precisely

Precisely EnterWorks

EnterWorks EDI Preprocessing Guide

Version 10.4.8

Notices

Trademarks

"EnterWorks" and the "EnterWorks" logo are registered trademarks and "Enable PIM", "EnterWorks PIM", "EnterWorks Process Exchange" and "EnterWorks Product Information Management" are trademarks of Precisely.

Third-party Acknowledgments

Windows, .NET, IIS, SQL Server, Word, and Excel are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Java and all Sun and Java based trademarks are trademarks or registered trademarks of the Oracle Corporation in the United States and other countries.

Oracle is a registered trademark and Oracle 10g is a trademark of Oracle Corporation.

Pentium is a registered trademark of Intel Corporation in the United States and other countries.

JBoss is a registered trademark of Red Hat, Inc.

All other trademarks and registered trademarks are the property of their respective holders.

All icons and graphics, with the exception of the "e." logo, were obtained from West Coast Icons and Design at http://www.bywestcoast.com.

EnterWorks

Contents

EnterWorks

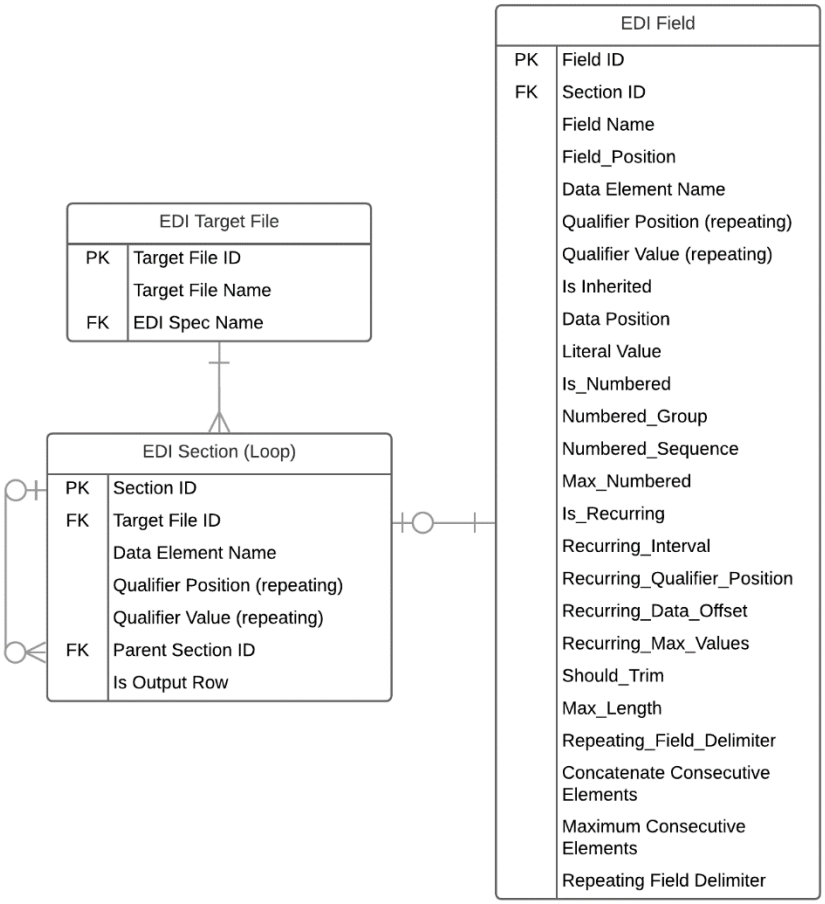
Overview

This document details the preprocessing of EDI files and converting them into files that can be imported into Enable using Import Templates.

EDI files are comprised of a sequence of Data Elements, with each Data Element having a sequence of fields (by position). Some or all these fields need to be transformed into tabular form for import into Enable. The processing that accomplishes this reads each data element sequentially, extracting the values from one or more fields in each data element, placing those values into a “staging bucket” and at appropriate times, writes the contents of the “staging bucket” to the corresponding target file. The specific behavior for an EDI file as to what Data Elements are expected, what fields to extract, what target files to create and when to write records to those files is defined in an EDI Specification.

EDI Preprocessing Specification

The following data model diagram defines the contents of an EDI Specification with each entity described in detail in subsequent sections:



EDI SpecName

Identifies a specification. The model and processing support defining multiple specifications. When the preprocessing is invoked, the EDI SpecName, source file name, and target file base name are specified.

Qualified Data Element

Several of the specification objects include what is referred to as a Qualified Data Element. This is a data element that has been identified using the following information:

- **Data_Element_name** – this is always required
- **Qualifier_Position/Qualifier_Value** – optional position/value pairs of data values from the EDI Data Element that must match the specification definition. If none are specified, then the Data Element name matching is sufficient.
- **Section_ID** – identifies the EDI Section in which the Data Element is expected. If the current EDI Section is different, the Data Element is not qualified. When the EDI data file is processed, the preprocessing keeps track of what the Current EDI Section is so that when comparisons are made to Qualified Data Elements, only matching sections will be applied. The Section ID must be defined.

EDI Target File

Each EDI file will be preprocessed and converted into one or more target files. Each target file is defined by a linked EDI Target File record. The **Target File Name** defines the base name for the file. Each preprocessing job may prepend this file name with a unique identifier for the job (so each file will have the same prefix). Each EDI Specification (identified by **EDI Spec Name**) will generate one or more target files. Each target file will be processed independently from the other target files but in parallel (one pass through the source EDI Document)

EDI Section (Loop)

Some Data Elements with the same qualifiers may appear multiple times in an EDI document but need to be mapped to different target fields. Each EDI Section may repeat in the EDI File (e.g., multiple products or multiple SKUs). To isolate the common Data Elements, the EDI file can be partitioned through the definition of sections. These sections may be nested. For example, a Product section may have multiple SKU sections. The start of each section is denoted by a matching Qualified Data Element within the context of the current EDI section. For example, if Products contain multiple SKUs and both Products and SKUs have measurements, then a MEA data element for Height would be attributed to the Product if the current section is Product or to the SKU if the current section is SKU.

The EDI Sections are hierarchical in nature. Except for the top EDI Section, each subsequent EDI Section must be a child of another EDI Section. Multiple EDI Sections can have the same parent EDI Section.

Example 1 – an EDI file with no looping and no duplicate data elements can be expressed with a single EDI Section containing one of the first Data Elements found in the EDI File:

- EDI Section - Data Element Name = GS, Is Output Row = Yes

File:

```
ISA*00*          *00*          *14*2143981411     *12*314493968901
*180610*1239*U*00401*000895068*0*P*>
GS*SC*2143981411*314493968901*20180610*1239*1049331*X*00401
ST*832*1049331
BCT*PS**2
REF*11*314493968901
CUR*SE*USD
LIN**GS*S0931F15373430*MF*DalTile*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field Tile
Floor 18x18 PK18.00
DTM*197*20180610
REF*19**DalTile
PID*F*TRN***Heathland Field Tile Floor 18x18 PK18.00
PID*F*MAC***CERFLO
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
PID*F*CO***Heathland
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
MEA**SU*18*SF
CTP**LPR*1.11*1*SF
DTM*007*20180604
SLN*1**O******SK*HL0218181P2***MG*HL02*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field
Tile Floor 18x18 PK18.00
PID*F*73***RAFFIA Unpolished
PID*F*35***HL02UP
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
CTP**LPR*1.11*1*SF
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
LIN**GS*S0222F04110650*MF*DalTile*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco Accent
12x12
DTM*197*20180609
REF*19**DalTile
PID*F*TRN***Salerno Deco Accent 12x12
PID*F*MAC***CERWAL
PID*F*PNA***Salerno Deco Accent 12x12
PID*F*CO***Salerno
MEA**LN*1.00000000*EZ
MEA**WD*1.00000000*EZ
MEA**SU*9*SF
CTP**LPR*18.77*1*SF
DTM*007*20170405
SLN*1**O******SK*SL871212DECO1P2***MG*SL87*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco
Accent 12x12
PID*F*73***UNIVERSAL Glazed
PID*F*35***SL87G
PID*F*PNA***Salerno Deco Accent 12x12
CTP**LPR*18.77*1*SF
MEA**LN*1.00000000*EZ
MEA**WD*1.00000000*EZ
LIN**GS*S0222F04110651*MF*DalTile*ST*0222***SZ*3x12*MS*S0222F04110651*MN*Salerno Deco Accent
3x12
```

Example 2 – an EDI File contains multiple Products, each having one or more SKUs:

- EDI Section - Data Element Name = GS
  - EDI Section - Data Element Name = LIN
    - EDI Section - Data Element Name = SLN, Is Output Row = Yes

Example File:

```
ISA*00*          *00*          *14*2143981411     *12*314493968901
*180610*1239*U*00401*000895068*0*P*>
GS*SC*2143981411*314493968901*20180610*1239*1049331*X*00401
ST*832*1049331
BCT*PS**2
REF*11*314493968901
CUR*SE*USD
```

```
LIN**GS*S0931F15373430*MF*DalTile*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field Tile
Floor 18x18 PK18.00
DTM*197*20180610
REF*19**DalTile
PID*F*TRN***Heathland Field Tile Floor 18x18 PK18.00
PID*F*MAC***CERFLO
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
PID*F*CO***Heathland
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
MEA**SU*18*SF
CTP**LPR*1.11*1*SF
DTM*007*20180604
SLN*1**O******SK*HL0218181P2***MG*HL02*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field
Tile Floor 18x18 PK18.00
PID*F*73***RAFFIA Unpolished
PID*F*35***HL02UP
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
CTP**LPR*1.11*1*SF
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
LIN**GS*S0222F04110650*MF*DalTile*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco Accent
12x12
DTM*197*20180609
REF*19**DalTile
PID*F*TRN***Salerno Deco Accent 12x12
PID*F*MAC***CERWAL
PID*F*PNA***Salerno Deco Accent 12x12
PID*F*CO***Salerno
MEA**LN*1.00000000*EZ
MEA**WD*1.00000000*EZ
MEA**SU*9*SF
CTP**LPR*18.77*1*SF
DTM*007*20170405
SLN*1**O******SK*SL871212DECO1P2***MG*SL87*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco
Accent 12x12
PID*F*73***UNIVERSAL Glazed
PID*F*35***SL87G
PID*F*PNA***Salerno Deco Accent 12x12
CTP**LPR*18.77*1*SF
MEA**LN*1.00000000*EZ
MEA**WD*1.00000000*EZ
SLN*1**O******SK*MH0612241P6***MG*MH06*ST*1188***SZ*12x24*MS*S1188F27456177*MN*Modern Hearth
Field Tile Floor 12x24
PID*F*73***CHIMNEY CORNER Unpolished
PID*F*35***MH06UP
PID*F*PNA***Modern Hearth Field Tile Floor 12x24
CTP**LPR*4.43*1*SF
MEA**LN*2.00000000*EZ
MEA**WD*1.00000000*EZ
LIN**GS*S0222F04110651*MF*DalTile*ST*0222***SZ*3x12*MS*S0222F04110651*MN*Salerno Deco Accent
3x12
```

Example 3 – an EDI File contains multiple Products, each having one or more SKUs, with each SKU having 3 sets of Measurements (e.g, Package, Each, and Case):

- EDI Section - Data Element Name = GS
  - EDI Section - Data Element Name = LIN
    - EDI Section - Data Element Name = SLN, Is Output Row = Yes
      - EDI Section - Data Element Name = PID (F, CAS)
      - EDI Section - Data Element Name = PID (F, PKG)
      - EDI Section - Data Element Name = PID (F, EA)

In this example, there are 3 sibling EDI Sections below the EDI Section designated Is Output Row. This means that when these sections are encountered, they do not trigger outputting any data to the file.

Example File:

```
ISA*00*          *00*          *14*2143981411     *12*314493968901
*180610*1239*U*00401*000895068*0*P*>
GS*SC*2143981411*314493968901*20180610*1239*1049331*X*00401
ST*832*1049331
BCT*PS**2
REF*11*314493968901
CUR*SE*USD
LIN**GS*S0931F15373430*MF*DalTile*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field Tile
Floor 18x18 PK18.00
DTM*197*20180610
REF*19**DalTile
PID*F*TRN***Heathland Field Tile Floor 18x18 PK18.00
PID*F*MAC***CERFLO
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
PID*F*CO***Heathland
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
MEA**SU*18*SF
CTP**LPR*1.11*1*SF
DTM*007*20180604
SLN*1**O******SK*HL0218181P2***MG*HL02*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field
Tile Floor 18x18 PK18.00
PID*F*73***RAFFIA Unpolished
PID*F*35***HL02UP
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
CTP**LPR*1.11*1*SF
PID*F*CAS***HL02UP
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
PID*F*PKG***HL02UP
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
PID*F*EA***HL02UP
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
LIN**GS*S0222F04110650*MF*DalTile*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco Accent
12x12
```

Example 4 – Same as example 3 except the multiple sets of dimensions are at the Product level:

- EDI Section - Data Element Name = GS
  - EDI Section - Data Element Name = LIN
    - EDI Section - Data Element Name = PID (F, CAS)
    - EDI Section - Data Element Name = PID (F, PKG)
    - EDI Section - Data Element Name = PID (F, EA)
    - EDI Section - Data Element Name = SLN, Is Output Row = Yes

In this example, since the three PID EDI Sections do not have a child EDI Section with Is Output Row = Yes, they still don't trigger the output of a data row.

Example File:

```
ISA*00*          *00*          *14*2143981411     *12*314493968901
*180610*1239*U*00401*000895068*0*P*>
GS*SC*2143981411*314493968901*20180610*1239*1049331*X*00401
ST*832*1049331
BCT*PS**2
```

```
REF*11*314493968901
CUR*SE*USD
LIN**GS*S0931F15373430*MF*DalTile*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field Tile
Floor 18x18 PK18.00
DTM*197*20180610
REF*19**DalTile
PID*F*TRN***Heathland Field Tile Floor 18x18 PK18.00
PID*F*MAC***CERFLO
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
PID*F*CO***Heathland
PID*F*CAS***HL02UP
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
PID*F*PKG***HL02UP
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
PID*F*EA***HL02UP
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
MEA**SU*18*SF
CTP**LPR*1.11*1*SF
DTM*007*20180604
SLN*1**O******SK*HL0218181P2***MG*HL02*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field
Tile Floor 18x18 PK18.00
PID*F*73***RAFFIA Unpolished
PID*F*35***HL02UP
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
CTP**LPR*1.11*1*SF
LIN**GS*S0222F04110650*MF*DalTile*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco Accent
12x12
```

Example 5 – An EDI File containing multiple Products, each having one or more SKUs but the output file is for Products only.

- EDI Section - Data Element Name = GS
  - EDI Section - Data Element Name = LIN, Is Output Row = Yes

Example File:

```
ISA*00*          *00*          *14*2143981411     *12*314493968901
*180610*1239*U*00401*000895068*0*P*>
GS*SC*2143981411*314493968901*20180610*1239*1049331*X*00401
ST*832*1049331
BCT*PS**2
REF*11*314493968901
CUR*SE*USD
LIN**GS*S0931F15373430*MF*DalTile*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field Tile
Floor 18x18 PK18.00
DTM*197*20180610
REF*19**DalTile
PID*F*TRN***Heathland Field Tile Floor 18x18 PK18.00
PID*F*MAC***CERFLO
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
PID*F*CO***Heathland
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
MEA**SU*18*SF
CTP**LPR*1.11*1*SF
DTM*007*20180604
SLN*1**O******SK*HL0218181P2***MG*HL02*ST*0931***SZ*18x18*MS*S0931F15373430*MN*Heathland Field
Tile Floor 18x18 PK18.00
PID*F*73***RAFFIA Unpolished
PID*F*35***HL02UP
PID*F*PNA***Heathland Field Tile Floor 18x18 PK18.00
CTP**LPR*1.11*1*SF
MEA**LN*1.50000000*EZ
MEA**WD*1.50000000*EZ
```

```
LIN**GS*S0222F04110650*MF*DalTile*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco Accent
12x12
DTM*197*20180609
REF*19**DalTile
PID*F*TRN***Salerno Deco Accent 12x12
PID*F*MAC***CERWAL
PID*F*PNA***Salerno Deco Accent 12x12
PID*F*CO***Salerno
MEA**LN*1.00000000*EZ
MEA**WD*1.00000000*EZ
MEA**SU*9*SF
CTP**LPR*18.77*1*SF
DTM*007*20170405
SLN*1**O******SK*SL871212DECO1P2***MG*SL87*ST*0222***SZ*12x12*MS*S0222F04110650*MN*Salerno Deco
Accent 12x12
PID*F*73***UNIVERSAL Glazed
PID*F*35***SL87G
PID*F*PNA***Salerno Deco Accent 12x12
CTP**LPR*18.77*1*SF
MEA**LN*1.00000000*EZ
MEA**WD*1.00000000*EZ
LIN**GS*S0222F04110651*MF*DalTile*ST*0222***SZ*3x12*MS*S0222F04110651*MN*Salerno Deco Accent
3x12
```

If the **Is_Output_Row** flag is set for the section, then when the data element for this section is encountered, it will trigger an output of a data row and clearing of field values, providing that data has been collected for this section. If a data element matches a parent section, then a data row is output providing data has been collected for the section with the Output Row flag, then data is cleared from the Data Row section and each parent section up to and including the section identified by the current data element.

For example, suppose the EDI specification for **Example 2** above is going to generate an output file at the SKU level. When the first LIN data element is encountered, the processing will check if there is any data collected for the SLN section. Since none has, no row is output. Similarly when the first SLN data element is encountered, the processing checks to see if any SLN data has been collected – since none has, no row is output. If a subsequent element is SLN, then when the processing checks to see if SLN section data has been collected and will find that it has, so a data row is output and the data fields in the SLN section (and any child sections) are cleared. If an LIN data element is encountered, and the processing finds data in the SLN section then a row is output and the SLN and LIN data fields are cleared. If after encountering an LIN section (and the previous LIN and SLN data is output), another LIN data element is encountered, since no SLN data has been collected, no data row is generated. But the previous Product section data is cleared before saving the data for the current LIN record.

When comparing the next Data Element to the specification, the processing will first check if the Data Element matches the Qualified Data Element for the current section (indicating a loop), each of its parent sections, and then any child sections. After checking for matching sections, the processing will compare against the EDI Fields defined for the current section as well as any EDI field in a parent section having the Is Inherited flag set. This alleviates the need for all section elements to precede a nested loop but also provides a mechanism for distinguishing between an otherwise identical Qualified Data Element at the parent vs. child section level.

EDI Field

The columns of each EDI Section within the same EDI Target File are defined by a collection of EDI Field records. The **Field_Name** defines the column name in the target file. The **Section_ID**, **Data_Element_Name**, **Qualifier_Position**, and **Qualifier_Value** comprise the Qualified Data Element which is used to identify if the source EDI record contains the value to be mapped.

For example, the MEA data element defines a measurement. The second position value is the Measurement Qualifier and may have the value "LN" for "Standard Length". So, to define the EDI Field named "Width", the **Section_ID** would be set to the ID of the appropriate section, the **Data_Element_Name** would be set to "MEA", the **Qualifier_Position** would be set to "2" and the **Qualifier_Value** would be set to "LN". When each MEA data element is processed, only the one with the LN qualifier found in the current EDI Section (or for an inherited EDI Field from a parent section) will be stored in the Width field. More than one qualifier can be specified with the **Qualifier_Position** and **Qualifier_Value** containing a matching delimited list of entries. The Qualifier fields will be empty if there is not a qualifier for the **Data_Element_Name**. For recurring fields (see **Is_Recurring** below) the Qualifier_Position is relative to the **Recurring_Qualifier_Position** and **Recurring_Interval**.

If an EDI specification has two "MEA" "LN" data elements that must be mapped to separate EDI Fields (e.g., Package Width and Width), then two EDI Section entries need to be defined and contain Qualified Data Elements that precede the two "MEA" data elements. Then the specification will have two EDI Fields with the **Data_Element_Name** of "MEA" and Qualifier Position of 2 and Qualifier Value of "LN" but one will point to the first EDI Section and the second will point to the second EDI Section. Then when the EDI file is processed, the Current EDI Section will be set to one and then the other. When the "MEA" "LN" data element is encountered, the value for the measurement will be mapped to the appropriate target field.

The **Is_Inherited** flag specifies that if the matching data element is found in a child EDI Section, it will still be processed for the parent EDI Section containing it.

The **Data_Position** identifies the position in the data element from which the EDI Field is populated. The first position following the Data_Element_Name is 1. For recurring fields (see **Is_Recurring** below), the **Data_Position** is relative to the **Is_Recurring_Base_Position**, with 1 being the first position.

If the **Literal_Value** is defined, then the **Data Position** is ignored, and the specified value is saved in the EDI Field instead. The value "[blank]" (without quotes) can be used to specify an empty literal value.

In cases where the output file needs to have the fields in a specific order (i.e., position-dependent format), the **Field_Position** attribute can be set to the column position, starting with the number 1. If this field is not populated, then the order of the columns will be non-deterministic.

In some cases, the EDI file may have multiple records (or occurrences) with fields representing the same data whose values need to be stored in numbered columns. For example, a file may have multiple records, each of which defines a Feature Bullet and the corresponding values need to be stored the columns Feature Bullet 1, Feature Bullet 2, etc. Such fields would be defined once in the EDI_Field repository with the **Field_Name** containing the name of the field without the number (e.g., "Feature Bullet") and the **Is_Numbered** field set to Yes. The **Max_Numbered** field specifies the total number of columns to be defined for the field, regardless of how many values are in the EDI file. For example, if a

product only has 2 values for Feature Bullet x, and the **Max_Numbered** is set to 8, then the file will contain 8 numbered columns for Feature Bullet with only the first two columns having data. If there are multiple fields that are interleaved, (e.g., Certification Agency 1, Certification Number 1, Certification Agency 2, Certification Number 2, etc.), the **Numbered_Group** attribute must be set to the same name for each field to be interleaved and the **Numbered_Sequence** attribute specifying the order those fields are to be listed. When a Numbered field is processed, no matter how each value is retrieved (e.g., from multiple EDI records, or multiple/recurring fields in a single EDI record, or a combination of both), the values will be stored in the order they were encountered.

Some EDI records may contain recurring fields in the same record, each being identified by a qualifier followed by the value. Any one specific qualifier may appear in any of the possible recurring field positions. For example, the following LIN record all need to populate the same target fields:

```
LIN*099977543214*EN*2099977543217*VN*ABC001*CH*USA*UP*099977543214*HD*AA10211002^
LIN*099977543214*VN*ABC001*UP*099977543214*EN*2099977543217*HD*AA10211002*CH*USA^
LIN*099977543214*HD*AA10211002*UP*099977543214*EN*2099977543217*VN*ABC001*CH*USA^
```

With using fixed mapping, each field to be extracted above would need to be defined five times, one for each possible position. Instead of a single EDI_Field can be defined for a given field with the flag **Is_Recurring** set to Yes along with the following Recurring attributes set to provide the specifics and the EDI Preprocessing will pick up the field in any of the possible positions. When **Is_Recurring** is set to Yes, the following EDI_Field attributes are applied differently:

- **Qualifier_Position** – relative position of the recurring qualifier. Normally, **Qualifier_Position** is an absolute and fixed value. For recurring fields, it is relative to the base position as describe below. The values for the qualifier position for recurring fields should be consecutive and start with 1.
- **Data_Position** - relative position of the recurring value. Normally, **Data_Position** is an absolute and fixed value. For recurring fields, it is relative to the base position as described below.

The additional fields are:

- **Recurring_Interval** – specifies how many positions each recurrent field requires (this is typically 2: one for the data qualifier and one for the value). This value should be one more than the total number of **Qualifier_Position** and **Qualifier_Value** pairs defined for the field.

- **Recurring_Base_Position** – base position of the first recurring qualifier. This will be the position of the first qualifier for the first recurring field. The position of each recurring qualifier can be calculated with the formula:

  actual_qualifier_position = (**Recurring_Interval** * occurrence-1) + (**Recurring_Base_Position** -1) + **Qualifier_Position**.

  For example, in the three example LIN records shown above, to extract the EN field, the **Recurring_Base_Position** would be set to 2, the **Recurring_Interval** set to 2, **Qualifier_Position** set to 1, the **Qualifier_Value** set to EN, the **Data_Position** set to 2, and the **Recurring_Max_Values** set to 5.

- **Recurring_Max_Values** – maximum number of occurrences of a recurring field in a record.

The **Concatenate_Consecutive_Elements** flag indicates whether the data value from consecutive qualified Data Elements should be concatenated. If set to Yes, they will be concatenated. If the **Maximum_Consecutive_Elements** is set, then only the specified number of elements will be concatenated. Any additional consecutive elements will be ignored.

Each EDI File has a designated repeating field delimiter. If this delimiter is encountered in a value, it will be converted to the delimiter defined by **Repeating_Field_Delimiter**. If it is undefined, then no conversion will take place.

Generating Multiple Files

If the EDI Specification identifies multiple files, then a separate specification is defined for each file and the processing for each file is completely separate from the others, but the source EDI file is processed in a single pass.

For example, if an EDI File is to generate a Product file and a SKU file, two EDI Target Files will be defined, and each will have a complete set of EDI Sections and EDI Fields. For the Product file, the EDI Section set to be the Is Output Row would be for the LIN data element (Example 5 above). Since the SKU data would not be stored, there would be no need to define any fields nor sections containing SKU data. For the SKU file, the first EDI section would be the LIN Data Element and its child section would be the SLN Data Element, which would be set to Is Output Row (Example 2 above). If consecutive LIN Data Elements are encountered (with no SLN Data Elements), then the Product data will not be saved to the SKU file. Any time an SLN Data Element is encountered, then its data would be written the next time the next SLN, or LIN Data Element is encountered, or once the end of the file is reached.

Importing the EDI Specification from a Spreadsheet

The EDI Mapping Specification spreadsheet can be used to facilitate the creation and management of the EDI mappings. This file format includes the data for all three repositories: EDI_Target_File, EDI_Section, and EDI_Field.

EDI_Specification_Name	Target_File_ID	Target_File_Name	Section_Data_Element_Name	Section_Qualifier_Position	Section_Qualifier_Value	Is_Output_Row	Parent_Section	Section_ID	Field_Name	Field_Position	Field_Data_Element_Name	Field_Qualifier_Position	Field_Qualifier_Value	Data_Position	Literal_Value	Is_Inherited	Is_Numbered	Numbered_Group	Numbered_Sequence	Max_Numbered	Is_Recurring	Recurring_Interval	Recurring_Base_Position	Recurring_Max_Values	Should_Trim	Max_Length	Repeating_Field_Delimiter	Concatenate_Consecutive_Elements	Maximum_Consecutive_Elements	Field_ID
B1	1003	EDI_To_B1.csv	BCT	1	PC	0		1000						0		Yes									Yes			No		
B1	1003	EDI_To_B1.csv	LIN			1		1001	Record ID	1	LIN			0	[blank]	No									Yes			No		1001
B1	1003	EDI_To_B1.csv	LIN			0		1001	Usage Indicator	2	LIN			0	[blank]	No									Yes			No		1002
B1	1003	EDI_To_B1.csv	BCT	1	PC	0		1001	Seller ID Qualifier	3	N1	1	SE	3		Yes									Yes			No		1003
B1	1003	EDI_To_B1.csv	BCT	1	PC	0		1001	Seller ID	4	N1	1	SE	4		Yes									Yes			No		1004
B1	1003	EDI_To_B1.csv	LIN			0		1001	Time Stamp	5	LIN			0	[blank]	No									Yes			No		1005
B1	1003	EDI_To_B1.csv	LIN			0		1001	Update Status	6	LIN			0	[blank]	No									Yes			No		1006
B1	1003	EDI_To_B1.csv	LIN			0		1001	IDW Item Control Number	7	LIN			1		No									Yes			No		1007
B1	1003	EDI_To_B1.csv	LIN			0		1001	UPC	8	LIN	1	UP	2		No					Yes	2	2	10	Yes			No		1008
B1	1003	EDI_To_B1.csv	LIN			0		1001	GTIN	9	LIN	1	UK	2		No					Yes	2	2	10	Yes			No		1009
B1	1003	EDI_To_B1.csv	LIN			0		1001	EAN	10	LIN	1	EN	2		No					Yes	2	2	10	Yes			No		1010
B1	1003	EDI_To_B1.csv	LIN			0		1001	Catalog Number	11	LIN	1	VN	2		No					Yes	2	2	10	Yes			No		1011
B1	1003	EDI_To_B1.csv	LIN			0		1001	IDW Index ID	12	LIN	1	A3	2		No					Yes	2	2	10	Yes			No		1012

The data from this file can be imported with a multi-repository import template or by importing separately into each repository. If the latter is performed, the ID columns must be pre-populated in the file.

Preprocessing Steps

Processing sequence for processing an EDI Mapping Specification file

1. Load the EDI Specification into Memory
2. Define output files (including headers) and set up staging buckets for each EDI Target File
3. For each data element in the file:
    a. Read the data element and parse its data from the file
    b. For each output file:
        i. Check if the Data Element matches an EDI Section in any of the current EDI Section's parent EDI Sections, a sibling EDI Section to the current EDI Section, or a direct child EDI Section to the current EDI Section:
            1. If new EDI Section has a direct descendant to the EDI Section with Is Output Row and some data has been previously stored for the EDI Section with Is Output Row:
                a. Output contents of staging bucket in file.
            2. Clear the fields mapped from the new EDI Section all descendant EDI Sections
            3. Update the current EDI Section to the new EDI Section
        ii. For each defined field in the current EDI Section or Is Inherited field from a parent EDI Section:
            1. If data element from the file matches qualified data element for the EDI Field (including current EDI Section)
                a. Extract the data by Data Position and place in staging bucket
        iii. Next field.
    c. Next file.
4. Next data element
5. For each output file:
    a. If any staging bucket EDI Field in the Is Output Row EDI Section has been defined output the contents of the staging bucket to the file.
    b. Close the output file
6. Next output file