Excel Template

In this section we will run through the Excel template and the different ways you can navigate through it and understand the references and information contained in the template, as well as how to use it to your advantage and thus save time and effort while loading a **Master Data Package** into the MES.

Helper Sheets

The first worksheets you see in the Excel template are helper sheets that give you information and are also used as configuration for the entire template. To understand what information they hold and what they are used for, see Help Sheets.

Explaining the Format

The different names and types of the sheets contained in the Excel template are described below:

Prefix	Type
No prefix	Sub-sheet of previous prefixed sheet
`<DM>`	Dynamic Model
`<SM>`	Static Model
`<GT>`	Generic Table
`<ST>`	Smart Table
`<UP>`	Update Model
`<LOOKUP>`	Reference for the longer name.

Note

An example for `<UP>` could be the following:

Name	Team	ShiftPlan	Workgroup	Position
Operator_2	Team-E	Assembly Preparation	WeighAndDispense	WeighAndDispense Certification
Operator_3	Team-E	Assembly Preparation	Mixing & Injection	Mixing Certification

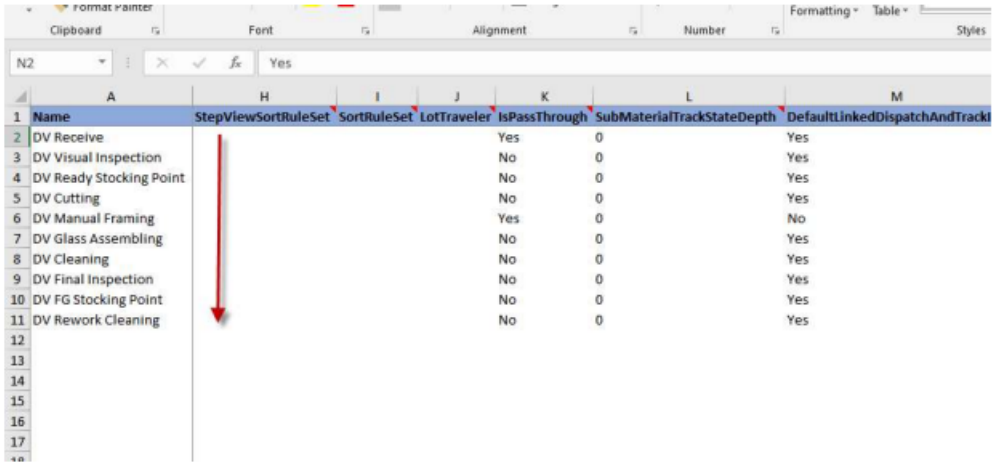
<UP>EmployeeShiftDetails

This tab is used to assign employees to a Shift Plan and then to a Workgroup of the Shift Plan.

Info

See Help Sheets for more information on the `<LOOKUP>`.

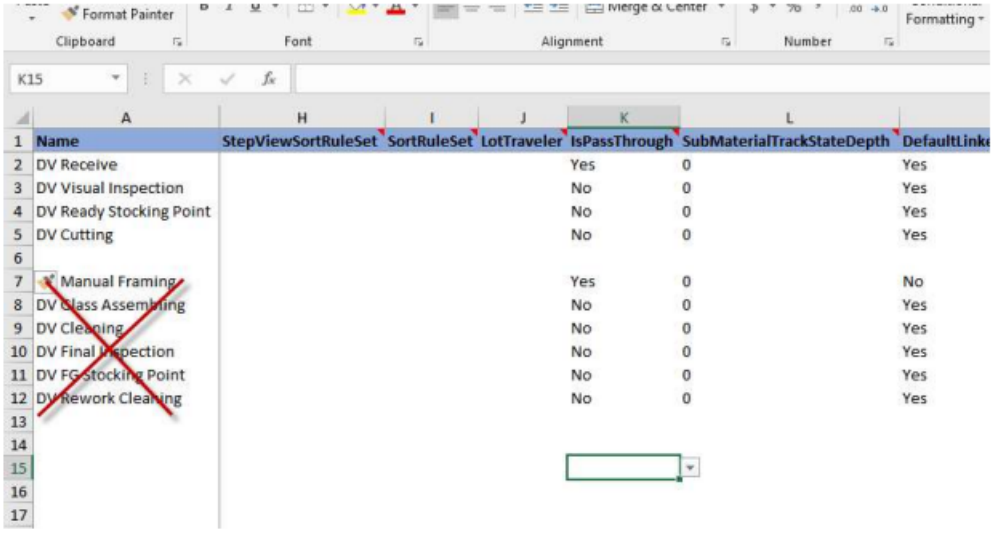
The system reads the Excel file in the order that the worksheets are placed in the file, starting from the left-most sheet and moving rightward. Each sheet is read from left to right and from top to bottom, creating fields as a linear FIFO (First-In-First-Out) sequence.



	A	H	I	J	K	L	M
1	Name	StepViewSortRuleSet	SortRuleSet	LotTraveler	IsPassThrough	SubMaterialTrackStateDepth	DefaultLinkedDispatchAndTrackI
2	DV Receive				Yes	0	Yes
3	DV Visual Inspection				No	0	Yes
4	DV Ready Stocking Point				No	0	Yes
5	DV Cutting				No	0	Yes
6	DV Manual Framing				Yes	0	No
7	DV Glass Assembling				No	0	Yes
8	DV Cleaning				No	0	Yes
9	DV Final Inspection				No	0	Yes
10	DV FG Stocking Point				No	0	Yes
11	DV Rework Cleaning				No	0	Yes

Warning

The system will stop reading if a blank line appears. You can look at each sheet as a specific entity type, with each column representing an entity property or relation and each row representing a new entity.



	A	H	I	J	K	L	
1	Name	StepViewSortRuleSet	SortRuleSet	LotTraveler	IsPassThrough	SubMaterialTrackStateDepth	DefaultLinke
2	DV Receive				Yes	0	Yes
3	DV Visual Inspection				No	0	Yes
4	DV Ready Stocking Point				No	0	Yes
5	DV Cutting				No	0	Yes
6							
7	Manual Framing				Yes	0	No
8	DV Glass Assembling				No	0	Yes
9	DV Cleaning				No	0	Yes
10	DV Final Inspection				No	0	Yes
11	DV FG Stocking Point				No	0	Yes
12	DV Rework Cleaning				No	0	Yes
13							
14							
15							
16							
17							

The name of the Excel sheets must match the MES entities being loaded and the name of the columns must match the entity properties exactly. When processing the data, the system will check if the object already exists, thus creating or updating it accordingly.

> **Note**
>
> The template does not have all the objects and fields that are available in the UI. You can, however, add more entity types (sheets) and entity properties (columns), provided they are available in the UI and exactly match the name of the entity and respective properties.

Versioned Entities

In the case of versioned entities, a new version is always created when loaded into the system. **Change Sets** are managed automatically by the tool so there is no need to create them manually either in the Excel template or in the GUI. However, the approval process for any versioned object can only be performed using the GUI and you can access any **Change Sets** currently under approval in the Change Sets.

For more information on loading entity revisions and versions, see Revision Configuration.

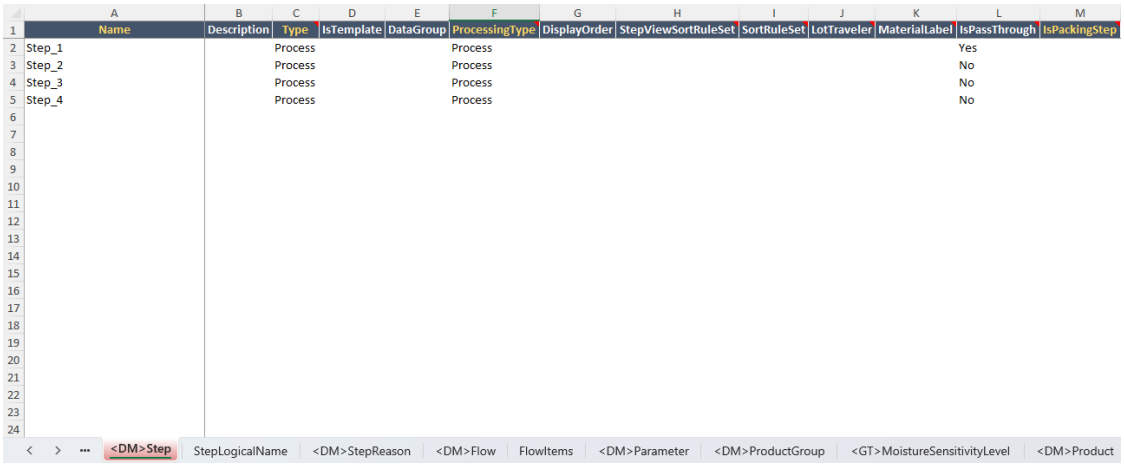
Modeling Sequence

It is highly recommended to follow an appropriate modeling sequence since there are a lot of interdependent fields that have strict rules of precedence.

Flow/Step example

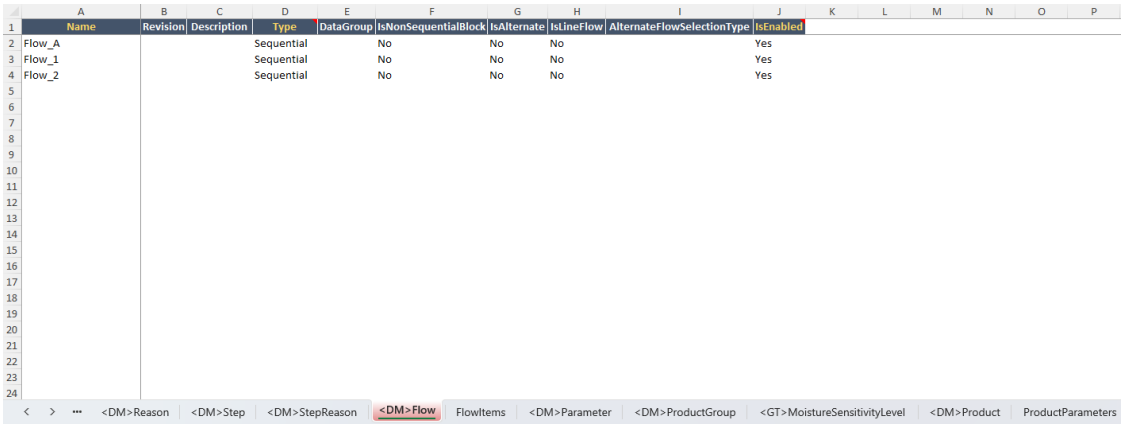
A **Flow** is composed of **Flow Items** of type **Step** and **Flow**. When a **Flow** contains **Flow Items** of type **Flow**, they are denoted as **Child Flows** of the **Parent Flow** for clarity. To create a **Flow** structure, you should follow the sequence below:

1. Create the **Steps**.



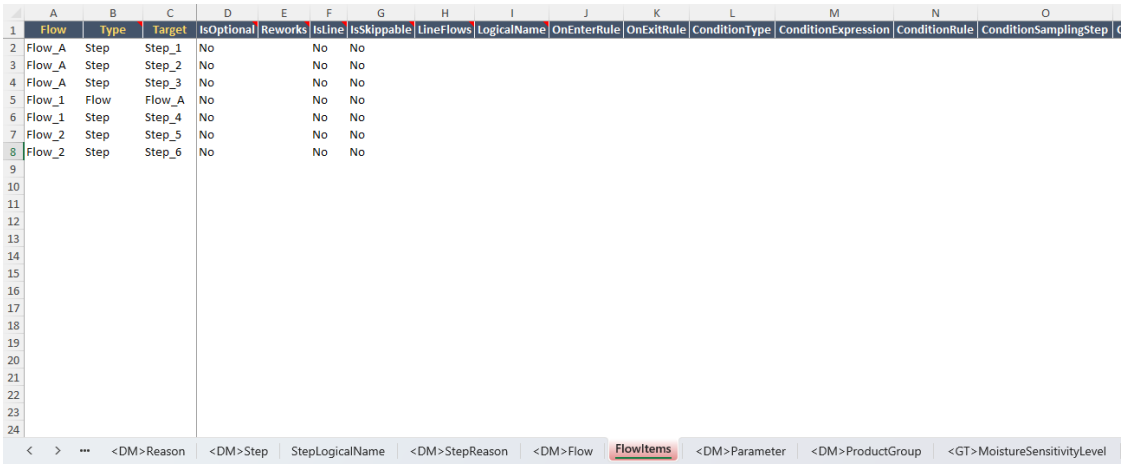
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Name	Description	Type	IsTemplate	DataGroup	ProcessingType	DisplayOrder	StepViewSortRuleSet	SortRuleSet	LotTraveler	MaterialLabel	IsPassThrough	IsPackingStep
2	Step_1		Process			Process						Yes	
3	Step_2		Process			Process						No	
4	Step_3		Process			Process						No	
5	Step_4		Process			Process						No	

2. Create the **Flows**.



Name	Revision	Description	Type	DataGroup	IsNonSequentialBlock	IsAlternate	IsLineFlow	AlternateFlowSelectionType	IsEnabled
Flow_A			Sequential		No	No	No		Yes
Flow_1			Sequential		No	No	No		Yes
Flow_2			Sequential		No	No	No		Yes

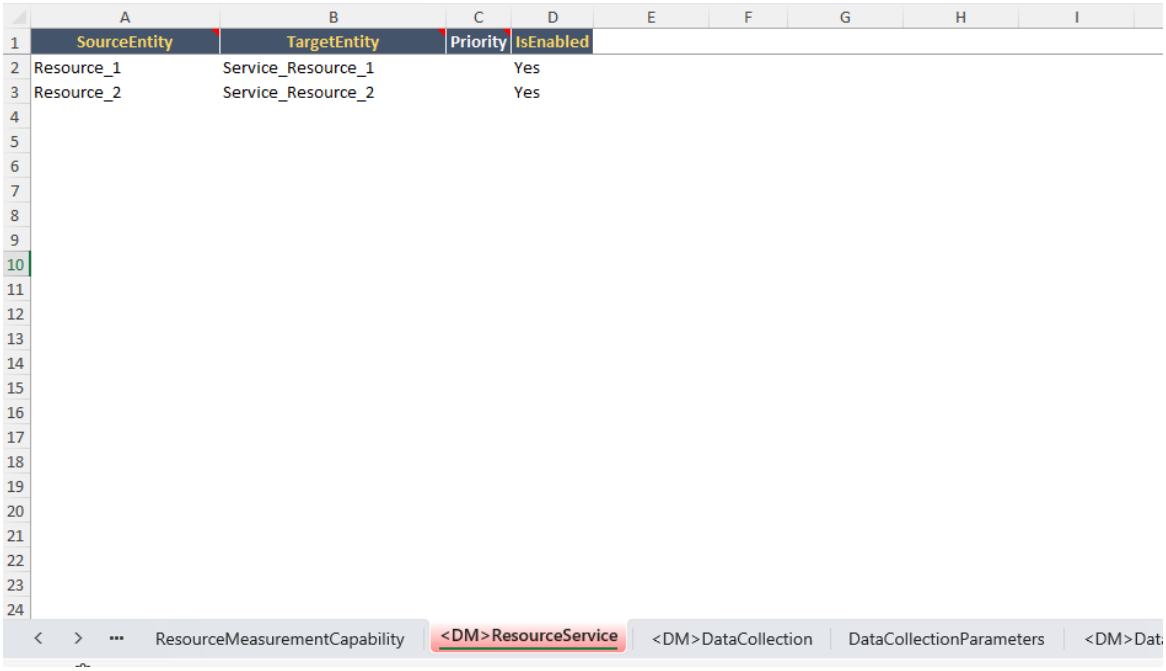
3. Utilize the **FlowItems** sheet to create the desired **Flow** structure.



Flow	Type	Target	IsOptional	Reworks	IsLine	IsSkippable	LineFlows	LogicalName	OnEnterRule	OnExitRule	ConditionType	ConditionExpression	ConditionRule	ConditionSamplingStep
Flow_A	Step	Step_1	No		No	No								
Flow_A	Step	Step_2	No		No	No								
Flow_A	Step	Step_3	No		No	No								
Flow_1	Flow	Flow_A	No		No	No								
Flow_1	Step	Step_4	No		No	No								
Flow_2	Step	Step_5	No		No	No								
Flow_2	Step	Step_6	No		No	No								

Resource/Service example

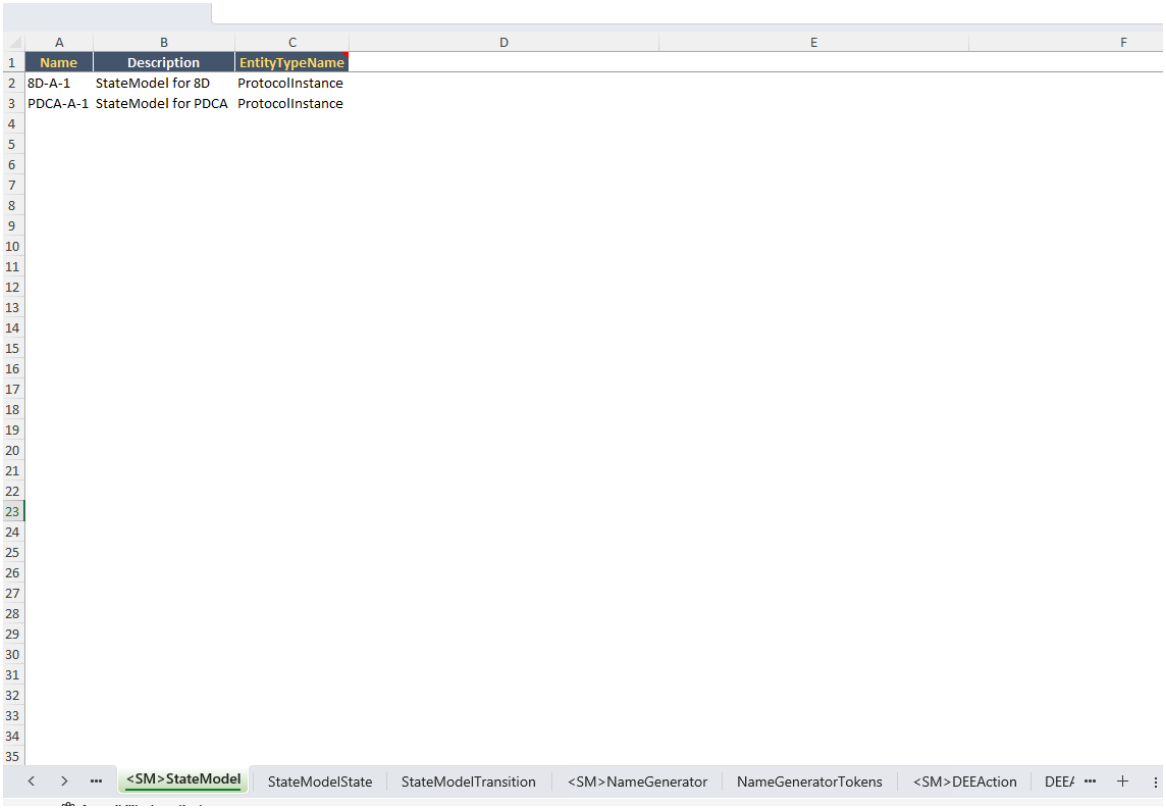
In another quick example, the **Service** must be created before the **Resource** creation and only after it is possible to fulfill the **ResourceService** sheet.



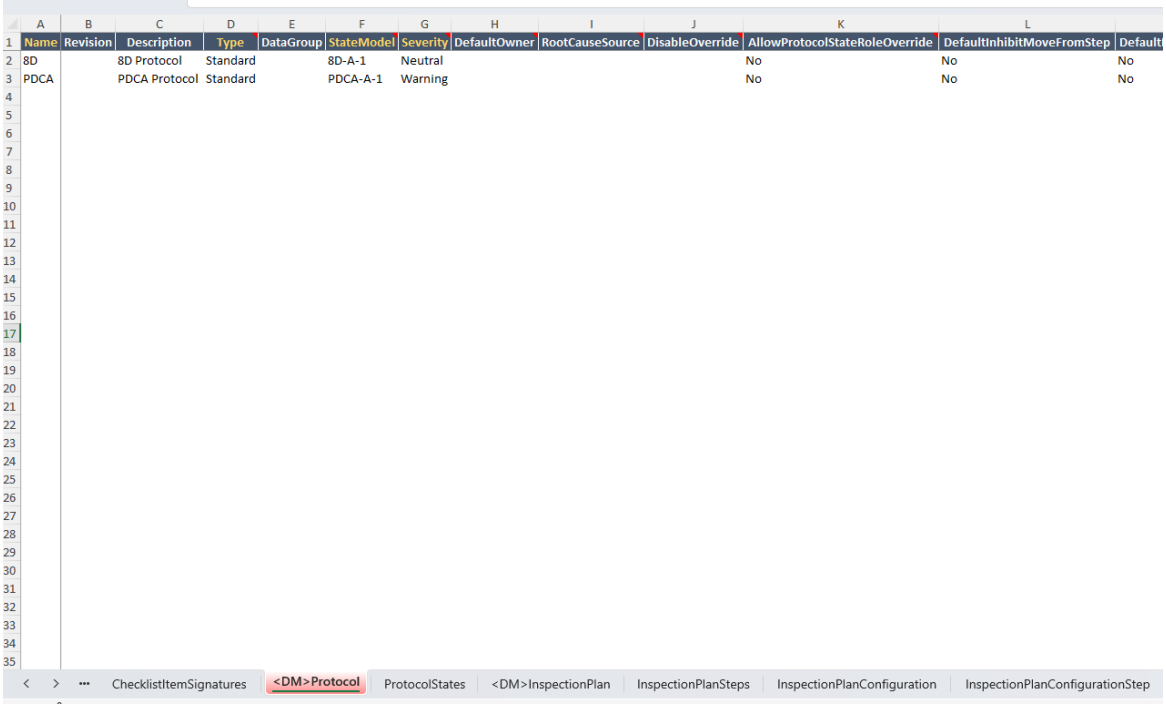
SourceEntity	TargetEntity	Priority	IsEnabled
Resource_1	Service_Resource_1		Yes
Resource_2	Service_Resource_2		Yes

State Model/Protocol example

In the case of **Protocols**, you must create the **State Model** and load both the **State Model** and the **Protocol** at the same time.



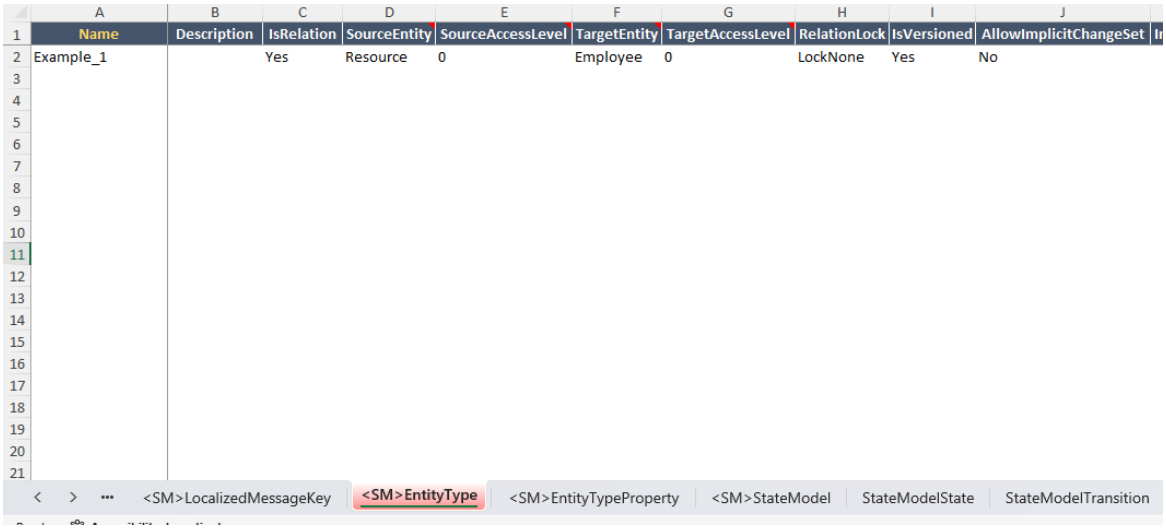
Name	Description	EntityTypeName
8D-A-1	StateModel for 8D	ProtocolInstance
PDCA-A-1	StateModel for PDCA	ProtocolInstance



Name	Revision	Description	Type	DataGroup	StateModel	Severity	DefaultOwner	RootCauseSource	DisableOverride	AllowProtocolStateRoleOverride	DefaultInhibitMoveFromStep	Default
8D		8D Protocol	Standard		8D-A-1	Neutral				No	No	No
PDCA		PDCA Protocol	Standard		PDCA-A-1	Warning				No	No	No

Relations

Most of the relations between objects are added automatically when two objects require contextual sharing of information due to their current operation state. You can manually add relations between objects by using the **EntityType** sheet and filling in the **SourceEntity** and the specific **TargetEntity**.



Name	Description	IsRelation	SourceEntity	SourceAccessLevel	TargetEntity	TargetAccessLevel	RelationLock	IsVersioned	AllowImplicitChangeSet
Example_1		Yes	Resource	0	Employee	0	LockNone	Yes	No

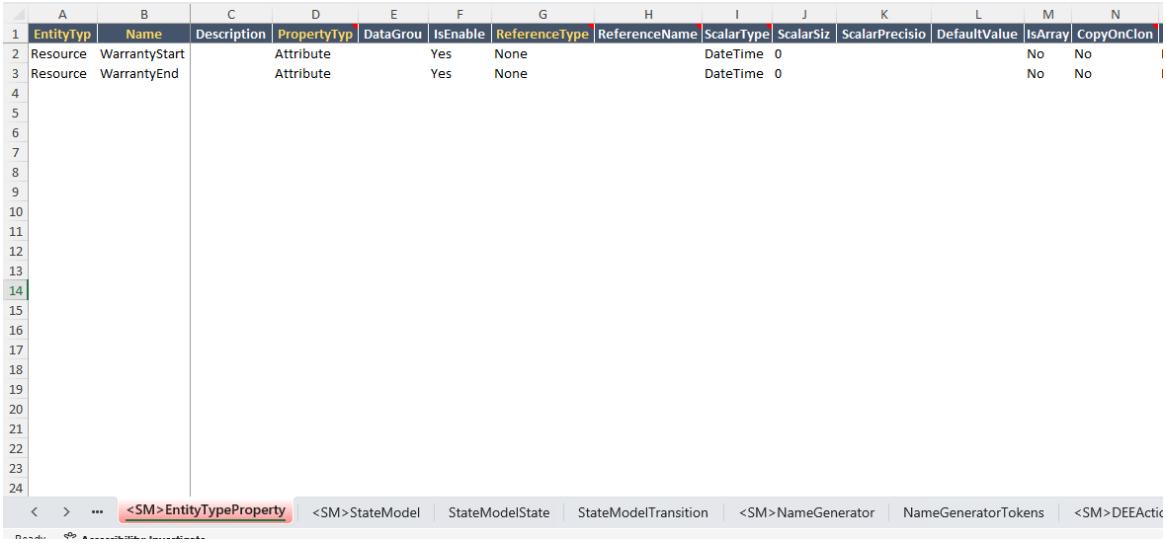
Note

If you want to create a relation between versioned entities, use the **EntityTypeProperty** sheet to define whether the Source or Target entities belong to the version, to the definition or to the complete entity. The **Reference Type** field in that sheet defines that relationship and has three possible options:

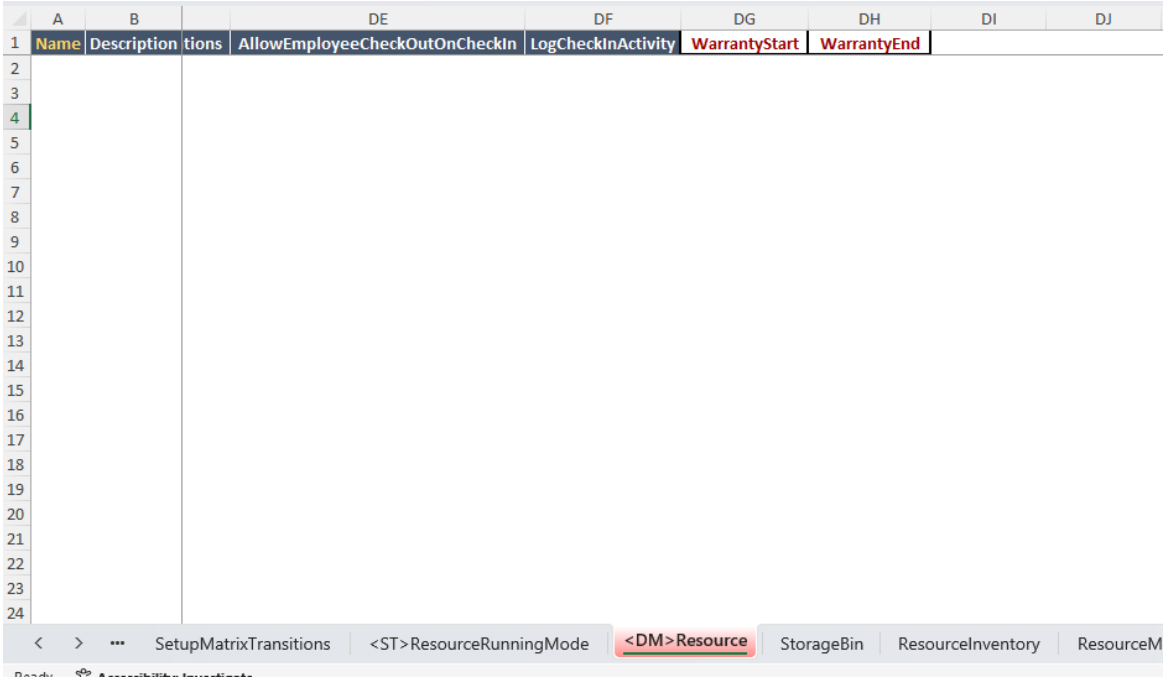
- `EntityDefinition`
- `EntityVersion`
- `Entity`

Attributes

If you want to add specific attributes to an entity, you must create the attribute in the **EntityTypeProperty** sheet and then add the columns with the attribute names in the target Entity.



EntityTyp	Name	Description	PropertyTyp	DataGrou	IsEnable	ReferenceType	ReferenceName	ScalarType	ScalarSiz	ScalarPrecisio	DefaultValue	IsArray	CopyOnClon
Resource	WarrantyStart		Attribute		Yes	None		DateTime	0			No	No
Resource	WarrantyEnd		Attribute		Yes	None		DateTime	0			No	No



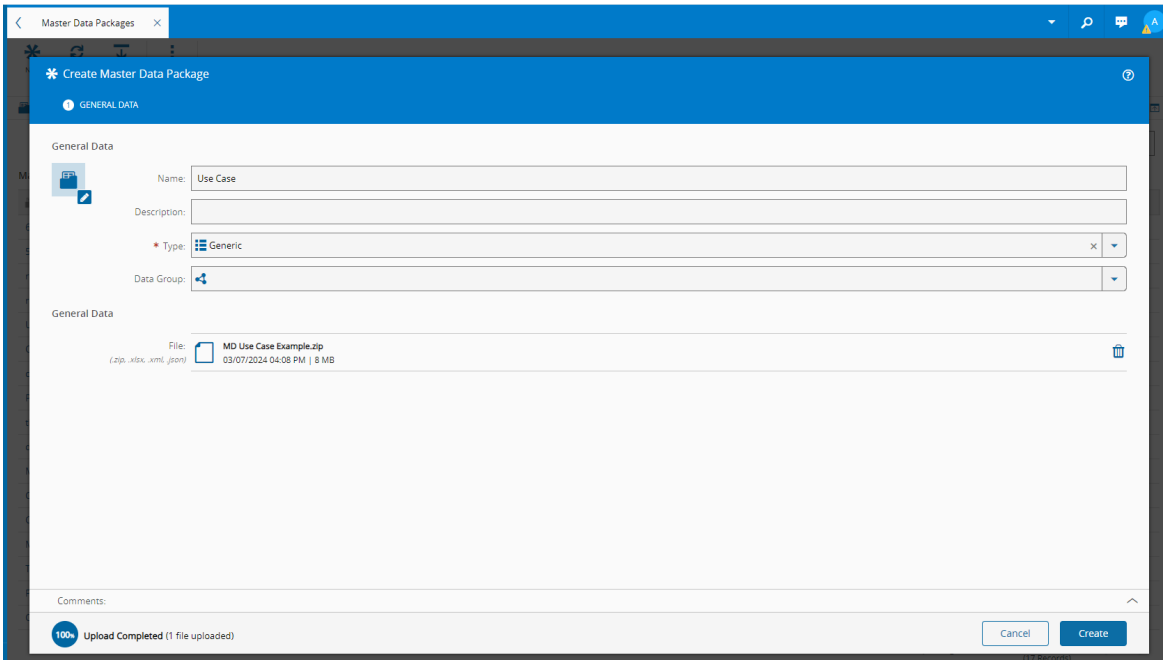
Warning

The empty spaces in the entity names are taken into consideration and can cause failures when loading the **Master Data Package**. Make sure you trim all leading and lagging spaces from the labels of any data element.

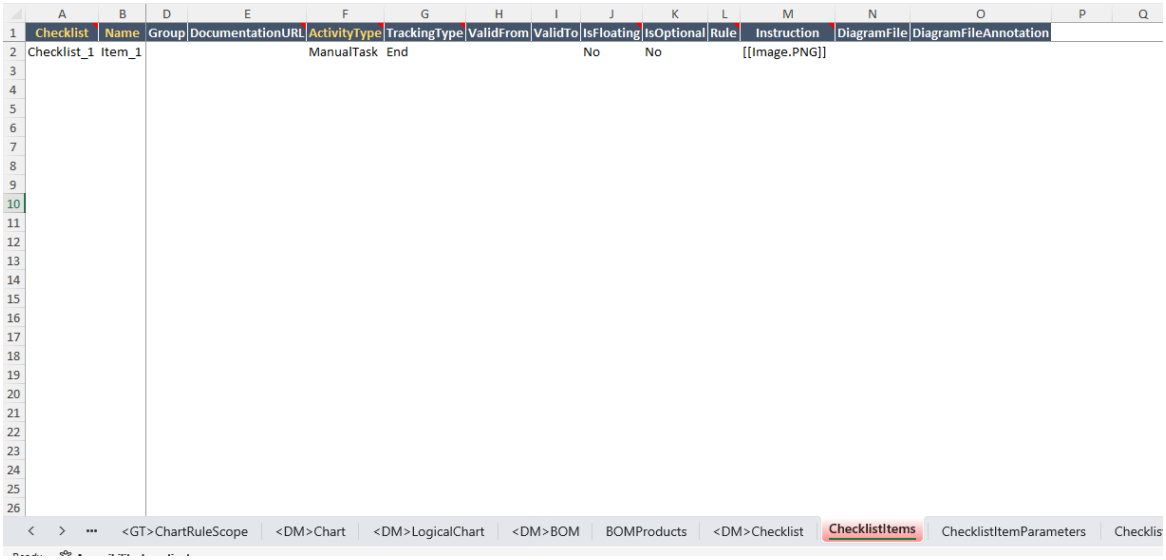
External Files

You can upload specific types of external files that will be included in the **Master Data Package**. They must be specified manually and included in a `.zip` file that will be used to upload the data into the system:

- Images - used in **Checklists**, **Data Collections** and Notes in Future Actions, among others.
- Objects - system objects like **Queries**, **Printable Documents** or **FabLives**, among others.
- Documents - files can be associated to most entity types, like PDFs or other files that can be displayed in the system.

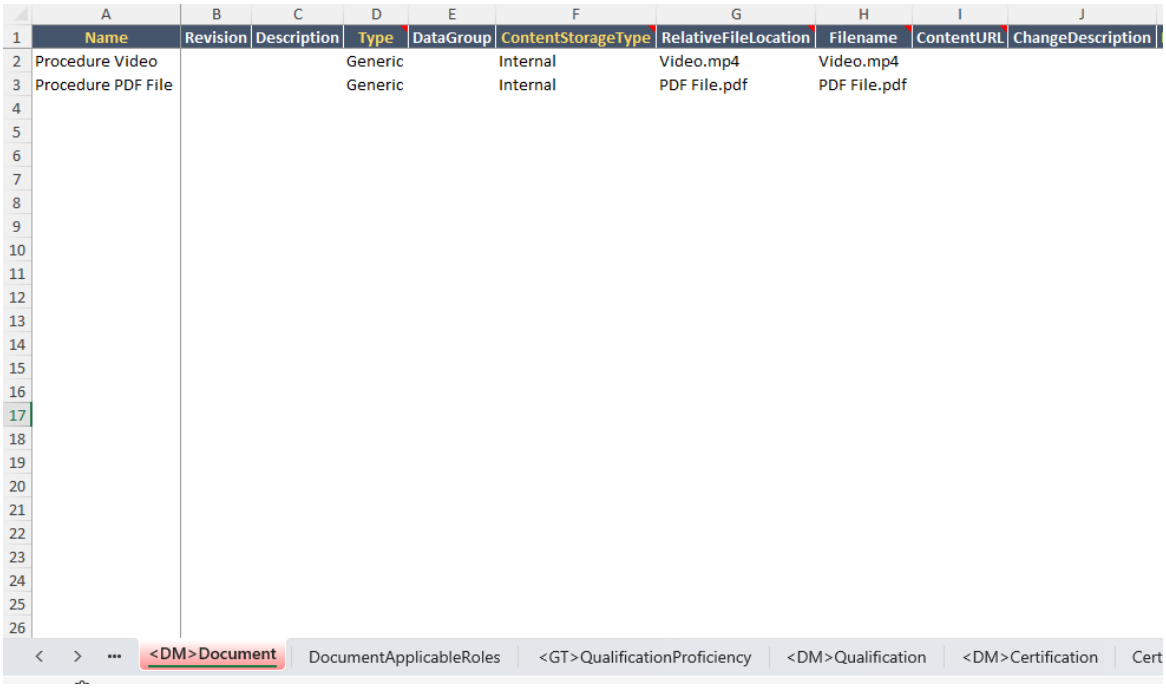


To reference the files in a **Checklist**, you must delimit the file name and the specific format with `[[` and `]]`, like the example below:



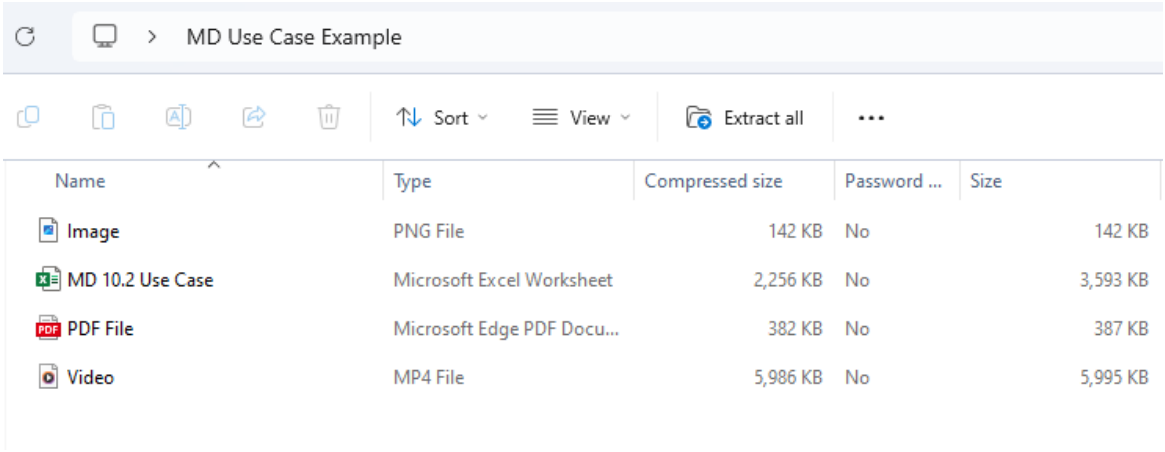
Checklist	Name	Group	DocumentationURL	ActivityType	TrackingType	ValidFrom	ValidTo	IsFloating	IsOptional	Rule	Instruction	DiagramFile	DiagramFileAnnotation
Checklist_1	Item_1			ManualTask	End			No	No		[[Image.PNG]]		

For **Document** loading the user must fill the appropriate values for `RelativeFileLocation` and `Filename`, the latter with the name of the object that will be included in the `.zip` file.



Name	Revision	Description	Type	DataGroup	ContentStorageType	RelativeFileLocation	Filename	ContentURL	ChangeDescription
Procedure Video			Generic		Internal	Video.mp4	Video.mp4		
Procedure PDF File			Generic		Internal	PDF File.pdf	PDF File.pdf		

The content of a `.zip` file that reflects the model shown above would look like this:



Name	Type	Compressed size	Password ...	Size
Image	PNG File	142 KB	No	142 KB
MD 10.2 Use Case	Microsoft Excel Worksheet	2,256 KB	No	3,593 KB
PDF File	Microsoft Edge PDF Docu...	382 KB	No	387 KB
Video	MP4 File	5,986 KB	No	5,995 KB