</SourceColumn> <!-- This is the list that contains the lookup values --> <LookupListTitle>Items</LookupListTitle> <!-- This is the field in the lookup list the value of which matches the value in the source column--> <LookupFieldInternalName>Title</LookupFieldInternalName> <!-- This is the CAML value type Text,Number,DateTime,Guid,MultiChoice,Lookup for the field on in the lookup list that matches the value specified in the source column --> <LookupFieldCAMLType>Text</LookupFieldCAMLType> </ImportMapping>
ManagedMetaDataAutoAdd	Find the term set which DestinationField is linked to. Find a term in that set matching the value in SourceColumn. Set DestinationField to that term id. If the term does not exist DIFS will try and create it.

	"Andrews, Robert" would become.  If the term set is nested use, for example, a source value of "Managers\Andrews, Robert" or "UK\Staff\Managers\ Andrews, Robert" May not work for SharePoint Online. <ImportMapping xsi:type="ImportMapping_ManagedMetaDataAutoAdd"> <DestinationField>Employee Name</DestinationField> <SourceColumn>Employee Name</SourceColumn> </ImportMapping>
ManagedMetaDataCSOM	Find the term set which DestinationField is linked to. Find a term in that set matching the value in SourceColumn. Set DestinationField to that term id. Supports Single and Multi-value fields. Multi-value fields should be separated by a semi colon. The mapping will find the term (e.g. SQL) anywhere in a nested term set and will set the item to the value of the first matching term it finds. Will not work on SharePoint 2010. Title | Employee Number | Employee Name | Skills 1002 | 1002 | Adams, David | SharePoint 1001 | 1001 | Andrews, Robert | SharePoint;SQL Would become;  <ImportMapping xsi:type="ImportMapping_ManagedMetaDataCSOM">

	<DestinationField>Skills</DestinationField> <SourceColumn>Skills</SourceColumn> </ImportMapping>
Native	Get the value of SourceColumn and set destination with no conversion. This assumes that the source and destination have the same data type, e.g.both Numeric or both DateTime or that they are close enough that .Net can automatically cast the value, e.g. Integer to Numeric. <ImportMapping xsi:type="ImportMapping_Native"> <DestinationField>EmployeeNumber</DestinationField> <SourceColumn>Employee Number</SourceColumn> </ImportMapping>
User	Get the value of SourceColumn, get the user object for that person in SharePoint. Set DestinationField (which must be a person field of course) to that value. The value of Source could be "Bob Smith" or his email eg Bob.Smith@company.onmicrosoft.com should also work. <ImportMapping xsi:type="ImportMapping_User"> <DestinationField>Client Manager</DestinationField> <SourceColumn>ClientManager</SourceColumn> </ImportMapping>

6.3 SourceColumns

This section defines the columns that appear in the source data that tell DIFS how to create the item in SharePoint.

In all instances the value of the setting is the name of the column in the source data.

I.e.

```
<SourceFileNameAndPath>FullName</SourceFileNameAndPath>
```

Infers…

FullName	ContentType
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version1.txt	Document
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version2.txt	Document
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version3.txt	Document

The columns only need to be in your source data set if they are Mandatory. So for example SourceFileNameAndPath is mandatory for importing documents but not folders or items.

6.3.1 SourceFileNameAndPath

The name of the column in the source data which provides the full path to the file being imported.

Mandatory for documents.

6.3.2 ContentType

The name of the column in the source data which provides the name of the content type to set the item to.

Mandatory.

6.3.3 DestinationSubFolder

The name of the column in the source data which provides the sub folder into which to import the item.

6.3.4 DestinationFileName

The name of the column in the source data which provides the destination file name / folder name of the item to create in SharePoint. The values in that column cannot include any characters which are invalid for SharePoint such as ?!\ etc.

Mandatory for documents and folders.

6.3.5 Publish

The name of the column in the source data which contains the value “Yes” when the item is to be published. This column is particularly useful if you are trying to produce a specific version history in SharePoint.

To use the Publish column the destination library must have minor versions enabled.

The ItemExistsBehaviour must be set to overwrite.

Working with this configuration:

```
<SourceColumns>
  <SourceFileNameAndPath>FullName</SourceFileNameAndPath>
  <ContentType>ContentType</ContentType>
  <DestinationSubFolder>DestinationSubDirectories</DestinationSubFolder>
  <DestinationFileName>DestinationFileName</DestinationFileName>
  <Publish>Publish</Publish>
</SourceColumns>
```

And with this source data:

FullName	ContentType	Title	DestinationFileName	Publish
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version1.txt	Document	Version 1	TestA.txt	FALSE
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version2.txt	Document	Version 2	TestA.txt	TRUE
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version3.txt	Document	Version 3	TestA.txt	FALSE
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version4.txt	Document	Version 4	TestA.txt	TRUE

You will end up with this version history:

2.0 10/11/2016 1:13 AM

Title Version 4

1.1 10/6/2016 7:00 AM

Title Version 3

1.0 10/11/2016 1:13 AM

Title Version 2

0.1 10/6/2016 6:59 AM

Title Version 1

Where the publish column is forcing the item to be published it becomes a major version.

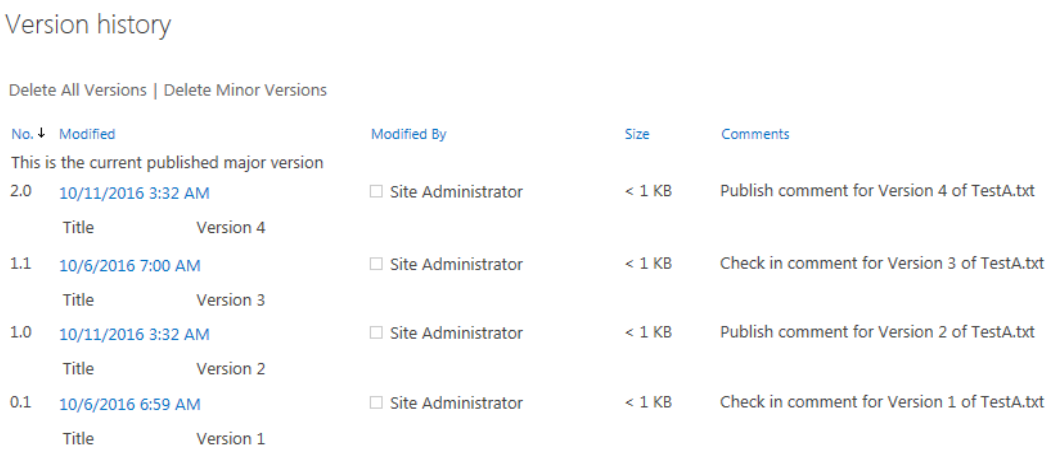
6.3.6 PublishComment

The name of the column in the source data which contains the value to use as the PublishComment.

The following source data:

FullName	ContentType	Title	DestinationFileName	Publish	PublishComment	CheckInComment
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version1.txt	Document	Version 1	TestA.txt	FALSE	Publish comment for Version 1 of TestA.txt	Check in comment for Version 1 of TestA.txt
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version2.txt	Document	Version 2	TestA.txt	TRUE	Publish comment for Version 2 of TestA.txt	Check in comment for Version 2 of TestA.txt
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version3.txt	Document	Version 3	TestA.txt	FALSE	Publish comment for Version 3 of TestA.txt	Check in comment for Version 3 of TestA.txt
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version4.txt	Document	Version 4	TestA.txt	TRUE	Publish comment for Version 4 of TestA.txt	Check in comment for Version 4 of TestA.txt

Will produce the following version history.



6.3.7 CheckInComment

The name of the column in the source data which contains the value to use as the Check InC omment.

The following source data:

FullName	ContentType	Title	DestinationFileName	Publish	PublishComment	CheckInComment
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version1.txt	Document	Version 1	TestA.txt	FALSE	Publish comment for Version 1 of TestA.txt	Check in comment for Version 1 of TestA.txt
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version2.txt	Document	Version 2	TestA.txt	TRUE	Publish comment for Version 2 of TestA.txt	Check in comment for Version 2 of TestA.txt
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version3.txt	Document	Version 3	TestA.txt	FALSE	Publish comment for Version 3 of TestA.txt	Check in comment for Version 3 of TestA.txt
\\vmware-host\Shared Folders\TestData\Generic\Versions\Version4.txt	Document	Version 4	TestA.txt	TRUE	Publish comment for Version 4 of TestA.txt	Check in comment for Version 4 of TestA.txt

Will produce the following version history.