

WebSphere Business Integration Message Broker



Message Models: Reference

Version 5 Release 0

WebSphere Business Integration Message Broker



Message Models: Reference

Version 5 Release 0

Note

Before using this information and the product it supports, read the information in the Notices appendix.

Third Edition (August 2004)

This edition applies to IBM® WebSphere® Business Integration Message Broker Version 5.0 and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2000, 2004. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this topic collection. v

Message model reference information. . . 1

Message set preferences	1
Editors	1
Validation	2
XML Schema Importer	2
Message set properties	3
Custom Wire Format message set properties.	4
XML Wire Format message set properties.	7
TDS Format message set properties	12
Documentation properties for all message set objects	20
Message definition file properties	20
Message definition file includes properties	20
Message definition file imports properties	21
Documentation properties for all message set objects	21
Message category properties.	21
Message category member properties.	21
Message model object properties	22
Logical properties for message model objects	22
Physical properties for message model objects.	46
Documentation properties for all message set objects	77
Message model object properties by object	77
Deprecated message model object properties	391
Logical properties for deprecated message model objects	391

Physical properties for deprecated message model objects	394
Documentation properties for all message set objects.	397
Deprecated message model object properties by object	397
Additional MRM domain information	501
Additional logical information.	502
Additional CWF information	503
Additional XML information	504
Additional TDS information	506
DateTime formats	527
Generated model representations.	533
Document generation.	533
WSDL generation	533
XML Schema generation.	538
Import formats	541
Importing from C: supported features	541
Importing from COBOL: supported features	544
Importing from XML Schema: unsupported features	548
Importing from XML DTD: unsupported features	551

Appendix. Notices 553

Trademarks	555
----------------------	-----

About this topic collection

Always refer to the WebSphere Business Integration Message Broker online information center to access the most current information. The information center is periodically updated on the document update site.

The PDFs that you can download from that Web site might not contain the most current information.

The topic content included in the PDF does not include the "Related Links" sections provided in the online topics. Links within the topic content itself are included, but are active only if they link to another topic in the same PDF collection (when the link includes a page number). Links to topics outside this topic collection attempt to link to a PDF that is called after the topic identifier (for example, ac12340_.pdf) and therefore fail. You can identify invalid links like this because they have no associated page number. Use the online information to navigate freely between topics.

Do not provide feedback on this PDF. Refer to the online information to ensure that you have access to the most current information, and use the Feedback link that appears at the end of each topic to report any errors or suggestions for improvement. Using the Feedback link provides precise information about the location of your comment.

The content of these topics is created for viewing online; you might find that the formatting and presentation of some figures, tables, examples, and so on are not optimized for the printed page. Text highlighting might also have a different appearance.

Message model reference information

Message model reference information is available for:

- “Message set preferences”
- “Message set properties” on page 3
- “Message definition file properties” on page 20
- “Message category properties” on page 21
- “Message model object properties” on page 22
- “Deprecated message model object properties” on page 391
- “Additional MRM domain information” on page 501
- “Generated model representations” on page 533
- “Import formats” on page 541

Message set preferences

You have the ability to alter a number of the preferences that affect the way certain areas of message set processing is handled. The areas are;

- “Editors”
- “Validation” on page 2
- “XML Schema Importer” on page 2

Editors

The following options are available to set as preferences for the message set editors;

Property	Type	Meaning
Auto Select Marker Object in Tree Viewer	Check box	Selects the given object in the Outline View after a tasklist entry is selected.
Show error images in outline view	Check Box	This will show error images in the outline view. For example, if an element has an error it will show the error image.

Message Definition tab order

Property	Type	Meaning
Overview then Properties	Button	Select this option if you want to have the Overview tab shown before the Properties tab. The Overview will be the view that is displayed on opening a message definition file in the message definition editor.
Properties then Overview	Button	Select this option if you want to have the Properties tab shown before the Overview tab. The Properties will be the view that is displayed on opening a message definition file in the message definition editor.

Validation

You can customize a number of the warning messages that the message set validation will generate. This is done through the Message Set Validation Preference page.

Any warning or error that falls into any of the following categories below can be customized on a category basis. The customization consists of both severity and priority. The severity can be any of the following:

- Error
- Warning
- Info
- Ignore

The priority can be any of the following as long as severity is not Ignore:

- High
- Normal
- Low

If severity is Ignore the user will not be able to change the priority.

Message set validation settings

The following is a list of the categories that can be customized;

- Use of deprecated constructs
- Messages with abstract global elements
- Tagged/Delimited String group content
- Facet runtime validation differences
- Type/Element substitution runtime validation differences
- Mixed content runtime validation differences
- Wildcard runtime validation differences
- Zero Custom Wire Format length count
- Zero Tagged/Delimited String Format length count
- Empty Tagged/Delimited String Format tag

XML Schema Importer

You can customize the following categories that affect the way in which an XML schema is imported into a message set that does not support namespaces.

Category	Modify	Reject
Import	Converts Import to Include	Import fails if it sees an Import
Redefine	Removes the Redefine statements	Import fails if it sees a Redefine
List	Changes type base to xsd:string	Import fails if it sees a List
Union	Changes type base to xsd:string	Import fails if it sees a Union
Abstract Complex Type	Sets abstract to false	Import fails if it sees an Abstract Complex Type
Abstract Element	Sets abstract to false	Import fails if it sees an Abstract Element

Message set properties

General message set properties

The table below defines the properties that you can set to customize the message set.

Property	Type	Meaning
Message set ID	String	This is a unique identifier that is automatically generated for you when you create the message set. You cannot change this property.
Default Wire Format	String	Specify the default wire format that is used if a format cannot be deduced from the message's MQRFH2 header, or was not specified as a property of the input node at which the message is received by a message flow. The default is empty (not set).
Message Type Prefix	String	<p>This property is used when you define multipart messages (see Multipart messages for a definition and explanation of multipart messages).</p> <p>The value that you specify is used as an absolute or relative path to the innermost message from the outermost, and is used as a prefix to the value of the Message Type property specified for the outermost message (specified either in the MQRFH2 header of the message, or in the input node of the message flow).</p> <p>If you set a value, it must be in the form <code>id1/id2/.../idn</code> where <code>id1</code> is the identifier of the outermost message, <code>id2</code> is the identifier of the next element or message, and <code>idn</code> is the identifier of the innermost message. The default value is blank (not set).</p> <p>The table below, describing the use of the message set property <i>Message Type Prefix</i>, shows how this value is combined with the <i>Message Type</i> property of an input message.</p>
Runtime Parser	Enumerated Type	<p>Select the message parser for messages belonging to this set from the drop-down list. This determines how the message is interpreted when it is received by the broker. You can choose from:</p> <p>MRM (the default). If you specify MRM, you must assign the message set to the brokers that will receive these messages, and deploy. The deploy action creates a runtime dictionary (RTD) against which the MRM parser invoked by the broker checks the received message.</p> <p>XML, JMSMap, or JMSStream. Choose one of these parsers if you want to model a generic self-defining XML, JMSMap, or JMSStream message. Messages defined in this way are interpreted by the generic XML parser, not the MRM parser. (You do not have to assign these message sets to brokers, nor deploy them to create RTDs.)</p>
Use namespaces	Check box	<p>Select this property if you want to use namespaces within the message set.</p> <p>Once this has been enabled, it cannot be disabled.</p>
Treat Length facet as MaxLength	Check box	<p>Select this property if you want the COBOL importer to create a <code>maxLength</code> facet for a fixed length string element rather than a length facet.</p> <p>The default is for this property to be set.</p>

Use of the message set property Message Type Prefix

The table below shows the implications of using the property *Message Type Prefix*. Note that message type or message prefix may describe might be elements or messages.

Message Type property example	Message Type Prefix not set	Message Type Prefix set
Simple Message Type:msg_type	Results in the simple Message Type:msg_type	Results in the path Message Type: /msg_prefix_1/.../msg_prefix_n/msg_type
Path Message Type:msg_type_1/.../msg_type_m	Results in the path Message Type:/msg_type_1/.../msg_type_m	Results in the combined path Message Type: /msg_prefix_1.../msg_prefix_n /msg_type_1/.../msg_type_m
Simple absolute Message Type:/msg_type	Results in the simple Message Type:msg_type	Results in the simple Message Type:msg_type An error is raised if Message Type Prefix is set to any value other than msg_type.
Path absolute Message Type:/msg_type_1/.../msg_type_m	Results in the path Message Type:/msg_type_1/.../msg_type_m	Results in the path Message Type:/msg_type_1/.../msg_type_m An error is raised if all identifiers in Message Type Prefix do not match the corresponding identifiers in the resulting path.

The message set does not have any properties that are dictated by membership of a larger object, because this is the largest message object as defined by the MRM.

In addition to message set properties, each of the physical formats have properties that can be updated. There is also a documentation property for a message set. Details of these can be found at;

- “Custom Wire Format message set properties”
- “XML Wire Format message set properties” on page 7
- “TDS Format message set properties” on page 12
- “Documentation properties for all message set objects” on page 20

Custom Wire Format message set properties

The table below defines the properties that you can set for the message set. Some of the message set properties (marked with an asterisk (*)) are relevant only if the message being processed is *not* using WebSphere MQ as the transport protocol. If the transport protocol is WebSphere MQ, values are derived from the message headers (for example, MQMD), and the message set properties, if set, are ignored.

Binary representation of boolean values

Property	Type	Meaning
Boolean True Value	STRING	Enter up to eight hexadecimal digits. Do not include the hexadecimal indicator (0x) preceding this number. Each digit is a half byte. The maximum length is 4 bytes. You must enter an even number of digits (a whole number of bytes). This value must be different from, but the same length as, the <i>Boolean False Value</i> . The default value is 00000001.
Boolean False Value	STRING	Enter up to eight hexadecimal digits. Do not include the hexadecimal indicator (0x) preceding this number. Each digit is a half byte. The maximum length is 4 bytes. You must enter an even number of digits (a whole number of bytes). This value must be different from, but the same length as, the <i>Boolean True Value</i> . The default value is 00000000.

Property	Type	Meaning
Boolean Null Value	STRING	Enter up to eight hexadecimal digits. Do not include the hexadecimal indicator (0x) preceding this number. Each digit is a half byte. The maximum length is 4 bytes. You must enter an even number of digits (a whole number of bytes). This value can be the same as either <i>Boolean True Value</i> or <i>Boolean False Value</i> , or different. The default value is 00000000.

Binary representation of decimal values

Property	Type	Meaning
Packed Decimal Positive Code	Enumerated Type	Select the positive sign used for packed decimal numbers from the drop-down list. The default is C, which indicates that 0x0C is used as the positive sign, which is the usual value. You can also select F, which indicates that 0x0F is used as the positive sign: some systems require this setting.

Output settings

These settings are used when messages are being serialized.

Property	Type	Meaning
Byte Alignment Pad	String	<p>If the xsd:element Custom Wire Format properties Byte Alignment, Leading Skip Count, and Trailing Skip Count cause bytes to be skipped in the bit stream when the message is serialized, this property supplies the character to be used in the skipped positions. Set this character in one of the following ways:</p> <ul style="list-style-type: none"> Select SPACE (the default), NUL, or 0 from the drop-down list. Enter a character between quotation marks, for example "c" or 'c', where c is any alphanumeric character. Enter a decimal character code in the form YY where YY is a decimal value. Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.
Policy for Missing Elements	Enumerated Type	<p>The setting of this property governs the action taken by the broker when fields are missing from the message tree when the message is serialized:</p> <ul style="list-style-type: none"> Use <i>Default Value</i> (the default). If a <i>Default Value</i> exists for the element, output it; otherwise throw an exception. Use <i>Null Value</i>. If the <i>Nilable</i> property of the element is checked, and an <i>Encoding Null Value</i> is specified for the element, output the <i>Encoding Null Value</i> according to the rules defined by the <i>Encoding Null</i> property. Otherwise an exception is thrown.

DateTime settings

Property	Type	Meaning
Derive default dateTime format from logical type	Button	Select this option if the dateTime format that will be used is to be taken from the dateTime format specified in the properties of an object that has one of the dateTime types. For example, a type gDay.

Property	Type	Meaning
Use default dateTime Format	Button and dateTime	<p>Select this option if you want to specify a dateTime format that is different from the logical dateTime format.</p> <p>Specify the default format for objects of type dateTime for this physical format. You can override this property for a dateTime object within a complex type.</p> <p>The initial value for this property is yyyy-MM-dd'T'HH:mm:ssZZZ, which you can change by over-typing.</p> <p>For more information about dateTime formats, see “DateTime formats” on page 527.</p>
Century Window	Integer	<p>This property determines how two-digit years are interpreted. Specify the two digits that start a 100-year window that contains the current year. For example, if you specify 89, and the current year is 2002, all two-digit dates are interpreted as being in the range 1989 to 2088.</p> <p>The initial value is 53, which you can change by over-typing.</p>
Days in First Week of Year	Enumerated Type	<p>Specify the number of days of the new year that must fall within the first week.</p> <p>The start of a year usually falls in the middle of a week. If the number of days in that week is less than the value specified here, the week is considered to be the last week of the previous year; hence week 1 starts some days into the new year. Otherwise it is considered to be the first week of the new year; hence week 1 starts some days before the new year.</p> <p>Select Use Broker Locale, which causes the broker to get the information from the underlying platform, or select a number from the drop-down list. The initial value is 4.</p>
First Day of Week	Enumerated Type	<p>Specify the day on which each new week starts.</p> <p>Select Use Broker Locale, which causes the broker to get the information from the underlying platform, or select a value from the drop-down list. The initial value is Monday.</p>
Strict DateTime Checking	Check box	<p>Select this option if you want to restrict dateTimes to a valid dateTime format. This will not allow 35th March to be processed as 4th April, and 10:79 to be processed as 11:19. If <i>Strict DateTime Checking</i> is set, receiving an incorrect dateTime will cause an error. The default is to not to restrict dateTimes.</p>
Time Zone	Enumerated Type	<p>The value that you set for this property is used if the value that you specified for the <i>Default DateTime Format</i> property does not include Time Zone information.</p> <p>The initial value is Use Broker Locale which causes the broker to get the information from the underlying platform.</p> <p>You can change this using the drop down box.</p>
Daylight Savings Time	Check box	<p>Select this option if the area in the <i>Time Zone</i> property observes daylight savings time. If it does not observe daylight savings time, this option should not be selected.</p> <p>For example, if an area is selected in <i>Time Zone</i> and this option is not selected, the value passed will represent the time zone without the daylight savings time.</p> <p>Default is not to observe daylight savings time.</p>

Character and numeric encoding for non-WebSphere MQ messages

These settings are used only for messages with no MQMD.

Property	Type	Meaning
Default CCSID*	INTEGER	Enter a numerical value for the default Coded Character Set Identifier. The default is 500. If the input message is a WebSphere MQ message, the equivalent attribute set for the queue manager is used and this property is ignored.
Byte Order*	Enumerated Type	Select either Big Endian (the default) or Little Endian from the drop-down list to specify the byte order of numbers that are represented as binary integers. In C, this is equivalent to data type short or long. In COBOL, this is equivalent to a PIC 9 COMP, COMP-4, COMP-5 or BINARY data type. Your choice must match the encoding with which messages are created. Big Endian is normally the correct option for messages created on UNIX or z/OS, Little Endian for Windows. This property is not used if the message is received across the WebSphere MQ transport protocol: in this case, the property is deduced from the MQMD of the message, or from the broker queue manager's encoding.
Packed Decimal Byte Order*	Enumerated Type	Select Big Endian (the default) or Little Endian from the drop-down list to specify the byte order of numbers that are represented as packed decimal. In COBOL, this is equivalent to PIC 9 COMP-3 data type. (There is no equivalent data type in C.) Your choice must match the encoding with which messages are created. Big Endian is normally the correct option for messages created on UNIX or z/OS, Little Endian for Windows NT.
Float Format*	Enumerated Type	Select one of S390 (the default), IEEE, or Reverse IEEE from the drop-down list to specify the byte order of numbers in the message that are represented as floating point.

XML Wire Format message set properties

The tables below define the properties for the XML Wire Format for the message set.

Namespace settings

Property	Type	Meaning
Namespace URI	String	Enter the namespace name that will identify which namespace you are using for the associated prefix.
Prefix	String	Enter the prefix to associate the element and attribute names you use it with to the namespace name.

Namespace schema locations

Property	Type	Meaning
Namespace URI	String	Enter the namespace name that will identify which namespace you are using.
Schema location	String	Enter the location of the schema for the associated namespace name that will be used to validate objects within the namespace.

XML declarations

Property	Type	Meaning
Suppress XML Declaration	Check box	Select the check box to suppress the XML declaration. If selected, the declaration (for example, <code><?xml version='1.0'></code>) is suppressed.
Standalone Document	Enumerated Type	<p>Select Yes, No, or Null from the drop-down list. If Null is selected, no standalone declaration is present in the XML declaration. If you select Yes or No, the declaration <code>standalone = "yes"</code> or <code>standalone = "no"</code> is added to the XML declaration when the output message is written.</p> <p>The setting of this property does not determine if an external DTD subset is loaded: external DTD subsets are never loaded in this release.</p> <p>If you set <i>Suppress XML Declaration</i> to Yes, this property is ignored.</p>
Output Namespace Declaration	Enumerated Type	<p>The <i>Output Namespace Declaration</i> property controls where the namespace declarations will placed in the output XML document. Select from:</p> <ul style="list-style-type: none"> At start of document. Declarations for all of the entries in the <i>Namespace schema locations</i> table above will be output as attributes of the message in the output XML document. The disadvantage of this option is that in some cases unnecessary declarations may be output. As required. Declarations will only be output when required by an element or attribute that is in that namespace. The disadvantage of this option is that the same namespace declaration may need to be output more than once in the output XML document. <p>The default option is At start of document.</p> <p>This property is only active if namespaces are enabled for this message set.</p>

XML document type settings

Property	Type	Meaning
Suppress DOCTYPE	Check box	If you select the check box, the DOCTYPE (DTD) declaration is suppressed.
DOCTYPE System ID	String	<p>Specify the System ID for DOCTYPE external DTD subset (if DOCTYPE is present). This is normally set to the name of the generated (or imported) DTD for a message set.</p> <p>If <i>Suppress DOCTYPE</i> is set, this property is ignored and cannot be changed (the field is disabled). The default value is <code>www.mrmnames.net/</code> followed by the message set identifier.</p>
DOCTYPE Public ID	String	<p>Specify the Public ID for DOCTYPE external DTD subset (if DOCTYPE is present, and System ID is specified).</p> <p>If <i>Suppress DOCTYPE</i> is set, this property is ignored and cannot be changed (the field is disabled). The default value is the message set identifier.</p>

Property	Type	Meaning
DOCTYPE Text	String	<p>Use this property to add additional DTD declarations. It is not parsed by the XML parser and thus it might not be valid XML. You can include ENTITY definitions or internal DTD declarations. It is a string (up to 32KB) in which new line and tab characters are replaced by \n and \t respectively.</p> <p>The content is not parsed, and appears in the output message. If there is an in-line DTD, the content of this property takes precedence.</p> <p>If you have set <i>Suppress DOCTYPE</i>, this property is ignored and cannot be changed (the field is disabled).</p> <p>For more information, see “In-line DTDs and the DOCTYPE text property” on page 12.</p> <p>The default value is empty (not set).</p>

Root Tag Name

Property	Type	Meaning
Root Tag Name	String	<p>Specify the name of the message set root tag. You can leave this property blank, in which case no wrapper tags are used for messages (that is, the message tag is the root of the document). The name can be followed by a space and additional text for attribute/value pairs to appear with the root tag.</p> <p>The default value is MRM.</p>

Suppress Timestamp Comment

Property	Type	Meaning
Suppress Timestamp Comment	Check box	<p>If selected, the timestamp comment string in the XML output is suppressed.</p> <p>If not selected, the comment is not suppressed, and a comment of the form <!--MRM Generated XML Output on: Tue Apr 23 09:34:42 2002--> is included in the output message.</p> <p>The default is for the check box to be selected.</p>

Enable Versioning Support

Property	Type	Meaning
Enable Versioning Support	Check box	<p>If this is selected, versioning support is enabled. This property specifies whether XML namespace definitions are coded for the root tag in the message, together with namespace qualifiers for any elements that do not belong to the default namespace. These namespace definitions are used to represent the message set dependency information, which is used to support the exchange of messages between applications that are based on different customizations of the same message set.</p> <p>The default is for the check box to be selected, for compatibility with MRM XML messages in earlier releases. If you did not use MRM XML messages in earlier releases, you are strongly recommended to ensure this check box is not selected.</p>

XML representation of boolean values

Property	Type	Meaning
Boolean True Value	String	Specify the string that is used to encode and recognize BOOLEAN true values. When an XML document is parsed, the string 1 is always accepted as true for a BOOLEAN element. Enter a string of up to 254 characters. The default is true. 1 is also valid.
Boolean False Value	String	Specify the string that is used to encode and recognize BOOLEAN false values. When an XML document is parsed, the string 0 is always accepted as false for a BOOLEAN element. Enter a string of up to 254 characters. The default is false. 0 is also valid.

XML representation of null values

Property	Type	Meaning
Encoding Numeric Null	Enumerated Type	Specify the null encoding for numeric fields. This provides a method of affirming by comparison that the element is null. You must select one of the following values from the drop-down list: <ul style="list-style-type: none"> • NULLEmpty. The empty string is used as the comparison. This is the default value. • NULLAttribute. The property <i>Encoding Null Num Val</i> of the parent element is used. • NULLValue. The value of property <i>Encoding Null Num Val</i> is used. • NULLElement. The property <i>Encoding Null Num Val</i> of the child element is used. • NULLValAttr. This option is valid only for elements instantiated in a complex type with property <i>Member Render</i> set to <i>XMLElementAttrVal</i> or <i>XMLElementAttrIDVal</i>. See “XML Null handling options” on page 504 for full details.
Encoding Numeric Null Value	String	Specify the value to qualify the <i>Encoding Null Num</i> property, if you have set that to NULLAttribute, NULLValue, or NULLElement. Refer to “XML Null handling options” on page 504 for further information.
Encoding Non-Numeric Null	Enumerated Type	Specify the null encoding for non numeric fields. This is a method of affirming that the element is null. The options are identical to those available for property <i>Encoding Null Num</i> .
Encoding Non-Numeric Null Value	String	Specify the value to qualify the <i>Encoding Null Non-Num</i> property. Refer to “XML Null handling options” on page 504 for further information.

DateTime settings

Property	Type	Meaning
Derive default dateTime format from logical type	Button	Select this option if the dateTime format that will be used is to be taken from the dateTime format specified in the properties of an object that has one of the dateTime types. For example, a type gDay.

Property	Type	Meaning
Use default dateTime Format	Button and dateTime	<p>Select this option if you want to specify a dateTime format that is different from the logical dateTime format.</p> <p>Specify the default format for objects of type dateTime for this physical format. You can override this property for a dateTime object within a complex type.</p> <p>The initial value for this property is yyyy-MM-dd'T'HH:mm:ssZZZ, which you can change by over-typing.</p> <p>For more information about dateTime formats, see “DateTime formats” on page 527.</p>
Century Window	Integer	<p>This property determines how two-digit years are interpreted. Specify the two digits that start a 100-year window that contains the current year. For example, if you specify 89, and the current year is 2002, all two-digit dates are interpreted as being in the range 1989 to 2088.</p> <p>The initial value is 53, which you can change by over-typing.</p>
Days in First Week of Year	Enumerated Type	<p>Specify the number of days of the new year that must fall within the first week.</p> <p>The start of a year usually falls in the middle of a week. If the number of days in that week is less than the value specified here, the week is considered to be the last week of the previous year; hence week 1 starts some days into the new year. Otherwise it is considered to be the first week of the new year; hence week 1 starts some days before the new year.</p> <p>Select Use Broker Locale, which causes the broker to get the information from the underlying platform, or select a number from the drop-down list. The initial value is 4.</p>
First Day of Week	Enumerated Type	<p>Specify the day on which each new week starts.</p> <p>Select Use Broker Locale, which causes the broker to get the information from the underlying platform, or select a value from the drop-down list. The initial value is Monday.</p>
Strict DateTime Checking	Check box	<p>Select this option if you want to restrict dateTimeS to a valid dateTime format. This will not allow 35th March to be processed as 4th April, and 10:79 to be processed as 11:19. If <i>Strict DateTime Checking</i> is set, receiving an incorrect dateTime will cause an error. The default is to not to restrict dateTimeS.</p>
Time Zone	Enumerated Type	<p>The value that you set for this property is used if the value that you specified for the <i>Default DateTime Format</i> property does not include Time Zone information.</p> <p>The initial value is Use Broker Locale which causes the broker to get the information from the underlying platform.</p> <p>You can change this using the drop down box.</p>
Daylight Savings Time	Check box	<p>Select this option if the area in the <i>Time Zone</i> property observes daylight savings time. If it does not observe daylight savings time, this option should not be selected.</p> <p>For example, if an area is selected in <i>Time Zone</i> and this option is not selected, the value passed will represent the time zone without the daylight savings time.</p> <p>Default is not to observe daylight savings time.</p>

In-line DTDs and the DOCTYPE text property

You can include in-line DTDs in your messages, and you can specify additional information by setting the property *DOCTYPE Text*, but you must be aware of the action taken by the parser when it constructs an output message:

1. If you take any action that causes the output message to be regenerated, for example if you configure a Compute node to create a new output message by coding ESQL statements like `SET OutputRoot.MRM.Field1 = xxx`:
 - If you have set the property *Suppress DOCTYPE* for the message set in which you have defined this message to Yes, both DOCTYPE information (specified in the *DOCTYPE Text* property for the message set or message) and in-line DTD are excluded from the output message.
 - If you have set the property *Suppress DOCTYPE* for the message set in which you have defined this message to No.
 - The in-line DTD is preserved if possible.
 - Otherwise, if the message is self-defining, the message set *DOCTYPE Text* property information is included in the output message.
 - Otherwise (the message is not self-defining), the message level *DOCTYPE Text* property information is included in the output message.
2. If you do not take any action that causes the output message to be regenerated, the parser generates an output message that is a direct copy of the input message. This occurs if you have configured a Compute node in the message flow to copy the message using `SET OutputRoot = InputRoot` (explicitly, or by checking the *Copy entire message* check box), and you do not modify the message in any way in this or any other node. In this case the in-line DTD is retained in the output message but any information that you specify in the *DOCTYPE Text* property for the message set or message is not included.

TDS Format message set properties

The tables below shows the message set properties that you can set for the TDS Format. “Default TDS Message set properties” on page 17 shows the defaults for each of the industry standards for each of these properties.

Messaging Standard

Property	Type	Meaning
Messaging Standard	Enumerated Type	<p>Specify the standard to be used for this wire format. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none">• ACORD AL3• EDIFACT• SWIFT• UNKNOWN• X12• TLOG <p>Select UNKNOWN if you are defining your own tagged/delimited messages, or are using a standard that is not included in the list above.</p> <p>The selected value controls the default values for a number of the other properties.</p> <p>The default is UNKNOWN.</p>

Data element separation

Property	Type	Meaning
Group Indicator	String	Specify the value of a special character or string that precedes the data belonging to a group or complex type within the bit stream. If you set the group or type property <i>Group Indicator</i> , it overrides this value.
Group Terminator	String	Specify the value of a special character or string that terminates data belonging to a group or a complex type within the bit stream. If you set the group or type property <i>Group Terminator</i> , it overrides this value.
Delimiter	String	Specify the value of a special character or string that specifies the delimiter used between data elements. This property applies only to the delimited <i>Data Element Separation</i> methods (Tagged Delimited, All Elements Delimited, and Variable Elements Delimited).
Suppress Absent Element Delimiters	Enumerated type	Use this property to select if you want delimiters to be suppressed for elements that are missing within a message. Select from: <ul style="list-style-type: none"> • End Of Type. Use this option to suppress the delimiter when an element is missing. For example, if the model has been defined to have up to 3 elements and only 2 are present, the last delimiter can be omitted from the message. • Never. Use this option to ensure that even if optional elements are not present, all delimiters will be written out. This option should be used when the delimiter used to delimit parent and child objects is the same. For example, if an optional child element is missing, message processing applications could not tell where the child elements in a message ended and the next parent element started if the delimiters are all the same.
Tag Data Separator	Button and String	Specify the value of a special character or string that separates the Tag from the data. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides <i>Length of Tag</i> . This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).
Length of Tag	Integer	Specify the length of a tag value. When the message is parsed, this allows tags to be extracted from the bit stream if the <i>Tag Data Separator</i> property is not set. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides this value. This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).

Character data settings

Property	Type	Meaning
Default CCSID	Integer	CCSID (Coded Character Set Identification) specifies the mapping between character codes and symbols. You must specify a code set that is supported by WebSphere Business Integration Message Broker. This property stores the default CCSID for the message bit stream, but this value can be overridden when the message is processed (for example by the CCSID in the header of a WebSphere MQ input message).

Property	Type	Meaning
Trim Fixed Len String	Enumerated Type	<p>Specify if fixed length strings must be trimmed. You can select from:</p> <ul style="list-style-type: none"> No Trim Leading White Spaces Trailing White Spaces Trim Both Trim Padding Chars <p>The trim of padding characters occurs from the left or right depending on the <i>Justification</i> property for the element.</p> <p>You might need to use this if you have data input that is mapped to a numeric simple type. For example, if the input data has leading spaces, you can set this property to Leading White Spaces to avoid data conversion problems processing these fields.</p>
Escape Character	String	Specify the escape character that is used to allow special reserved characters (such as delimiters) to be included as part of data. You must specify a single character only, or a mnemonic that represents a single character.
Reserved Characters	String	<p>Specify the special reserved characters that must be preceded by the escape character if they are to be included as part of the data. The Escape Character, delimiters, and group indicators are usually included in this list.</p> <p>If the set of reserved characters is to be updated dynamically (in the case of EDIFACT and X12 when delimiters and so on are specified in service strings), you must use the mnemonics supplied to specify characters in this list.</p> <p>If you have specified reserved characters, an Escape Character must also be specified.</p>

Numeric Settings

Property	Type	Meaning
Decimal Point	String	Specify the character that is used to separate the whole part of a number from its fraction.
Strict Numeric Checking	Boolean	<p>This is used in relation with the <i>Messaging Standard</i> property, <i>Virtual Decimal Point</i> property and the <i>Precision</i> property of a global element. This allows you to apply stricter rules for the checking of numbers. ¹</p> <p>The default is for this property not to be set.</p>

Property	Type	Meaning
Note: 1. The rules for <i>Strict Numeric Checking</i> are: <ul style="list-style-type: none"> If the <i>Precision</i> property of a global element is set to All Significant Digits then there should only be a decimal separator if there is a fractional part to the value. If the <i>Precision</i> property of a global element is set to Explicit Decimal Point then the decimal separator must always be present, even if the fractional part is missing. If the <i>Precision</i> property of a global element is set to Exponential Notation then the incoming value must be in exponential notation. Exponential notation is only allowed for floating numbers. If the <i>Precision</i> property of a global element is set to a specific value, then the specific number of digits after the decimal separator must be present. All values contain at least one digit in the integer part of the number. If a <i>Virtual Decimal Point</i> of a global element has been set, the number must not have a decimal point. Except for EDIFACT the decimal separator should only be the specified value and '.' should not be permitted. For EDIFACT both '.' and the specified separator are permitted. In this case the decimal separator should be specified as ',' and the code will also permit '.' to be used. Except for exponential functions, only digits 0-9, the decimal separator and positive and negative signs are permitted. For exponential functions the characters 'e' and 'E' are also permitted. Padding characters are permitted only if they are in a position to be stripped from the number. 		

Representation of boolean values

Property	Type	Meaning
Boolean True Value	String	Specify the value of the string representing the boolean true value. The default value is 1.
Boolean False Value	String	Specify the value of the string representing the boolean false value. The default value is 0.
Boolean Null Value	String	Specify the value of the string representing the boolean null value. The default value is 0.

DateTime settings

Property	Type	Meaning
Derive default dateTime format from logical type	Button	Select this option if the dateTime format that will be used is to be taken from the dateTime format specified in the properties of an object that has one of the dateTime types. For example, a type gDay.
Use default dateTime Format	Button and dateTime	<p>Select this option if you want to specify a dateTime format that is different from the logical dateTime format.</p> <p>Specify the default format for objects of type dateTime for this physical format. You can override this property for a dateTime object within a complex type.</p> <p>The initial value for this property is yyyy-MM-dd'T'HH:mm:ssZZZ, which you can change by over-typing.</p> <p>For more information about dateTime formats, see "DateTime formats" on page 527.</p>

Property	Type	Meaning
Century Window	Integer	<p>This property determines how two-digit years are interpreted. Specify the two digits that start a 100-year window that contains the current year. For example, if you specify 89, and the current year is 2002, all two-digit dates are interpreted as being in the range 1989 to 2088.</p> <p>The initial value is 53, which you can change by over-typing.</p>
Days in First Week of Year	Enumerated Type	<p>Specify the number of days of the new year that must fall within the first week.</p> <p>The start of a year usually falls in the middle of a week. If the number of days in that week is less than the value specified here, the week is considered to be the last week of the previous year; hence week 1 starts some days into the new year. Otherwise it is considered to be the first week of the new year; hence week 1 starts some days before the new year.</p> <p>Select Use Broker Locale, which causes the broker to get the information from the underlying platform, or select a number from the drop-down list. The initial value is 4.</p>
First Day of Week	Enumerated Type	<p>Specify the day on which each new week starts.</p> <p>Select Use Broker Locale, which causes the broker to get the information from the underlying platform, or select a value from the drop-down list. The initial value is Monday.</p>
Strict DateTime Checking	Check box	<p>Select this option if you want to restrict dateTimes to a valid dateTime format. This will not allow 35th March to be processed as 4th April, and 10:79 to be processed as 11:19. If <i>Strict DateTime Checking</i> is set, receiving an incorrect dateTime will cause an error. The default is to not to restrict dateTimes.</p>
Time Zone	Enumerated Type	<p>The value that you set for this property is used if the value that you specified for the <i>Default DateTime Format</i> property does not include Time Zone information.</p> <p>The initial value is Use Broker Locale which causes the broker to get the information from the underlying platform.</p> <p>You can change this using the drop down box.</p>
Daylight Savings Time	Check box	<p>Select this option if the area in the <i>Time Zone</i> property observes daylight savings time. If it does not observe daylight savings time, this option should not be selected.</p> <p>For example, if an area is selected in <i>Time Zone</i> and this option is not selected, the value passed will represent the time zone without the daylight savings time.</p> <p>Default is not to observe daylight savings time.</p>

TDS Mnemonics

The Tagged/Delimited String Format (TDS) uses mnemonics for a number of properties for a message set, complex type, or both. These TDS mnemonics and their associated properties are listed in the table below.

Mnemonic string	Meaning	Default value	Associated property
<EDIFACT_CS>	Component data separator in EDIFACT	:	Message set and type, <i>Delimiter</i>
<EDIFACT_DS>	Data element separator in EDIFACT	+	Message set and type, <i>Delimiter</i>

Mnemonic string	Meaning	Default value	Associated property
<EDIFACT_TAGDATA_SEP>	Tag data separator in EDIFACT This is overridden with the same value as that which overrides <EDIFACT_DS>	+	Message set and type, <i>Tag Data Separator</i>
<EDIFACT_DEC_NOTATION>	Decimal notation in EDIFACT	.	Message set, <i>Decimal Point</i>
<EDIFACT_ESC_CHAR>	Escape character in EDIFACT	?	Message set, <i>Escape Character</i>
<EDIFACT_GROUP_TERM>	Tag terminator in EDIFACT	'	Message set, <i>Tag Terminator</i>
<X12_GROUP_TERM>	Tag terminator in X12	!	Message set level, <i>Tag Terminator</i>
<X12_DS>	Data element separator for X12	*	Message set and type, <i>Delimiter</i>
<X12_CS>	Component data element separator for X12	:	Message set and type, <i>Delimiter</i>
<LT>	Represents the less than character, which is a reserved character	<	
<GT>	Represents the greater than character, which is a reserved character	>	
<CR>	Represents the carriage return character	%X0D	
<LF>	Represents the line feed character	%X0A	

Mnemonics are also supported for the following control characters:

<ACK> (x'06')	<BEL> (x'07')	<BS> (x'08')	<CAN> (x'18')
<CR> (x'0D')	<DC1> (x'11')	<DC2> (x'12')	<DC3> (x'13')
<DC4> (x'14')	<DLE> (x'10')	 (x'19')	<ENQ> (x'05')
<EOT> (x'04')	<ESC> (x'1B')	<ETB> (x'17')	<ETX> (x'03')
<FF> (x'0C')	<FS> (x'1C')	<GS> (x'1D')	<GT> (x'3E')
<HT> (x'09')	<LF> (x'0A')	<LT> (x'3C')	<NAK> (x'15')
<NUL> (x'00')	<RS> (x'1E')	<SI> (x'0F')	<SO> (x'0E')
<SOH> (x'01')	<SP> (x'20')	<STX> (x'02')	<SUB> (x'1A')
<SYN> (x'16')	<US> (x'1F')	<VT> (x'0B')	

You can enter a mnemonic in the form <U+xxxx> where xxxx are hexadecimal numbers up to a value of FFFF. These numbers represent a Unicode character, not a character in your local code page or the code page in which message data is formatted. None of the characters in this structure are case sensitive. Do not enclose spaces inside the angle brackets.

Default TDS Message set properties

The following tables define the defaults for the message set properties for the TDS Format for each of the industry standard messages that you can define. For more information about the TDS Format, see “TDS Format message set properties” on page 12 and “TDS Mnemonics” on page 16.

Default message set property values for TDS (part 1 of 2)

Property	Messaging standard = ACORD AL3	Messaging standard = EDIFACT
TDS Format Identifier	ACORD_AL3	EDIFACT
Default CCSID	367	367
Trim Fix Len String	No Trim	Trim Both
Group Indicator	Empty	Empty
Group Terminator	Empty	<EDIFACT_GROUP_TERM>
Tag Data Separator	Empty	<EDIFACT_TAGDATA_SEP>
Length of Tag	Empty	Empty
Delimiter	Empty	<EDIFACT_CS>
Decimal Point	.	<EDIFACT_DEC_NOTATION>
Escape Character	Empty	<EDIFACT_ESC_CHAR>
Reserved Chars	Empty	<EDIFACT_ESC_CHAR> <EDIFACT_TAGDATA_SEP> <EDIFACT_GROUP_TERM> <EDIFACT_CS>
Output Compression Technique	SimpleALCharCompression	n/a
Input Compression Technique	SimpleALCharCompression	n/a
Boolean True Representation	Y	1
Boolean False Representation	N	0
Boolean Null Representation	N	0
Default DateTime Format	blank ¹	blank ¹
Default Time Zone ID	blank ¹	blank ¹
Century Window	53	53
Days in First Week of Year	4	4
First Day of Week	Monday	Monday
Note:		
1. The value of blank causes the value to be obtained from the broker's locale.		

Default message set property values for TDS (part 2 of 2)

Property	Messaging standard = SWIFT	Messaging standard = UNKNOWN	Messaging standard = X12
TDS Wire Format Identifier	SWIFT	TDS	X12
Default CCSID	37	376	367
Trim Fix Len String	Trim Both	No Trim	Trim Both
Group Indicator	<CR><LF>:	Empty	Empty
Group Terminator	<CR><LF>-	Empty	<X12_GROUP_TERM>
Tag Data Separator	:	Empty	Empty
Length of Tag	Empty	Empty	Empty
Delimiter	<CR><LF>:	Empty	<X12_CS>
Decimal Point	,	.	.
Escape Character	Empty	Empty	None

Property	Messaging standard = SWIFT	Messaging standard = UNKNOWN	Messaging standard = X12
Reserved Chars	Empty	Empty	Empty
Output Compression Technique	n/a	None	n/a
Input Compression Technique	n/a	None	n/a
Boolean True Value	1	1	1
Boolean False Value	0	0	0
Boolean Null Value	0	0	0
Default DateTime Format	blank ¹	blank ¹	blank ¹
Default Time Zone ID	blank ¹	blank ¹	blank ¹
Century Window	80	53	53
Days in First Week of Year	4	4	4
First Day of Week	Monday	Monday	Monday

Note:

1. The value of blank causes the value to be obtained from the broker's locale.

Default type property values for TDS (part 1 of 2)

Property	Messaging standard = ACORD AL3	Messaging standard = EDIFACT
Group Indicator	Empty	Empty
Group Terminator	Empty	<EDIFACT_GROUP_TERM>
Tag Data Separator	Empty	<EDIFACT_DATA_SEP>
Length of Tag	Empty	Empty
Data Element Separation	Fixed Length AL3	All Elements Delimited
Delimiter	not applicable	<EDIFACT_CS>
Length of Encoded Length	not applicable	not applicable
Extra Chars in Encoded Length	not applicable	not applicable

Default type property values for TDS (part 2 of 2)

Property	Messaging standard = SWIFT	Messaging standard = UNKNOWN	Messaging standard = X12
Group Indicator	<CR><LF>:	Empty	Empty
Group Terminator	<CR><LF>-	Empty	<X12_GROUP_TERM>
Tag Data Separator	:	Empty	Empty
Length of Tag	Empty	Empty	Empty
Data Element Separation	Tagged Delimited	Fixed Length	All Elements Delimited
Delimiter	<CR><LF>:	Empty	<X12_CS>
Length of Encoded Length	not applicable	not applicable	not applicable

Property	Messaging standard = SWIFT	Messaging standard = UNKNOWN	Messaging standard = X12
Extra Chars in Encoded Length	not applicable	not applicable	not applicable

Documentation properties for all message set objects

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Message definition file properties

Namespace

Property	Type	Meaning
Prefix	String	The namespace prefix for the target namespace of this file. This field cannot be changed after the message definition file has been created.
Target Namespace	String	The target namespace for the message definition file. All global objects created within the file will have this namespace by default. This field cannot be changed after the message definition file has been created.

Default namespaces for local objects

Property	Type	Meaning
Elements	String	The default namespace for all local elements within this message definition file.
Attributes	String	The default namespace for all local attributes within this message definition file.

Property	Type	Meaning
Default block	String	The default value for the block attribute for all complex types and elements within this message definition file.
Default final		The default value for the final attribute for all complex types and elements within this message definition file.

Message definition file includes properties

Property	Type	Meaning
Schema Location	String	For each message definition file that has been included in this message definition file, this field displays its location. The location is displayed as a relative path from the message definition file to the included file.

Message definition file imports properties

Property	Type	Meaning
Schema Location	String	For each message definition file that has been imported into this message definition file, this field displays its location. The location is displayed as a relative path from the message definition file to the imported file.
Prefix	String	Displays the namespace prefix for each imported message definition file.
Namespace	String	Displays the namespace URI for each imported message definition file.

Documentation properties for all message set objects

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Message category properties

A message category does not have a name property as it takes its name from the name of the message category file.

Property	Type	Meaning
Message Category Kind	Enumerated type	This property describes the purpose of the message category. Choose from: <ul style="list-style-type: none">• other. This is the default.• wsdl. If the message category is to participate in the generation of WSDL documents, this property must be set to 'wsdl'. When the WSDL document is generated, the name of the message category provides the name for the <wsdl:operation> element generated for eligible messages in the message category.
Documentation	String	The documentation property of an object is where you can add information to enhance the understanding of that objects function. It is a string field and any standard alphanumeric characters can be used.

Message category member properties

Property	Type	Meaning
Role Type	Enumerated type	The role that the message plays in the message category. Select from: <ul style="list-style-type: none">• wsdl:input• wsdl:output• wsdl:fault If the message is to participate in the generation of WSDL documents then one of wsdl:input, wsdl:output, wsdl:fault must be selected. When the WSDL document is generated, a WSDL role element of the appropriate kind (<wsdl:input>, <wsdl:output>, <wsdl:fault>) will be created for the message, the name of which is provided by the Role Name property.

Property	Type	Meaning
Role Name	String	A descriptive name for the role this message plays in the message category. If this message participates in the generation of WSDL documents then the Role Name provides the name for the WSDL role element created for the message.
Documentation	String	The documentation property of an object is where you can add information to enhance the understanding of that objects function. It is a string field and any standard alphanumeric characters can be used.

Message model object properties

There are two ways of accessing the reference information for the properties of message model objects. The following topics allow you to access the property information by property kind:

- “Logical properties for message model objects”
- “Physical properties for message model objects” on page 46
- “Documentation properties for all message set objects” on page 20

Alternatively, you can access the property information by object, starting from:

- “Message model object properties by object” on page 77

Deprecated objects are dealt with separately. For further information, see “Deprecated message model object properties” on page 391

Logical properties for message model objects

Logical property information is available for the following objects:

- “Attribute group reference logical properties” on page 23
- “Attribute reference logical properties” on page 23
- “Complex type logical properties” on page 24
- “Element reference logical properties” on page 28
- “Global attribute logical properties” on page 29
- “Global attribute group logical properties” on page 31
- “Global element logical properties” on page 32
- “Global group logical properties” on page 34
- “Group reference logical properties” on page 36
- “Key logical properties” on page 38
- “Keyref logical properties” on page 38
- “Local element logical properties” on page 39
- “Local group logical properties” on page 41
- “Message logical properties” on page 43
- “Simple type logical properties” on page 44
- “Unique logical properties” on page 45
- “Wildcard attribute logical properties” on page 45
- “Wildcard element logical properties” on page 46

Attribute group reference logical properties

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Attribute reference logical properties

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Property	Type	Meaning
Usage	Enumerated type	<p>The usage property is used in conjunction with the <i>Value</i> property found in an attribute object. The default for the <i>Usage</i> property is optional.</p> <p>Select from;</p> <ul style="list-style-type: none"> • optional. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute can appear once and can have any value. – Where the <i>Value</i> property is set to default, the attribute can appear once. If it does not appear, its value will be the data that has been entered in the <i>Value</i> property. If it does appear it will be the value given. – Where the <i>Value</i> property is set to fixed, the attribute can appear once. If it does appear, its value must match the data that has been entered in the <i>Value</i> property. If it does not appear its value will be the data that has been entered in the <i>Value</i> property. • prohibited. The attribute must not appear. • required. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute must appear once and can have any value. – Where the <i>Value</i> property is set to fixed, the attribute must appear once and it must match the data that has been entered in the <i>Value</i> property.

Complex type logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Base Type	Enumerated type	You can use this property to select a type (simple or complex) that is used as the starting point to define a new complex type that is derived by restriction or extension.
Derived By	Enumerated type	<p>If this property is active, select from;</p> <ul style="list-style-type: none"> • restriction. If a complex type is derived by restriction, the content model of the complex type is a subset of the base type. • extension. If the complex type is derived by extension, the content model of the complex type is the content model of the base type plus the content model specified in the type derivation. <p>Derivation by list or union is not supported.</p>

Content

The table below shows the valid settings for *Composition* and *Content Validation*. These properties are actually located on the group which defines the content of this type. They can only be edited if the *Local group* button is selected. If the *Global group* button is selected, these properties are taken from the global group identified by the *Group name* field.

Valid children in a complex type that depend on both *Composition* and *Content Validation* are shown in “Content Validation properties for complex types” on page 27.

Property	Type	Meaning
Local Group	Button	You should select this if the content of your complex type is a local group.

Property	Type	Meaning
Composition	Enumerated type	<p>The property describes how the message tree is structured and is used in combination with the property <i>Content Validation</i>.</p> <p><i>Composition</i> determines, for example, if the elements within the tree can appear in any order, or if the order is predefined.</p> <p>If you set this property to Ordered Set or Sequence, the order of elements in the input message when the message is parsed, and the order in the logical tree when the output message is constructed by the parser, is important. If the order is not correct, the parser might generate an error, or might produce unexpected results. Therefore you must take care to include ESQL SET statements in the correct order when you create a message in a Compute node.</p> <p>Select from:</p> <ul style="list-style-type: none"> • Empty • sequence. If you select this option, you can only define children that are simple types, complex types, elements, or groups. These children, if present, must appear in the specified order. They can repeat and can be duplicated. • choice. If you select this option, you can define children that are simple types, complex types, or elements. Only one of the defined children of the complex type can be present, but repeating children are allowed. Use this option if you want to model C unions and COBOL REDEFINES in a Custom Wire Format, or an XML DTD element that uses choice in an XML Wire Format, or some industry standard tagged/delimited messages (for example SWIFT) use this format. • all. The elements in an all group can appear in any order. Each element can appear once, or not at all. An all group can only contain elements - groups are not allowed. An all group can only be used at the top level of a complex type - it cannot be a member of another group within a type. • unorderedSet. If you select this option, you can define only elements as children. The elements can repeat but cannot be duplicated. Child elements can appear in any order. • orderedSet. If you select this option, you can define only elements as children. These elements, if present, must appear in the specified order, and they can repeat but cannot be duplicated. This is the default value for new complex types. • message. If you select this option, you can define only messages as children. They can repeat, but they cannot be duplicated. Like Choice, only one of the defined children can be present. If the complex type includes more than one message, the bit stream contains the exact length of the embedded message, and is not padded to the length of the longest. Use this option to model multipart messages, which are used in some industry standards, for example, SWIFT. For more information, see the section on multipart messages in Multipart messages.

Property	Type	Meaning
Content Validation	Enumerated type	<p><i>Content Validation</i> controls how the broker responds to undeclared content and specifies where the objects that are included within the complex type are defined, if at all. It is used in combination with the <i>Composition</i> property.</p> <p>Options:</p> <ul style="list-style-type: none"> • Closed. The complex type can only contain the child elements that you have added to it. • Open Defined. The complex type can contain any valid element defined within the message set. • Open. The complex type can contain any valid element, not just those that you have added to this complex type. <p>See “Combinations of Composition and Content Validation” on page 123 for further details of these options.</p>
Group Reference	Button	You should select this if the content of your complex type is a reference to a group object
Group Name	Enumerated type	The <i>Group Name</i> is the name of the group that this complex type is referring to. The groups available to be referenced can be selected from the drop down list.
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>
Mixed	Check box	Select this where the complex type has mixed content and contains character data alongside sub-elements.

Substitution settings

Property	Type	Meaning
Final	Multiple selection enumerated type	<p>The final attribute on a complex type controls whether other types may be derived from it. Valid values are extension/restriction/all. You can select from one or more of the following:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit type substitution by elements whose types are restrictions of the head element’s type. • extension. Prohibit type substitution by elements whose types are extensions of the head element’s type. • all. Prohibit substitution by any method. <p>To select more than one, you will need to type the selection into the property field.</p>

Property	Type	Meaning
Block	Multiple selection enumerated type	<p>The block attribute on a complex type restricts the types of substitutions which are allowed for elements based on that type. In the WebSphere Business Integration Message Broker its effect is the same as if the block attribute were copied from the complex type onto every element based on the complex type. You can select from one or more of the following:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit type substitution by elements whose types are restrictions of the head element's type. • extension. Prohibit type substitution by elements whose types are extensions of the head element's type. • all. Prohibit substitution by any method. <p>To select more than one, you will need to type the selection into the property field.</p>
Abstract	Check box	If selected, no elements based on this type can appear in the message.

Content Validation properties for complex types:

Content Validation specifies where the objects that are included within a complex type or group are defined, if at all. It is used in combination with *Composition*.

The first table below shows the valid settings for *Content Validation* if *Composition* is set to Message, and the second table shows the valid settings for *Content Validation* if *Composition* is not set to Message.

Content Validation options if Composition is set to Message

Option	Meaning
Open	When a message is parsed, this complex type or group can contain any message, not just those that you have defined in this message set. You can use this option for sparse messages (see Predefined and self-defining elements and messages for a definition of sparse messages).
Closed	When a message is parsed, this complex type or group can only contain the messages that are members of this complex type or group. This is always the case for messages represented in CWF format.
Open Defined	When a message is parsed, this complex type or group can contain any message defined within the message set.

Content Validation options if Composition is not set to Message

Option	Meaning
Open	When a message is parsed, this complex type or group can contain any elements and not just those that you have defined in this message set (see Predefined and self-defining elements and messages for a definition of sparse messages).
Closed	When a message is parsed, this complex type or group can only contain the elements that are members of this complex type or group.
Open Defined	When a message is parsed, this complex type or group can contain any element that you have defined within the message set.

Element reference logical properties

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Occurrences

Property	Type	Meaning
Min Occurs	Integer	Specify the minimum number of times that the object can repeat. The default is 1. If the value is set to 0, then the object is optional. If a value is set, it must be less than or equal to the value in <i>Max Occurs</i> .
Max Occurs	Integer	Specify the maximum number of times that the object can repeat. The default is 1. If this property is not set, then the object can not occur more than once. It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.

The Min Occurs and Max Occurs properties are used in conjunction with an elements Value properties. The table below summarizes how an element reference can be constrained.

Min Occurs	Max Occurs	Fixed	Default	Notes
1	1			The element must appear once and can have any value.
1	1	Delta		The element must appear once and it must match the data that has been entered in the <i>Value</i> property. In this example the element must contain the text <i>Delta</i> .
2	-1	Delta		The element must appear twice or more and it must match the data that has been entered in the <i>Value</i> property. In this example there will be at least two elements that must contain the text <i>Delta</i> .
0	1			The element is optional and can appear once and have any value.
0	1	Delta		The element is optional and can appear once. If it does appear, its value must match the data that has been entered in the <i>Value</i> property. If it does not appear its value will be the data that has been entered in the <i>Value</i> property.
0	1		Delta	The element is optional and can appear once. If it does not appear, its value will be the data that has been entered in the <i>Value</i> property. If it does appear it must be the value given in the element.

Min Occurs	Max Occurs	Fixed	Default	Notes
0	2		Delta	The element is optional and can appear once, twice or not at all. If the element does not appear it is not provided. If the element appears and it is empty, it set to the data held in the <i>Value</i> property, else it is the value given in the element.
0	0			The element is prohibited and must not appear.

Global attribute logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>XML</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • int • string • boolean • hexBinary • dateTime • date • time • decimal • float • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Value properties

The *Value* properties are used in conjunction with the *Usage* property in an Attribute Reference or a Local Attribute.

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>

Global attribute group logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <code>xml</code> or any variant (for example <code>XML</code>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>

Global element logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <code>xml</code> or any variant (for example <code>Xml</code>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • <code>int</code> • <code>string</code> • <code>boolean</code> • <code>hexBinary</code> • <code>dateTime</code> • <code>date</code> • <code>time</code> • <code>decimal</code> • <code>float</code> • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Value

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>
Nullable	Check box	Select this if you want the element to be able to be defined as null. This is distinct from being empty where there is no data in the element.

Substitution settings

Substitution Groups provide a means by which one element may be substituted for another in a message. The element which can be substituted is called the 'head' element, and the substitution group is the list of elements that may be used in its place. An element can be in at most one substitution group.

Property	Type	Meaning
Final	Enumerated type	<p>You use this property to limit the set of elements which may belong to its substitution group.</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type. • extension. Prohibit element substitution by elements whose types are extensions of the head element's type. • all. Prohibit substitution by any method.

Property	Type	Meaning
Block	Enumerated type	<p>You use this property to limit the set of elements which may be substituted for this element in a message.</p> <p>Select from:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type • extension. Prohibit element substitution by elements whose types are extensions of the head element's type • substitution. Prohibit element substitution by members of the element's substitution group. • #all. Prohibit substitution by any method.
Substitution Group	Enumerated type	Use this property to specify the name of a 'head' element. Setting this property indicates that this element is a member of the substitution group for the 'head' element.
Abstract	Check box	Select this if you do not want the element to appear in the message, but require one of the members of its substitution group to appear in its place.

Global group logical properties

Valid children in a global group that depend on both **Composition** and **Content Validation** are shown in "Content Validation properties for complex types" on page 27.

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <code>xml</code> or any variant (for example <code>XML</code>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Composition	Enumerated type	<p>The following only applies to the element content of a complex type and has no effect on the attribute content of a complex type. Select from:</p> <ul style="list-style-type: none"> • Empty • sequence. If you select this option, you can define children that are elements or groups. These children, if present, must appear in the specified order. They can repeat and can be duplicated. • choice. If you select this option, you can define children that are elements or groups. Only one of the defined children of the complex type can be present, but repeating children are allowed. <p>When a complex type whose <i>Composition</i> is set to Choice is present in an output message, the bit stream contains the number of bytes represented by the longest item (padded with 0x00).</p> <p>Use this option if you want to model C unions and COBOL REDEFINES in a Custom Wire Format, or an XML DTD element that uses choice in an XML Wire Format, or some industry standard tagged/delimited messages (for example SWIFT) use this format.</p> <ul style="list-style-type: none"> • all. The elements in an all group can appear in any order. Each element can appear once, or not at all. An all group can only contain elements - groups are not allowed. An all group can only be used at the top level of a complex type - it cannot be a member of another group within a type. • unorderedSet. If you select this option, you can define only elements as children. The elements can repeat but cannot be duplicated. Child elements can appear in any order. • orderedSet. If you select this option, you can define only elements as children. These elements, if present, must appear in the specified order, and they can repeat but cannot be duplicated. This is the default value for new complex types. • message. If you select this option, you can define only messages as children. They can repeat, but they cannot be duplicated. Like choice, only one of the defined children can be present. <p>If the complex type includes more than one message, the bit stream contains the exact length of the embedded message, and is not padded to the length of the longest.</p> <p>Use this option to model multipart messages, which are used in some industry standards, for example, SWIFT. For more information, see the section on multipart messages in Multipart messages.</p>
Content Validation	Enumerated type	<p><i>Content Validation</i> controls how the broker responds to undeclared content and specifies where the objects that are included within the complex type are defined, if at all. It is used in combination with the <i>Composition</i> property.</p> <p>Options:</p> <ul style="list-style-type: none"> • Closed. The complex type can only contain the child elements that you have added to it. • Open Defined. The complex type can contain any valid element defined within the message set. • Open. The complex type can contain any valid element, not just those that you have added to this complex type. <p>See “Combinations of Composition and Content Validation” on page 123 for further details of these options.</p>

Group reference logical properties

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Occurrence properties

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Local attribute logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none">• - the hyphen• _ the underscore• . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • int • string • boolean • hexBinary • dateTime • date • time • decimal • float • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Value properties

The *Value* properties are used in conjunction with the *Usage* property in an Attribute Reference or a Local Attribute.

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>

Usage properties

Property	Type	Meaning
Usage	Enumerated type	<p>The usage property is used in conjunction with the <i>Value</i> property found in an attribute object. The default for the <i>Usage</i> property is optional.</p> <p>Select from;</p> <ul style="list-style-type: none"> • optional. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute can appear once and can have any value. – Where the <i>Value</i> property is set to default, the attribute can appear once. If it does not appear, its value will be the data that has been entered in the <i>Value</i> property. If it does appear it will be the value given. – Where the <i>Value</i> property is set to fixed, the attribute can appear once. If it does appear, its value must match the data that has been entered in the <i>Value</i> property. If it does not appear its value will be the data that has been entered in the <i>Value</i> property. • prohibited. The attribute must not appear. • required. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute must appear once and can have any value. – Where the <i>Value</i> property is set to fixed, the attribute must appear once and it must match the data that has been entered in the <i>Value</i> property.

Key logical properties

There are no properties to show.

Keyref logical properties

There are no properties to show.

Local element logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • int • string • boolean • hexBinary • dateTime • date • time • decimal • float • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Occurrences

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Value

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>
Nilable	Check box	Select this if you want the element to be able to be defined as null. This is distinct from being empty where there is no data in the element.

Substitution settings

Substitution Groups provide a means by which one element may be substituted for another in a message. The element which can be substituted is called the 'head' element, and the substitution group is the list of elements that may be used in its place. An element can be in at most one substitution group.

Property	Type	Meaning
Final	Enumerated type	<p>You use this property to limit the set of elements which may belong to its substitution group.</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type. • extension. Prohibit element substitution by elements whose types are extensions of the head element's type. • all. Prohibit substitution by any method.
Block	Enumerated type	<p>You use this property to limit the set of elements which may be substituted for this element in a message.</p> <p>Select from:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type • extension. Prohibit element substitution by elements whose types are extensions of the head element's type • substitution. Prohibit element substitution by members of the element's substitution group. • #all. Prohibit substitution by any method.
Substitution Group	Enumerated type	<p>Use this property to specify the name of a 'head' element. Setting this property indicates that this element is a member of the substitution group for the 'head' element.</p>
Abstract	Check box	<p>Select this if you do not want the element to appear in the message, but require one of the members of its substitution group to appear in its place.</p>

Local group logical properties

Valid children in a local group that depend on both **Composition** and **Content Validation** are shown in "Content Validation properties for complex types" on page 27.

Property	Type	Meaning
Composition	Enumerated type	<p>The following only applies to the element content of a complex type and has no effect on the attribute content of a complex type. Select from:</p> <ul style="list-style-type: none"> • Empty • sequence. If you select this option, you can define children that are elements or groups. These children, if present, must appear in the specified order. They can repeat and can be duplicated. • choice. If you select this option, you can define children that are elements or groups. Only one of the defined children of the complex type can be present, but repeating children are allowed. <p>When a complex type whose <i>Composition</i> is set to Choice is present in an output message, the bit stream contains the number of bytes represented by the longest item (padded with 0x00).</p> <p>Use this option if you want to model C unions and COBOL REDEFINES in a Custom Wire Format, or an XML DTD element that uses choice in an XML Wire Format, or some industry standard tagged/delimited messages (for example SWIFT) use this format.</p> <ul style="list-style-type: none"> • all. The elements in an all group can appear in any order. Each element can appear once, or not at all. An all group can only contain elements - groups are not allowed. An all group can only be used at the top level of a complex type - it cannot be a member of another group within a type. • unorderedSet. If you select this option, you can define only elements as children. The elements can repeat but cannot be duplicated. Child elements can appear in any order. • orderedSet. If you select this option, you can define only elements as children. These elements, if present, must appear in the specified order, and they can repeat but cannot be duplicated. This is the default value for new complex types. • message. If you select this option, you can define only messages as children. They can repeat, but they cannot be duplicated. Like choice, only one of the defined children can be present. <p>If the complex type includes more than one message, the bit stream contains the exact length of the embedded message, and is not padded to the length of the longest.</p> <p>Use this option to model multipart messages, which are used in some industry standards, for example, SWIFT. For more information, see the section on multipart messages in Multipart messages.</p>
Content Validation	Enumerated type	<p><i>Content Validation</i> controls how the broker responds to undeclared content and specifies where the objects that are included within the complex type are defined, if at all. It is used in combination with the <i>Composition</i> property.</p> <p>Options:</p> <ul style="list-style-type: none"> • Closed. The complex type can only contain the child elements that you have added to it. • Open Defined. The complex type can contain any valid element defined within the message set. • Open. The complex type can contain any valid element, not just those that you have added to this complex type. <p>See “Combinations of Composition and Content Validation” on page 123 for further details of these options.</p>

Occurrences

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Message logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>

Simple type logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <code>xml</code> or any variant (for example <code>Xml</code>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Base Type	Enumerated type	You can use this property to select a base type that is used as the starting point to define a new simple type that is derived by setting additional value constraints.

A simple type can also have “Simple type logical value constraints.”

Simple type logical value constraints:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

<p>Binary types</p> <ul style="list-style-type: none"> - base64Binary - hexBinary 	<p>Boolean types</p> <ul style="list-style-type: none"> - boolean 	<p>DateTime types</p> <ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<p>Decimal types</p> <ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
---	--	--	---

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
------------------------------------	---	------------------------------	--

Unique logical properties

There are no properties to show.

Wildcard attribute logical properties

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Property	Type	Meaning
Process Content	Enumerated type	<p>If a message contains an attribute that corresponds to a wildcard in the message model, <i>Process Content</i> defines how the attribute is validated.</p> <p>Select from;</p> <ul style="list-style-type: none"> • strict. The parser can only match against attributes declared in the specified namespace. • lax. The parser attempts to match against attributes declared in any accessible namespace. If the specified namespace cannot be found, an error will not be generated. • skip. If you select skip the parser does not perform any validation on the attribute.

Wildcard element logical properties

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Property	Type	Meaning
Process Content	Enumerated type	<p>If a message contains an attribute that corresponds to a wildcard in the message model, <i>Process Content</i> defines how the attribute is validated.</p> <p>Select from;</p> <ul style="list-style-type: none">• strict. The parser can only match against attributes declared in the specified namespace.• lax. The parser attempts to match against attributes declared in any accessible namespace. If the specified namespace cannot be found, an error will not be generated.• skip. If you select skip the parser does not perform any validation on the attribute.

Occurrences

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Physical properties for message model objects

Property information is available for objects within:

- “Custom Wire Format physical properties for message model objects” on page 47
- “XML wire format physical properties for message model objects” on page 53
- “Tagged/delimited string format physical properties for message model objects” on page 62

Custom Wire Format physical properties for message model objects

Custom wire format physical property information is available for the following objects:

- "Attribute group reference CWF properties"
- "Attribute reference CWF properties"
- "Complex type CWF properties" on page 48
- "Element reference CWF properties" on page 48
- "Global attribute CWF properties" on page 49
- "Global attribute group CWF properties" on page 49
- "Global element CWF properties" on page 49
- "Global group CWF properties" on page 49
- "Group reference CWF properties" on page 49
- "Key CWF properties" on page 50
- "Keyref CWF properties" on page 50
- "Local element CWF properties" on page 51
- "Local group CWF properties" on page 52
- "Message CWF properties" on page 52
- "Simple type CWF properties" on page 52
- "Unique CWF properties" on page 52
- "Wildcard attribute CWF properties" on page 53
- "Wildcard element CWF properties" on page 53

Attribute group reference CWF properties:

There are no properties to show.

Attribute reference CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<div>- base64Binary</div> <div>- hexBinary</div>	<div>- boolean</div>	<div>- date</div> <div>- dateTime</div> <div>- gDay</div> <div>- gMonth</div> <div>- gMonthDay</div> <div>- gYear</div> <div>- gYearMonth</div> <div>- time</div>	<div>- decimal</div> <div>- integer</div> <div>- negativeInteger</div> <div>- nonNegativeInteger</div> <div>- nonPositiveInteger</div> <div>- positiveInteger</div>

Float types <ul style="list-style-type: none"> - double - float 	Integer types <ul style="list-style-type: none"> - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort 	Interval types <ul style="list-style-type: none"> - duration¹ 	String types <ul style="list-style-type: none"> - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: <ol style="list-style-type: none"> 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types. 			

Complex type CWF properties:

|

There are no properties to show.

Element reference CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types <ul style="list-style-type: none"> - base64Binary - hexBinary 	Boolean types <ul style="list-style-type: none"> - boolean 	DateTime types <ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	Decimal types <ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
--	---	---	--

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Global attribute CWF properties:

| There are no properties to show.

Global attribute group CWF properties:

| There are no properties to show.

Global element CWF properties:

| There are no properties to show.

Global group CWF properties:

| There are no properties to show.

Group reference CWF properties: Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.

Property	Type	Meaning
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

Key CWF properties:

There are no properties to show.

Keyref CWF properties:

There are no properties to show.

Local attribute CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Local element CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken

Note:

1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.

Local group CWF properties:
Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

Message CWF properties:

| There are no properties to show.

Simple type CWF properties:

| There are no properties to show.

Unique CWF properties:

| There are no properties to show.

Wildcard attribute CWF properties:

| There are no properties to show.

Wildcard element CWF properties:

| There are no properties to show.

XML wire format physical properties for message model objects

XML wire format physical property information is available for the following objects:

- "Attribute group reference XML properties"
- "Attribute reference XML properties"
- "Complex type XML properties" on page 54
- "Element reference XML properties" on page 54
- "Global attribute XML properties" on page 55
- "Global attribute group XML properties" on page 56
- "Global element XML properties" on page 56
- "Global group XML properties" on page 56
- "Group reference XML properties" on page 56
- | • "Key XML properties" on page 56
- | • "Keyref XML properties" on page 57
- "Local attribute XML properties" on page 57
- "Local element XML properties" on page 57
- "Local group XML properties" on page 58
- "Message XML properties" on page 58
- "Simple type XML properties" on page 62
- | • "Unique XML properties" on page 62
- "Wildcard attribute XML properties" on page 62
- "Wildcard element XML properties" on page 62

Attribute group reference XML properties:

| There are no properties to show.

Attribute reference XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Complex type XML properties:

I There are no properties to show.

Element reference XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
---	----------------------------	--	---

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Global attribute XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken

Note:

1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.

Global attribute group XML properties:

There are no properties to show.

Global element XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken

Note:

1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.

Global group XML properties:

There are no properties to show.

Group reference XML properties:

There are no properties to show.

Key XML properties:

There are no properties to show.

Keyref XML properties:

There are no properties to show.

Local attribute XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types <ul style="list-style-type: none">- base64Binary- hexBinary	Boolean types <ul style="list-style-type: none">- boolean	DateTime types <ul style="list-style-type: none">- date- dateTime- gDay- gMonth- gMonthDay- gYear- gYearMonth- time	Decimal types <ul style="list-style-type: none">- decimal- integer- negativeInteger- nonNegativeInteger- nonPositiveInteger- positiveInteger
Float types <ul style="list-style-type: none">- double- float	Integer types <ul style="list-style-type: none">- byte- int- long- short- unsignedByte- unsignedInt- unsignedLong- unsignedShort	Interval types <ul style="list-style-type: none">- duration¹	String types <ul style="list-style-type: none">- anyURI- ENTITIES- ENTITY- ID- IDREF- IDREFS- language- Name- NCName- NMTOKEN- NMTOKENS- normalizedString- NOTATION- QName- stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Local element XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Local group XML properties:

There are no properties to show.

Message XML properties: Namespace schema locations

This property is only active if namespaces have been enabled.

Property	Type	Meaning
Namespace URI	String	A unique string, usually in the form of a URL that identifies the schema for this If namespaces have not been enabled, this property will display <no target namespace>. This property will override the same property at the message set level.
Schema location	String	Enter the location of the schema for the associated namespace name that will be used to validate objects within the namespace.

XML declarations

Property	Type	Meaning
Output Namespace Declaration	Enumerated Type	<p>The <i>Output Namespace Declaration</i> property controls where the namespace declarations will be placed in the output XML document.</p> <p>Select from:</p> <ul style="list-style-type: none"> At start of document. Declarations for all of the entries in the <i>Namespace schema locations</i> table above will be output as attributes of the message in the output XML document. The disadvantage of this option is that in some cases unnecessary declarations may be output. As required. Declarations will only be output when required by an element or attribute that is in that namespace. The disadvantage of this option is that the same namespace declaration may need to be output more than once in the output XML document. <p>The default option is At start of document.</p> <p>This property is only active if namespaces are enabled for this message set.</p>

XML document type settings

Property	Type	Meaning
DOCTYPE System ID	String	<p>Specify the System ID for DOCTYPE external DTD subset. It overrides the equivalent message set property setting for this particular message.</p> <p>If the message set property <i>Suppress DOCTYPE</i> is set to Yes, this parameter is ignored and cannot be changed (the field is disabled) .</p> <p>The default value is the value that you specified for the <i>DOCTYPE System ID</i> property for the message set.</p>
DOCTYPE Public ID	String	<p>Specify the Public ID for DOCTYPE external DTD subset. It overrides the equivalent message set property setting for this particular message.</p> <p>If the message set property <i>Suppress DOCTYPE</i> is set to Yes, this parameter is ignored and cannot be changed (the field is disabled) .</p> <p>The default value is the value that you specified for the <i>DOCTYPE Public ID</i> property for the message set.</p>
DOCTYPE Text	String	<p>Enter optional additional text to include within the DOCTYPE. It overrides the message set property for this particular message.</p> <p>If the message set property <i>Suppress DOCTYPE</i> is set to Yes, this parameter is ignored and cannot be changed (the field is disabled) .</p> <p>For more information, see “In-line DTDs and the DOCTYPE text property” on page 12.</p> <p>The default value is the value that you specified for the <i>DOCTYPE Text</i> property for the message set.</p>

Property	Type	Meaning
Root Tag Name*	String	<p>Specify the name of the root tag for a message bit stream XML document. It overrides the message set property set for this message.</p> <p>The default value is the value that you specified for the <i>Root Tag Name</i> property for the message set.</p>

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>

XML Message rendering options:

There are four properties on the XML layer that you can use to affect how the XML messages are rendered. The table below shows examples of the values that you can set for the *Member Render* property. In this table, the member element is referred to as *A*, and has the value value of element. The parent is referred to as *X*.

The effect of rendering options on XML output

To get XML rendered like this:	Set this Member Render property value:	Set these other property values:
<pre><X> <A>value of element </X></pre>	XMLElement (the default)	Member XML Name = A
<pre><X A='value of element' /></pre>	XMLAttribute	Member XML Name = A
<pre><X> <Field id='A'>value of element</Field> </X></pre>	XMLElementAttrID	Member XML Name = Field Member ID Attribute Name = id Member ID Attribute Value = A
<pre><X> </X></pre>	XMLElementAttrVal	Member XML Name = A Member Value Attribute Name = val
<pre><X> <Field id='A' val='value of element' /> </X></pre>	XMLElementAttrIDVal	Member XML Name = Field Member ID Attribute Name = id Member ID Attribute Value = A Member Value Attribute Name = val

Simple type XML properties:

There are no properties to show.

Unique XML properties:

There are no properties to show.

Wildcard attribute XML properties:

There are no properties to show.

Wildcard element XML properties:

There are no properties to show.

Tagged/delimited string format physical properties for message model objects

TDS format physical property information is available for the following objects:

- “Attribute group reference TDS properties”
- “Attribute reference TDS properties”
- “Complex type TDS properties” on page 63
- “Element reference TDS properties” on page 66
- “Global attribute TDS properties” on page 67
- “Global attribute group TDS properties” on page 67
- “Global element TDS properties” on page 67
- “Global group TDS properties” on page 68
- “Group reference TDS properties” on page 71
- “Key TDS properties” on page 71
- “Keyref TDS properties” on page 71
- “Local attribute TDS properties” on page 71
- “Local element TDS properties” on page 72
- “Local group TDS properties” on page 73
- “Message TDS properties” on page 76
- “Simple type TDS properties” on page 76
- “Unique TDS properties” on page 76
- “Wildcard attribute TDS properties” on page 76
- “Wildcard element TDS properties” on page 76

Attribute group reference TDS properties:

There are no properties to show.

Attribute reference TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Complex type TDS properties: Field Identification

If the complex type is based on a global group, the TDS properties listed below will actually be located on the global group. If this is the case, any changes to these properties will be applied to the global group, and will affect all references to the group (including any other complex types which are based on it).

Property	Type	Meaning
Data Element Separation	Enumerated Type	<p>Specify the method used to separate the data elements within the type. Select one of the following values:</p> <ul style="list-style-type: none"> • Tagged Delimited. This value indicates that all elements within the complex type are identified by a tag, and separated by the value specified in the optional <i>Delimiter</i> property (if specified). You must set the <i>Tag</i> property for all child elements of simple type, and you may set the <i>Delimiter</i> property to a non-empty value. See “Global element TDS properties” on page 67. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Fixed Length. This value indicates that each element is identified by a tag, and the data has a fixed length. There are no delimiters. You must set the <i>Tag</i> property for each of the child elements of this complex type, and each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Encoded Length. This value indicates that all elements within the complex type are separated by a tag, and a length field follows each tag. There are no delimiters. The tag can be fixed length, as set by <i>Length of Tag</i>, or variable length delimited by the <i>Tag Data Separator</i>. You must also set <i>Length Of Encoded Length</i> so that the parser knows the size of the length field, and set <i>Extra Chars in Encoded Length</i> to tell the parser how many to subtract from the value in <i>Length Of Encoded Length</i> to get the actual length of data that follows the length field. <p>This method provides a more flexible way of handling ACORD AL3 standard messages than Fixed Length AL3, by allowing different parts of the messages to be at different versions of the ACORD AL3 standard.</p> <ul style="list-style-type: none"> • All Elements Delimited. This value indicates that all elements within the complex type are separated by a delimiter. You must set the value in the <i>Delimiter</i> property. • Variable Length Elements Delimited. This value indicates that some of the elements within the complex type might be of variable length: if they are, they must be delimited by the value specified in the <i>Delimiter</i> property. • Use Data Pattern. This value indicates that the parser determines the elements by matching the data with the regular expression set in the element or type member <i>Data Pattern</i> property. See “Message definition file properties” on page 20. • Fixed Length. This value indicates that all elements within the complex type are fixed length. The next data element is accessed by adding the value of the <i>Length</i> property to the offset (see “Global element TDS properties” on page 67). If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length. Each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. • Fixed Length AL3. This value has a similar meaning to the separation type Fixed Length, but also indicates to the parser that a number of predefined rules with regard to missing optional elements, encoded lengths, and versioning must be applied. If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length AL3, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length AL3. • Undefined. This value is set automatically if you set the <i>Type Composition</i> property of a complex type to Message, and you cannot change it to any other value. <p>Do not set the <i>Type Composition</i> property to Empty, Choice, Unordered Set, Ordered Set, Sequence, or Simple Unordered Set. If you do, you will be unable to check in the type.</p>
Group Indicator	String	Specify the value of a special character or string that precedes the data belonging to a group or complex type within the bit stream.

Property	Type	Meaning
Group Terminator	String	Specify the value of a special character or string that terminates data belonging to a group or a complex type within the bit stream.
Delimiter	String	Specify the value of a special character or string that specifies the delimiter used between data elements. This property applies only to the delimited <i>Data Element Separation</i> methods (Tagged Delimited, All Elements Delimited, and Variable Elements Delimited).
Suppress Absent Element Delimiters	Enumerated type	Use this property to select if you want delimiters to be suppressed for elements that are missing within a message. Select from: <ul style="list-style-type: none"> • End Of Type. Use this option to suppress the delimiter when an element is missing. For example, if the model has been defined to have up to 3 elements and only 2 are present, the last delimiter can be omitted from the message. • Never. Use this option to ensure that even if optional elements are not present, all delimiters will be written out. This option should be used when the delimiter used to delimit parent and child objects is the same. For example, if an optional child element is missing, message processing applications could not tell where the child elements in a message ended and the next parent element started if the delimiters are all the same.
Observe Element Length	Check box	Applicable when <i>Data Element Separation</i> is All Elements Delimited and tells the TDS parser to take any <i>Length</i> property of child elements or attributes into account. The default value depends on the setting of the <i>Messaging Standard</i> property (at the message set level) and <i>Data Element Separation</i> properties. <ul style="list-style-type: none"> • Where <i>Data Element Separation</i> is All Elements Delimited and the <i>Messaging Standard</i> is set to TL0G , this property should be set. For all other messaging standards it should not be set. • Where <i>Data Element Separation</i> is Tagged Delimited this property should not be set. • Where <i>Data Element Separation</i> is Tagged Fixed Length, Fixed Length, Fixed Length AL3, or Variable Length Elements Delimited this property will be set and is disabled. • For all other data element separation methods, this property is not set and is disabled. Any other combination will generate a task list warning.
Tag Data Separator	Button and String	Specify the value of a special character or string that separates the Tag from the data. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides <i>Length of Tag</i> . This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).
Length of Tag	Integer	Specify the length of a tag value. When the message is parsed, this allows tags to be extracted from the bit stream if the <i>Tag Data Separator</i> property is not set. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides this value. This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).

Property	Type	Meaning
Length of Encoded Length	Integer	<p>Specifies the number of characters (not bytes) after a tag that are used for the length field. Enter a value from 0 to 2147483647.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length; it is not valid otherwise.</p> <p>The actual number of data characters parsed also depends on the value of the <i>Extra Chars in Encoded Length</i> property.</p>
Extra Chars in Encoded Length	Integer	<p>(Only valid if the <i>Data Element Separation</i> method is set to Tagged Encoded Length.) Specifies the number of extra characters included in the value found in the length field. (For example, the value in the length might include the size of the length field itself as well as the size of the data field, or it might be the total size of the tag, length, and data fields.)</p> <p>Enter a value from 0 to 2147483647. The parser subtracts this number from the number found in the length field to get the number of data characters that follow the length field.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length and the actual number of data characters is less than the value found in the length field.</p>

Element reference TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types <ul style="list-style-type: none"> - base64Binary - hexBinary 	Boolean types <ul style="list-style-type: none"> - boolean 	DateTime types <ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	Decimal types <ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types <ul style="list-style-type: none"> - double - float 	Integer types <ul style="list-style-type: none"> - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort 	Interval types <ul style="list-style-type: none"> - duration¹ 	String types <ul style="list-style-type: none"> - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken

Note:

1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.

Global attribute TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken

Note:

1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.

Global attribute group TDS properties:

There are no properties to show.

Global element TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types <ul style="list-style-type: none"> - base64Binary - hexBinary 	Boolean types <ul style="list-style-type: none"> - boolean 	DateTime types <ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	Decimal types <ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types <ul style="list-style-type: none"> - double - float 	Integer types <ul style="list-style-type: none"> - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort 	Interval types <ul style="list-style-type: none"> - duration¹ 	String types <ul style="list-style-type: none"> - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Global group TDS properties:

Field Identification

Property	Type	Meaning
Data Element Separation	Enumerated Type	<p>Specify the method used to separate the data elements within the type. Select one of the following values:</p> <ul style="list-style-type: none"> • Tagged Delimited. This value indicates that all elements within the complex type are identified by a tag, and separated by the value specified in the optional <i>Delimiter</i> property (if specified). You must set the <i>Tag</i> property for all child elements of simple type, and you may set the <i>Delimiter</i> property to a non-empty value. See “Global element TDS properties” on page 67. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Fixed Length. This value indicates that each element is identified by a tag, and the data has a fixed length. There are no delimiters. You must set the <i>Tag</i> property for each of the child elements of this complex type, and each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Encoded Length. This value indicates that all elements within the complex type are separated by a tag, and a length field follows each tag. There are no delimiters. The tag can be fixed length, as set by <i>Length of Tag</i>, or variable length delimited by the <i>Tag Data Separator</i>. You must also set <i>Length Of Encoded Length</i> so that the parser knows the size of the length field, and set <i>Extra Chars in Encoded Length</i> to tell the parser how many to subtract from the value in <i>Length Of Encoded Length</i> to get the actual length of data that follows the length field. This method provides a more flexible way of handling ACORD AL3 standard messages than Fixed Length AL3, by allowing different parts of the messages to be at different versions of the ACORD AL3 standard. • All Elements Delimited. This value indicates that all elements within the complex type are separated by a delimiter. You must set the value in the <i>Delimiter</i> property. • Variable Length Elements Delimited. This value indicates that some of the elements within the complex type might be of variable length: if they are, they must be delimited by the value specified in the <i>Delimiter</i> property. • Use Data Pattern. This value indicates that the parser determines the elements by matching the data with the regular expression set in the element or type member <i>Data Pattern</i> property. See “Message definition file properties” on page 20. • Fixed Length. This value indicates that all elements within the complex type are fixed length. The next data element is accessed by adding the value of the <i>Length</i> property to the offset (see “Global element TDS properties” on page 67). If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length. Each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. • Fixed Length AL3. This value has a similar meaning to the separation type Fixed Length, but also indicates to the parser that a number of predefined rules with regard to missing optional elements, encoded lengths, and versioning must be applied. If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length AL3, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length AL3. • Undefined. This value is set automatically if you set the <i>Type Composition</i> property of a complex type to Message, and you cannot change it to any other value. Do not set the <i>Type Composition</i> property to Empty, Choice, Unordered Set, Ordered Set, Sequence, or Simple Unordered Set. If you do, you will be unable to check in the type.

Property	Type	Meaning
Group Indicator	String	Specify the value of a special character or string that precedes the data belonging to a group or complex type within the bit stream.
Group Terminator	String	Specify the value of a special character or string that terminates data belonging to a group or a complex type within the bit stream.
Delimiter	String	Specify the value of a special character or string that specifies the delimiter used between data elements. This property applies only to the delimited <i>Data Element Separation</i> methods (Tagged Delimited, All Elements Delimited, and Variable Elements Delimited).
Suppress Absent Element Delimiters	Enumerated type	Use this property to select if you want delimiters to be suppressed for elements that are missing within a message. Select from: <ul style="list-style-type: none"> • End Of Type. Use this option to suppress the delimiter when an element is missing. For example, if the model has been defined to have up to 3 elements and only 2 are present, the last delimiter can be omitted from the message. • Never. Use this option to ensure that even if optional elements are not present, all delimiters will be written out. This option should be used when the delimiter used to delimit parent and child objects is the same. For example, if an optional child element is missing, message processing applications could not tell where the child elements in a message ended and the next parent element started if the delimiters are all the same.
Observe Element Length	Check box	Applicable when <i>Data Element Separation</i> is All Elements Delimited and tells the TDS parser to take any <i>Length</i> property of child elements or attributes into account. The default value depends on the setting of the <i>Messaging Standard</i> property (at the message set level) and <i>Data Element Separation</i> properties. <ul style="list-style-type: none"> • Where <i>Data Element Separation</i> is All Elements Delimited and the <i>Messaging Standard</i> is set to TLOG , this property should be set. For all other messaging standards it should not be set. • Where <i>Data Element Separation</i> is Tagged Delimited this property should not be set. • Where <i>Data Element Separation</i> is Tagged Fixed Length, Fixed Length, Fixed Length AL3, or Variable Length Elements Delimited this property will be set and is disabled. • For all other data element separation methods, this property is not set and is disabled. Any other combination will generate a task list warning.
Tag Data Separator	Button and String	Specify the value of a special character or string that separates the Tag from the data. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides <i>Length of Tag</i> . This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).
Length of Tag	Integer	Specify the length of a tag value. When the message is parsed, this allows tags to be extracted from the bit stream if the <i>Tag Data Separator</i> property is not set. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides this value. This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).

Property	Type	Meaning
Length of Encoded Length	Integer	<p>Specifies the number of characters (not bytes) after a tag that are used for the length field. Enter a value from 0 to 2147483647.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length; it is not valid otherwise.</p> <p>The actual number of data characters parsed also depends on the value of the <i>Extra Chars in Encoded Length</i> property.</p>
Extra Chars in Encoded Length	Integer	<p>(Only valid if the <i>Data Element Separation</i> method is set to Tagged Encoded Length.) Specifies the number of extra characters included in the value found in the length field. (For example, the value in the length might include the size of the length field itself as well as the size of the data field, or it might be the total size of the tag, length, and data fields.)</p> <p>Enter a value from 0 to 2147483647. The parser subtracts this number from the number found in the length field to get the number of data characters that follow the length field.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length and the actual number of data characters is less than the value found in the length field.</p>

Group reference TDS properties:
Field Identification

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Key TDS properties:

There are no properties to show.

Keyref TDS properties:

There are no properties to show.

Local attribute TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

Local element TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
---	--------------------------------	--	---

Float types <ul style="list-style-type: none"> - double - float 	Integer types <ul style="list-style-type: none"> - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort 	Interval types <ul style="list-style-type: none"> - duration¹ 	String types <ul style="list-style-type: none"> - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: <ol style="list-style-type: none"> 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types. 			

Local group TDS properties:

Field Identification

Property	Type	Meaning
Data Element Separation	Enumerated Type	<p>Specify the method used to separate the data elements within the type. Select one of the following values:</p> <ul style="list-style-type: none"> • Tagged Delimited. This value indicates that all elements within the complex type are identified by a tag, and separated by the value specified in the optional <i>Delimiter</i> property (if specified). You must set the <i>Tag</i> property for all child elements of simple type, and you may set the <i>Delimiter</i> property to a non-empty value. See “Global element TDS properties” on page 67. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Fixed Length. This value indicates that each element is identified by a tag, and the data has a fixed length. There are no delimiters. You must set the <i>Tag</i> property for each of the child elements of this complex type, and each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Encoded Length. This value indicates that all elements within the complex type are separated by a tag, and a length field follows each tag. There are no delimiters. The tag can be fixed length, as set by <i>Length of Tag</i>, or variable length delimited by the <i>Tag Data Separator</i>. You must also set <i>Length Of Encoded Length</i> so that the parser knows the size of the length field, and set <i>Extra Chars in Encoded Length</i> to tell the parser how many to subtract from the value in <i>Length Of Encoded Length</i> to get the actual length of data that follows the length field. <p>This method provides a more flexible way of handling ACORD AL3 standard messages than Fixed Length AL3, by allowing different parts of the messages to be at different versions of the ACORD AL3 standard.</p> <ul style="list-style-type: none"> • All Elements Delimited. This value indicates that all elements within the complex type are separated by a delimiter. You must set the value in the <i>Delimiter</i> property. • Variable Length Elements Delimited. This value indicates that some of the elements within the complex type might be of variable length: if they are, they must be delimited by the value specified in the <i>Delimiter</i> property. • Use Data Pattern. This value indicates that the parser determines the elements by matching the data with the regular expression set in the element or type member <i>Data Pattern</i> property. See “Message definition file properties” on page 20. • Fixed Length. This value indicates that all elements within the complex type are fixed length. The next data element is accessed by adding the value of the <i>Length</i> property to the offset (see “Global element TDS properties” on page 67). If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length. Each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. • Fixed Length AL3. This value has a similar meaning to the separation type Fixed Length, but also indicates to the parser that a number of predefined rules with regard to missing optional elements, encoded lengths, and versioning must be applied. If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length AL3, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length AL3. • Undefined. This value is set automatically if you set the <i>Type Composition</i> property of a complex type to Message, and you cannot change it to any other value. <p>Do not set the <i>Type Composition</i> property to Empty, Choice, Unordered Set, Ordered Set, Sequence, or Simple Unordered Set. If you do, you will be unable to check in the type.</p>

Property	Type	Meaning
Group Indicator	String	Specify the value of a special character or string that precedes the data belonging to a group or complex type within the bit stream.
Group Terminator	String	Specify the value of a special character or string that terminates data belonging to a group or a complex type within the bit stream.
Delimiter	String	Specify the value of a special character or string that specifies the delimiter used between data elements. This property applies only to the delimited <i>Data Element Separation</i> methods (Tagged Delimited, All Elements Delimited, and Variable Elements Delimited).
Suppress Absent Element Delimiters	Enumerated type	Use this property to select if you want delimiters to be suppressed for elements that are missing within a message. Select from: <ul style="list-style-type: none"> • End Of Type. Use this option to suppress the delimiter when an element is missing. For example, if the model has been defined to have up to 3 elements and only 2 are present, the last delimiter can be omitted from the message. • Never. Use this option to ensure that even if optional elements are not present, all delimiters will be written out. This option should be used when the delimiter used to delimit parent and child objects is the same. For example, if an optional child element is missing, message processing applications could not tell where the child elements in a message ended and the next parent element started if the delimiters are all the same.
Observe Element Length	Check box	Applicable when <i>Data Element Separation</i> is All Elements Delimited and tells the TDS parser to take any <i>Length</i> property of child elements or attributes into account. The default value depends on the setting of the <i>Messaging Standard</i> property (at the message set level) and <i>Data Element Separation</i> properties. <ul style="list-style-type: none"> • Where <i>Data Element Separation</i> is All Elements Delimited and the <i>Messaging Standard</i> is set to TLOG , this property should be set. For all other messaging standards it should not be set. • Where <i>Data Element Separation</i> is Tagged Delimited this property should not be set. • Where <i>Data Element Separation</i> is Tagged Fixed Length, Fixed Length, Fixed Length AL3, or Variable Length Elements Delimited this property will be set and is disabled. • For all other data element separation methods, this property is not set and is disabled. Any other combination will generate a task list warning.
Tag Data Separator	Button and String	Specify the value of a special character or string that separates the Tag from the data. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides <i>Length of Tag</i> . This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).
Length of Tag	Integer	Specify the length of a tag value. When the message is parsed, this allows tags to be extracted from the bit stream if the <i>Tag Data Separator</i> property is not set. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides this value. This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).

Property	Type	Meaning
Length of Encoded Length	Integer	<p>Specifies the number of characters (not bytes) after a tag that are used for the length field. Enter a value from 0 to 2147483647.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length; it is not valid otherwise.</p> <p>The actual number of data characters parsed also depends on the value of the <i>Extra Chars in Encoded Length</i> property.</p>
Extra Chars in Encoded Length	Integer	<p>(Only valid if the <i>Data Element Separation</i> method is set to Tagged Encoded Length.) Specifies the number of extra characters included in the value found in the length field. (For example, the value in the length might include the size of the length field itself as well as the size of the data field, or it might be the total size of the tag, length, and data fields.)</p> <p>Enter a value from 0 to 2147483647. The parser subtracts this number from the number found in the length field to get the number of data characters that follow the length field.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length and the actual number of data characters is less than the value found in the length field.</p>
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Message TDS properties:

Property	Type	Meaning
Message Key	STRING	Specify a unique value that identifies the message in the bit stream. This property is required if the message is embedded within another message. For further information, see Multipart messages.

Simple type TDS properties:

There are no properties to show.

Unique TDS properties:

There are no properties to show.

Wildcard attribute TDS properties:

There are no properties to show.

Wildcard element TDS properties:

There are no properties to show.

Documentation properties for all message set objects

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Message model object properties by object

The following objects have properties that can be viewed or set:

- “Attribute group reference properties”
- “Attribute reference properties” on page 78
- “Complex type properties” on page 119
- “Element reference properties” on page 128
- “Global attribute properties” on page 172
- “Global attribute group properties” on page 199
- “Global element properties” on page 200
- “Global group properties” on page 228
- “Group reference properties” on page 233
- “Key properties” on page 235
- “Keyref properties” on page 236
- “Local attribute properties” on page 236
- “Local element properties” on page 298
- “Local group properties” on page 362
- “Message properties” on page 368
- “Simple type properties” on page 373
- “Unique properties” on page 388
- “Wildcard attribute properties” on page 388
- “Wildcard element properties” on page 389

Attribute group reference properties

An attribute group reference can have the following properties;

- “Attribute group reference logical properties” on page 23
- “Attribute group reference CWF properties” on page 47
- “Attribute group reference XML properties” on page 53
- “Attribute group reference TDS properties” on page 62
- “Documentation properties for all message set objects” on page 20

Attribute group reference logical properties:

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Attribute group reference CWF properties:

There are no properties to show.

Attribute group reference XML properties:

There are no properties to show.

Attribute group reference TDS properties:

There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Attribute reference properties

An attribute reference can have the following properties;

- “Attribute reference logical properties” on page 23
- “Attribute reference CWF properties” on page 47
- “Attribute reference XML properties” on page 53
- “Attribute reference TDS properties” on page 62
- “Documentation properties for all message set objects” on page 20

Attribute reference logical properties:

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Property	Type	Meaning
Usage	Enumerated type	<p>The usage property is used in conjunction with the <i>Value</i> property found in an attribute object. The default for the <i>Usage</i> property is optional.</p> <p>Select from;</p> <ul style="list-style-type: none"> • optional. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute can appear once and can have any value. – Where the <i>Value</i> property is set to default, the attribute can appear once. If it does not appear, its value will be the data that has been entered in the <i>Value</i> property. If it does appear it will be the value given. – Where the <i>Value</i> property is set to fixed, the attribute can appear once. If it does appear, its value must match the data that has been entered in the <i>Value</i> property. If it does not appear its value will be the data that has been entered in the <i>Value</i> property. • prohibited. The attribute must not appear. • required. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute must appear once and can have any value. – Where the <i>Value</i> property is set to fixed, the attribute must appear once and it must match the data that has been entered in the <i>Value</i> property.

Attribute reference CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

<p>Binary types</p> <ul style="list-style-type: none"> - base64Binary - hexBinary 	<p>Boolean types</p> <ul style="list-style-type: none"> - boolean 	<p>DateTime types</p> <ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<p>Decimal types</p> <ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
<p>Float types</p> <ul style="list-style-type: none"> - double - float 	<p>Integer types</p> <ul style="list-style-type: none"> - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort 	<p>Interval types</p> <ul style="list-style-type: none"> - duration¹ 	<p>String types</p> <ul style="list-style-type: none"> - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken

Note:

1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.

CWF properties for attribute reference and local attribute binary types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Binary schema types: base64Binary, hexBinary

Physical representation

Property	Type	Meaning
Length Count	Button and Integer	<p>If you have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1.</p> <p>The maximum value that you can specify is 2147483647.</p> <p>The default value is empty (not set).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none">• 1 Bytes. This is the default value.• 2 Bytes• 4 Bytes• 8 Bytes• 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute boolean types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Boolean schema types: boolean

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none">• 1 Bytes. This is the default value.• 2 Bytes• 4 Bytes• 8 Bytes• 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>

Property	Type	Meaning
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute dateTime types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Fixed Length String. The element's length is determined by other length properties below. • Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. • Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. • Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. • Packed Decimal. The dateTime is coded as a Packed Decimal number. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. • Binary. The dateTime is encoded as a binary sequence of bytes. If you select this option, the range of symbols that you can specify for the Format String property is less than the range of symbols you can specify if you select a string option (see "DateTime formats" on page 527 for details). • Time Seconds. This value supports C time_t and Java Date and Time objects. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. • Time Milliseconds. This value supports C time_t and Java Date and Time objects. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. <p>The default value is fixed length string.</p>

Property	Type	Meaning
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see “DateTime defaults by logical type” on page 531.</p> <p>If you set the <i>Physical Type</i> to Binary, the template is restricted to those components defined in “DateTime as STRING data” on page 527. If you set the <i>Physical Type</i> to Packed Decimal, Time Seconds, or Time Milliseconds, the template is restricted to those components that represent numbers. In these cases, you must update this <i>DateTime Format</i> property.</p> <p>See “DateTime formats” on page 527 for details of date and time formats.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String, Packed Decimal, or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1 for all three physical types.</p> <p>The maximum value that you can specify is 256 for Fixed Length String, 10 for Packed Decimal, and 2147483647 for Binary.</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message’s CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message’s CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Property	Type	Meaning
Signed	Boolean	If you have set the <i>Physical Type</i> property to Packed Decimal, Time Seconds, or Time Milliseconds, select (the default) or unselect <i>Signed</i> . If you have selected another value for <i>Physical Type</i> , this property is invalid.
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. Use this option when the value you have set for <i>Encoding Null Value</i> to specify a null date is not a dateTime value, or does not conform to the standard dateTime format yyyy-MM-dd 'T'HH:mm:ss. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>If you set the <i>Encoding Null</i> property to NULLPadFill, this property is disabled (grayed out).</p> <p>If you set the <i>Encoding Null</i> property to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralValue, you can enter any value that is the same length as the field.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralFill, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

CWF properties for attribute reference and local attribute decimal types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	Select one of the following from the drop-down list: <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	Enter the number of bytes to specify the element length: <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. • If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i> .
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a decimal element containing 1234 with a Virtual Decimal value of 3 is 1.234. This is equivalent to 'V' or 'P' in a COBOL picture clause. There is no C equivalent</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute float types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Float schema types: double, float

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. Float. This equates to the data type FLOAT or DOUBLE in C or the COMP-1 or COMP-2 data type in COBOL. This is the default value. Packed Decimal. This equates to the COMP-3 data type in COBOL. External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> Elements that have <i>Physical Type</i> set to Integer, Packed Decimal, and Float are represented in the appropriate WebSphere MQ Encoding value. Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> If you set the <i>Physical Type</i> to Float, select a value from the drop-down list. The default value is 8. If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select or deselect (unsigned, the default) this property. If you have set <i>Physical Type</i> to Float, this is selected. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a float element containing 1234 with a Virtual Decimal value of 3 is 1.234.</p> <p>This is not applicable if you have set <i>Physical Type</i> to Float.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute integer types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. Packed Decimal. This equates to the COMP-3 data type in COBOL. External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. If you have set <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 6. If you have set <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 11.
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute string types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Fixed Length String. The element's length is determined by other length properties below. Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. <p>The default is Fixed Length String.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 0 (zero), the maximum value that you can specify is 2147483647</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Property	Type	Meaning
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • <code>NULLPadFill</code>. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • <code>NULLLiteralValue</code>. The <i>Encoding Null Value</i> is directly substituted as if it is a string. • <code>NULLLiteralFill</code>. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	STRING	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. If specified, its length must be equal to the length of the string element, with the exception of <code>NULLLiteralFill</code>.</p> <p>The default value is empty (not set).</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <code>c</code> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <code>YY</code> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <code>YY</code> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <code>xxxx</code> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Attribute reference XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types <ul style="list-style-type: none">- base64Binary- hexBinary	Boolean types <ul style="list-style-type: none">- boolean	DateTime types <ul style="list-style-type: none">- date- dateTime- gDay- gMonth- gMonthDay- gYear- gYearMonth- time	Decimal types <ul style="list-style-type: none">- decimal- integer- negativeInteger- nonNegativeInteger- nonPositiveInteger- positiveInteger
Float types <ul style="list-style-type: none">- double- float	Integer types <ul style="list-style-type: none">- byte- int- long- short- unsignedByte- unsignedInt- unsignedLong- unsignedShort	Interval types <ul style="list-style-type: none">- duration¹	String types <ul style="list-style-type: none">- anyURI- ENTITIES- ENTITY- ID- IDREF- IDREFS- language- Name- NCName- NMTOKEN- NMTOKENS- normalizedString- NOTATION- QName- stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

XML properties for attribute reference, element reference, local attribute, local element binary types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Binary schema types: base64Binary, hexBinary

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML properties for attribute reference, element reference, local attribute, local element boolean types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Boolean schema types: boolean

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element dateTime types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of dateTime formats.</p>

XML properties for attribute reference, element reference, local attribute, local element decimal types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element float types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Float schema types: double, float

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element integer types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Integer schema types: `byte`, `int`, `long`, `short`, `unsignedByte`, `unsignedInt`, `unsignedLong`, `unsignedShort`

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element string types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Attribute reference TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

TDS properties for attribute reference and element reference binary types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Binary schema types: base64Binary, hexBinary

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference boolean types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Boolean schema types: boolean

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference dateTime types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference decimal types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference float types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Float schema types: double, float

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference integer types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference

- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference string types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Complex type properties

A complex type can have the following properties;

- “Complex type logical properties” on page 24
- “Complex type CWF properties” on page 48
- “Complex type XML properties” on page 54
- “Complex type TDS properties” on page 63
- “Documentation properties for all message set objects” on page 20

Complex type logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>XML</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Base Type	Enumerated type	You can use this property to select a type (simple or complex) that is used as the starting point to define a new complex type that is derived by restriction or extension.
Derived By	Enumerated type	<p>If this property is active, select from;</p> <ul style="list-style-type: none"> • restriction. If a complex type is derived by restriction, the content model of the complex type is a subset of the base type. • extension. If the complex type is derived by extension, the content model of the complex type is the content model of the base type plus the content model specified in the type derivation. <p>Derivation by list or union is not supported.</p>

Content

The table below shows the valid settings for *Composition* and *Content Validation*. These properties are actually located on the group which defines the content of this type. They can only be edited if the *Local group* button is selected. If the *Global group* button is selected, these properties are taken from the global group identified by the *Group name* field.

Valid children in a complex type that depend on both *Composition* and *Content Validation* are shown in “Content Validation properties for complex types” on page 27.

Property	Type	Meaning
Local Group	Button	You should select this if the content of your complex type is a local group.

Property	Type	Meaning
Composition	Enumerated type	<p>The property describes how the message tree is structured and is used in combination with the property <i>Content Validation</i>.</p> <p><i>Composition</i> determines, for example, if the elements within the tree can appear in any order, or if the order is predefined.</p> <p>If you set this property to Ordered Set or Sequence, the order of elements in the input message when the message is parsed, and the order in the logical tree when the output message is constructed by the parser, is important. If the order is not correct, the parser might generate an error, or might produce unexpected results. Therefore you must take care to include ESQL SET statements in the correct order when you create a message in a Compute node.</p> <p>Select from:</p> <ul style="list-style-type: none"> • Empty • sequence. If you select this option, you can only define children that are simple types, complex types, elements, or groups. These children, if present, must appear in the specified order. They can repeat and can be duplicated. • choice. If you select this option, you can define children that are simple types, complex types, or elements. Only one of the defined children of the complex type can be present, but repeating children are allowed. Use this option if you want to model C unions and COBOL REDEFINES in a Custom Wire Format, or an XML DTD element that uses choice in an XML Wire Format, or some industry standard tagged/delimited messages (for example SWIFT) use this format. • all. The elements in an all group can appear in any order. Each element can appear once, or not at all. An all group can only contain elements - groups are not allowed. An all group can only be used at the top level of a complex type - it cannot be a member of another group within a type. • unorderedSet. If you select this option, you can define only elements as children. The elements can repeat but cannot be duplicated. Child elements can appear in any order. • orderedSet. If you select this option, you can define only elements as children. These elements, if present, must appear in the specified order, and they can repeat but cannot be duplicated. This is the default value for new complex types. • message. If you select this option, you can define only messages as children. They can repeat, but they cannot be duplicated. Like Choice, only one of the defined children can be present. If the complex type includes more than one message, the bit stream contains the exact length of the embedded message, and is not padded to the length of the longest. Use this option to model multipart messages, which are used in some industry standards, for example, SWIFT. For more information, see the section on multipart messages in Multipart messages.

Property	Type	Meaning
Content Validation	Enumerated type	<p><i>Content Validation</i> controls how the broker responds to undeclared content and specifies where the objects that are included within the complex type are defined, if at all. It is used in combination with the <i>Composition</i> property.</p> <p>Options:</p> <ul style="list-style-type: none"> • Closed. The complex type can only contain the child elements that you have added to it. • Open Defined. The complex type can contain any valid element defined within the message set. • Open. The complex type can contain any valid element, not just those that you have added to this complex type. <p>See “Combinations of Composition and Content Validation” on page 123 for further details of these options.</p>
Group Reference	Button	You should select this if the content of your complex type is a reference to a group object
Group Name	Enumerated type	The <i>Group Name</i> is the name of the group that this complex type is referring to. The groups available to be referenced can be selected from the drop down list.
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>
Mixed	Check box	Select this where the complex type has mixed content and contains character data alongside sub-elements.

Substitution settings

Property	Type	Meaning
Final	Multiple selection enumerated type	<p>The final attribute on a complex type controls whether other types may be derived from it. Valid values are extension/restriction/all. You can select from one or more of the following:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit type substitution by elements whose types are restrictions of the head element’s type. • extension. Prohibit type substitution by elements whose types are extensions of the head element’s type. • all. Prohibit substitution by any method. <p>To select more than one, you will need to type the selection into the property field.</p>

Property	Type	Meaning
Block	Multiple selection enumerated type	<p>The block attribute on a complex type restricts the types of substitutions which are allowed for elements based on that type. In the WebSphere Business Integration Message Broker its effect is the same as if the block attribute were copied from the complex type onto every element based on the complex type. You can select from one or more of the following:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit type substitution by elements whose types are restrictions of the head element's type. • extension. Prohibit type substitution by elements whose types are extensions of the head element's type. • all. Prohibit substitution by any method. <p>To select more than one, you will need to type the selection into the property field.</p>
Abstract	Check box	If selected, no elements based on this type can appear in the message.

Content Validation properties for complex types:

Content Validation specifies where the objects that are included within a complex type or group are defined, if at all. It is used in combination with *Composition*.

The first table below shows the valid settings for *Content Validation* if *Composition* is set to Message, and the second table shows the valid settings for *Content Validation* if *Composition* is not set to Message.

Content Validation options if Composition is set to Message

Option	Meaning
Open	When a message is parsed, this complex type or group can contain any message, not just those that you have defined in this message set. You can use this option for sparse messages (see Predefined and self-defining elements and messages for a definition of sparse messages).
Closed	When a message is parsed, this complex type or group can only contain the messages that are members of this complex type or group. This is always the case for messages represented in CWF format.
Open Defined	When a message is parsed, this complex type or group can contain any message defined within the message set.

Content Validation options if Composition is not set to Message

Option	Meaning
Open	When a message is parsed, this complex type or group can contain any elements and not just those that you have defined in this message set (see Predefined and self-defining elements and messages for a definition of sparse messages).
Closed	When a message is parsed, this complex type or group can only contain the elements that are members of this complex type or group.
Open Defined	When a message is parsed, this complex type or group can contain any element that you have defined within the message set.

Combinations of Composition and Content Validation:

The table below shows how the choices that you make are affected by the settings you choose for the Composition and Content Validation properties.

Valid children in complex types dependent on Composition and Content Validation

Composition	Content Validation	Valid children
Empty	Closed	None
Empty	Open	None
Empty	Open Defined	None
Sequence	Open	Elements, group references, embedded simple types
Sequence	Closed	Elements, group references, embedded simple types
Sequence	Open Defined	Elements, group references, embedded simple types
Choice	Closed	Elements, group references, embedded simple types
All	Closed	All children are valid
All	Open	All children are valid
All	Open Defined	All children are valid
Unordered Set	Open	Elements
Unordered Set	Closed	Elements
Unordered Set	Open Defined	Elements
Ordered Set	Open	Elements
Ordered Set	Closed	Elements
Ordered Set	Open Defined	Elements
Message	Open	Messages
Message	Closed	Messages
Message	Open Defined	Messages

Valid combinations of repeat and duplicate elements in complex types:

The table below defines the valid combinations of repeated and duplicate elements within a complex type, dependent on the Composition property value.

- A repeated element is an element that is included once within the complex type, and is defined with the property Min Occurs set to greater than 1. Repeated elements are therefore always contiguous and are always specified in the form A[n].
- A duplicate element is an element included more than once anywhere within the complex type. Duplicate elements do not have to be contiguous.

Repeated and duplicate elements in a complex type

Elements in type	Example	Unordered Set	Ordered Set	Sequence
No repeats, no duplicates	A, B, C	Yes	Yes	Yes
Repeated element (contiguous)	A[n], B, C	Yes	Yes	Yes
Duplicate element A (contiguous)	A, A, B, C	No	No	Yes

Elements in type	Example	Unordered Set	Ordered Set	Sequence
Duplicate element A (non-contiguous)	A, B, C, A	No	No	Yes

Complex type CWF properties:

| There are no properties to show.

Complex type XML properties:

| There are no properties to show.

Complex type TDS properties:
Field Identification

If the complex type is based on a global group, the TDS properties listed below will actually be located on the global group. If this is the case, any changes to these properties will be applied to the global group, and will affect all references to the group (including any other complex types which are based on it).

Property	Type	Meaning
Data Element Separation	Enumerated Type	<p>Specify the method used to separate the data elements within the type. Select one of the following values:</p> <ul style="list-style-type: none"> • Tagged Delimited. This value indicates that all elements within the complex type are identified by a tag, and separated by the value specified in the optional <i>Delimiter</i> property (if specified). You must set the <i>Tag</i> property for all child elements of simple type, and you may set the <i>Delimiter</i> property to a non-empty value. See “Global element TDS properties” on page 67. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Fixed Length. This value indicates that each element is identified by a tag, and the data has a fixed length. There are no delimiters. You must set the <i>Tag</i> property for each of the child elements of this complex type, and each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Encoded Length. This value indicates that all elements within the complex type are separated by a tag, and a length field follows each tag. There are no delimiters. The tag can be fixed length, as set by <i>Length of Tag</i>, or variable length delimited by the <i>Tag Data Separator</i>. You must also set <i>Length Of Encoded Length</i> so that the parser knows the size of the length field, and set <i>Extra Chars in Encoded Length</i> to tell the parser how many to subtract from the value in <i>Length Of Encoded Length</i> to get the actual length of data that follows the length field. <p>This method provides a more flexible way of handling ACORD AL3 standard messages than Fixed Length AL3, by allowing different parts of the messages to be at different versions of the ACORD AL3 standard.</p> <ul style="list-style-type: none"> • All Elements Delimited. This value indicates that all elements within the complex type are separated by a delimiter. You must set the value in the <i>Delimiter</i> property. • Variable Length Elements Delimited. This value indicates that some of the elements within the complex type might be of variable length: if they are, they must be delimited by the value specified in the <i>Delimiter</i> property. • Use Data Pattern. This value indicates that the parser determines the elements by matching the data with the regular expression set in the element or type member <i>Data Pattern</i> property. See “Message definition file properties” on page 20. • Fixed Length. This value indicates that all elements within the complex type are fixed length. The next data element is accessed by adding the value of the <i>Length</i> property to the offset (see “Global element TDS properties” on page 67). If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length. Each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. • Fixed Length AL3. This value has a similar meaning to the separation type Fixed Length, but also indicates to the parser that a number of predefined rules with regard to missing optional elements, encoded lengths, and versioning must be applied. If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length AL3, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length AL3. • Undefined. This value is set automatically if you set the <i>Type Composition</i> property of a complex type to Message, and you cannot change it to any other value. <p>Do not set the <i>Type Composition</i> property to Empty, Choice, Unordered Set, Ordered Set, Sequence, or Simple Unordered Set. If you do, you will be unable to check in the type.</p>
Group Indicator	String	Specify the value of a special character or string that precedes the data belonging to a group or complex type within the bit stream.

Property	Type	Meaning
Group Terminator	String	Specify the value of a special character or string that terminates data belonging to a group or a complex type within the bit stream.
Delimiter	String	Specify the value of a special character or string that specifies the delimiter used between data elements. This property applies only to the delimited <i>Data Element Separation</i> methods (Tagged Delimited, All Elements Delimited, and Variable Elements Delimited).
Suppress Absent Element Delimiters	Enumerated type	Use this property to select if you want delimiters to be suppressed for elements that are missing within a message. Select from: <ul style="list-style-type: none"> • End Of Type. Use this option to suppress the delimiter when an element is missing. For example, if the model has been defined to have up to 3 elements and only 2 are present, the last delimiter can be omitted from the message. • Never. Use this option to ensure that even if optional elements are not present, all delimiters will be written out. This option should be used when the delimiter used to delimit parent and child objects is the same. For example, if an optional child element is missing, message processing applications could not tell where the child elements in a message ended and the next parent element started if the delimiters are all the same.
Observe Element Length	Check box	Applicable when <i>Data Element Separation</i> is All Elements Delimited and tells the TDS parser to take any <i>Length</i> property of child elements or attributes into account. The default value depends on the setting of the <i>Messaging Standard</i> property (at the message set level) and <i>Data Element Separation</i> properties. <ul style="list-style-type: none"> • Where <i>Data Element Separation</i> is All Elements Delimited and the <i>Messaging Standard</i> is set to TL0G, this property should be set. For all other messaging standards it should not be set. • Where <i>Data Element Separation</i> is Tagged Delimited this property should not be set. • Where <i>Data Element Separation</i> is Tagged Fixed Length, Fixed Length, Fixed Length AL3, or Variable Length Elements Delimited this property will be set and is disabled. • For all other data element separation methods, this property is not set and is disabled. Any other combination will generate a task list warning.
Tag Data Separator	Button and String	Specify the value of a special character or string that separates the Tag from the data. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides <i>Length of Tag</i> . This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).
Length of Tag	Integer	Specify the length of a tag value. When the message is parsed, this allows tags to be extracted from the bit stream if the <i>Tag Data Separator</i> property is not set. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides this value. This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).

Property	Type	Meaning
Length of Encoded Length	Integer	<p>Specifies the number of characters (not bytes) after a tag that are used for the length field. Enter a value from 0 to 2147483647.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length; it is not valid otherwise.</p> <p>The actual number of data characters parsed also depends on the value of the <i>Extra Chars in Encoded Length</i> property.</p>
Extra Chars in Encoded Length	Integer	<p>(Only valid if the <i>Data Element Separation</i> method is set to Tagged Encoded Length.) Specifies the number of extra characters included in the value found in the length field. (For example, the value in the length might include the size of the length field itself as well as the size of the data field, or it might be the total size of the tag, length, and data fields.)</p> <p>Enter a value from 0 to 2147483647. The parser subtracts this number from the number found in the length field to get the number of data characters that follow the length field.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length and the actual number of data characters is less than the value found in the length field.</p>

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Element reference properties

An element reference can have the following properties:

- “Element reference logical properties” on page 28
- “Element reference CWF properties” on page 48
- “Element reference XML properties” on page 54
- “Element reference TDS properties” on page 66
- “Documentation properties for all message set objects” on page 20

Element reference logical properties:

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Occurrences

Property	Type	Meaning
Min Occurs	Integer	Specify the minimum number of times that the object can repeat. The default is 1. If the value is set to 0, then the object is optional. If a value is set, it must be less than or equal to the value in <i>Max Occurs</i> .
Max Occurs	Integer	Specify the maximum number of times that the object can repeat. The default is 1. If this property is not set, then the object can not occur more than once. It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.

The Min Occurs and Max Occurs properties are used in conjunction with an elements Value properties. The table below summarizes how an element reference can be constrained.

Min Occurs	Max Occurs	Fixed	Default	Notes
1	1			The element must appear once and can have any value.
1	1	Delta		The element must appear once and it must match the data that has been entered in the <i>Value</i> property. In this example the element must contain the text Delta.
2	-1	Delta		The element must appear twice or more and it must match the data that has been entered in the <i>Value</i> property. In this example there will be at least two elements that must contain the text Delta.
0	1			The element is optional and can appear once and have any value.
0	1	Delta		The element is optional and can appear once. If it does appear, its value must match the data that has been entered in the <i>Value</i> property. If it does not appear its value will be the data that has been entered in the <i>Value</i> property.
0	1		Delta	The element is optional and can appear once. If it does not appear, its value will be the data that has been entered in the <i>Value</i> property. If it does appear it must be the value given in the element.
0	2		Delta	The element is optional and can appear once, twice or not at all. If the element does not appear it is not provided. If the element appears and it is empty, it set to the data held in the <i>Value</i> property, else it is the value given in the element.
0	0			The element is prohibited and must not appear.

Element reference CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

CWF properties for element reference and local element binary types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Binary schema types: base64Binary, hexBinary

Physical representation

Property	Type	Meaning
Length Count	Button and Integer	If you have set <i>Length Type</i> to Count, enter the number of length units for the element. The minimum value that you can specify is 1. The maximum value that you can specify is 2147483647. The default value is empty (not set).

Property	Type	Meaning
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>

Property	Type	Meaning
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for element reference and local element boolean types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Boolean schema types: boolean

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for element reference and local element dateTime types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Fixed Length String. The element's length is determined by other length properties below. • Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. • Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. • Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. • Packed Decimal. The dateTime is coded as a Packed Decimal number. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. • Binary. The dateTime is encoded as a binary sequence of bytes. If you select this option, the range of symbols that you can specify for the Format String property is less than the range of symbols you can specify if you select a string option (see "DateTime formats" on page 527 for details). • Time Seconds. This value supports C time_t and Java Date and Time objects. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. • Time Milliseconds. This value supports C time_t and Java Date and Time objects. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. <p>The default value is fixed length string.</p>

Property	Type	Meaning
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see “DateTime defaults by logical type” on page 531.</p> <p>If you set the <i>Physical Type</i> to Binary, the template is restricted to those components defined in “DateTime as STRING data” on page 527. If you set the <i>Physical Type</i> to Packed Decimal, Time Seconds, or Time Milliseconds, the template is restricted to those components that represent numbers. In these cases, you must update this <i>DateTime Format</i> property.</p> <p>See “DateTime formats” on page 527 for details of date and time formats.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String, Packed Decimal, or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1 for all three physical types.</p> <p>The maximum value that you can specify is 256 for Fixed Length String, 10 for Packed Decimal, and 2147483647 for Binary.</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message’s CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message’s CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Property	Type	Meaning
Signed	Boolean	If you have set the <i>Physical Type</i> property to Packed Decimal, Time Seconds, or Time Milliseconds, select (the default) or unselect <i>Signed</i> . If you have selected another value for <i>Physical Type</i> , this property is invalid.
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. Use this option when the value you have set for <i>Encoding Null Value</i> to specify a null date is not a dateTime value, or does not conform to the standard dateTime format yyyy-MM-dd 'T'HH:mm:ss. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>If you set the <i>Encoding Null</i> property to NULLPadFill, this property is disabled (grayed out).</p> <p>If you set the <i>Encoding Null</i> property to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralValue, you can enter any value that is the same length as the field.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralFill, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for element reference and local element decimal types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. Packed Decimal. This equates to the COMP-3 data type in COBOL. External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a decimal element containing 1234 with a Virtual Decimal value of 3 is 1.234. This is equivalent to 'V' or 'P' in a COBOL picture clause. There is no C equivalent</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for element reference and local element float types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Float schema types: double, float

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none">• Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL.• Float. This equates to the data type FLOAT or DOUBLE in C or the COMP-1 or COMP-2 data type in COBOL. This is the default value.• Packed Decimal. This equates to the COMP-3 data type in COBOL.• External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none">• Elements that have <i>Physical Type</i> set to Integer, Packed Decimal, and Float are represented in the appropriate WebSphere MQ Encoding value.• Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none">• If you set the <i>Physical Type</i> to Float, select a value from the drop-down list. The default value is 8.• If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list.• If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10.• If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select or deselect (unsigned, the default) this property. If you have set <i>Physical Type</i> to Float, this is selected. This property is used in conjunction with <i>Sign Orientation</i> .
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a float element containing 1234 with a Virtual Decimal value of 3 is 1.234.</p> <p>This is not applicable if you have set <i>Physical Type</i> to Float.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for element reference and local element integer types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you have set <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 6. • If you have set <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 11.
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>

Property	Type	Meaning
Signed	Boolean	Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i> .
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for element reference and local element string types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none">• Fixed Length String. The element's length is determined by other length properties below.• Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units.• Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding.• Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. <p>The default is Fixed Length String.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 0 (zero), the maximum value that you can specify is 2147483647</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	STRING	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. If specified, its length must be equal to the length of the string element, with the exception of <code>NULLLiteralFill</code>.</p> <p>The default value is empty (not set).</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <code>c</code> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <code>YY</code> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <code>YY</code> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <code>xxxx</code> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>

Property	Type	Meaning
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Element reference XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

<p>Binary types</p> <ul style="list-style-type: none"> - base64Binary - hexBinary 	<p>Boolean types</p> <ul style="list-style-type: none"> - boolean 	<p>DateTime types</p> <ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<p>Decimal types</p> <ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
<p>Float types</p> <ul style="list-style-type: none"> - double - float 	<p>Integer types</p> <ul style="list-style-type: none"> - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort 	<p>Interval types</p> <ul style="list-style-type: none"> - duration¹ 	<p>String types</p> <ul style="list-style-type: none"> - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
<p>Note:</p> <p>1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.</p>			

XML properties for attribute reference, element reference, local attribute, local element binary types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Binary schema types: base64Binary, hexBinary

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML properties for attribute reference, element reference, local attribute, local element boolean types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Boolean schema types: boolean

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element dateTime types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of dateTime formats.</p>

XML properties for attribute reference, element reference, local attribute, local element decimal types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element float types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Float schema types: double, float

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element integer types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Integer schema types: `byte`, `int`, `long`, `short`, `unsignedByte`, `unsignedInt`, `unsignedLong`, `unsignedShort`

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element string types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Element reference TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

TDS properties for attribute reference and element reference binary types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Binary schema types: base64Binary, hexBinary

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference boolean types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Boolean schema types: boolean

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference dateTime types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference decimal types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference float types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- Float schema types: double, float

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference integer types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference

- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

TDS properties for attribute reference and element reference string types:

The TDS Format properties described here apply to:

- Objects: Attribute Reference, Element Reference
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Global attribute properties

A global attribute can have the following properties;

- “Global attribute logical properties” on page 29
- “Global attribute CWF properties” on page 49
- “Global attribute XML properties” on page 55
- “Global attribute TDS properties” on page 67
- “Documentation properties for all message set objects” on page 20

Global attribute logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • <code>int</code> • <code>string</code> • <code>boolean</code> • <code>hexBinary</code> • <code>dateTime</code> • <code>date</code> • <code>time</code> • <code>decimal</code> • <code>float</code> • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Value properties

The *Value* properties are used in conjunction with the *Usage* property in an Attribute Reference or a Local Attribute.

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>

Global attribute CWF properties:

There are no properties to show.

Global attribute XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

XML properties for global attribute and global element binary types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Binary schema types: base64Binary, hexBinary

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML properties for global attribute and global element boolean types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Boolean schema types: boolean

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element dateTime types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of dateTime formats.</p>

XML properties for global attribute and global element decimal types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element float types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Float schema types: double, float

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element integer types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element string types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

Global attribute TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

TDS properties for global attribute and global element binary types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Binary schema types: base64Binary, hexBinary

Field identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Physical representation

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p> <p>Regardless of the value of the <i>Data Element Separator</i> property, either the <i>Length</i> or <i>Length Reference</i> property must be set.</p>

TDS properties for global attribute and global element boolean types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Boolean schema types: boolean

Field identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.</p>

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.

TDS properties for global attribute and global element dateTime types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default DateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of date and time formats.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for global attribute and global element decimal types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>

Property	Type	Meaning
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present.
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for global attribute and global element float types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Float schema types: double, float

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>

Property	Type	Meaning
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present. • Exponential Notation: data is written out to the bit stream as a signed value having the format [sign1]a.bbb[e[sign2]ccc] where: <ul style="list-style-type: none"> – [sign1] is the value of Negative Sign if the value is negative – a is a single decimal digit – bbb is one or more decimal digits – [sign2] is the value of Negative Sign if the exponent is negative – ccc is exactly three decimal digits (the exponent) <p>[sign1] and [sign2] are absent if the value and exponent, respectively, are positive.</p> <p>For example, the value -123.456 is represented as -1.23456e002 and the value 0.00012 is represented as 1.2e-004 in the output bit stream (assuming the value of <i>Negative Sign</i> is "-" and <i>Sign Orientation</i> is Leading).</p> <p>The value -0.00012 is represented as 1.2*e*004 if <i>Negative Sign</i> is "*" and <i>Sign Orientation</i> is Trailing.</p>
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	<p>Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.</p>

Property	Type	Meaning
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • <code>NULLPadFill</code>. This is only valid for fixed length objects. This is the default value. • <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • <code>NULLLiteralValue</code>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <code>dateTime</code> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <code>dateTime</code> object to <code>NULLLogicalValue</code>, you must set this property to an ISO8601 <code>dateTime</code> format. These formats are described in "Date Time as STRING data" on page 527. For example, specify a value conforming to <code>yyyy-MM-dd'T'HH:mm:ss</code> such as <code>1970-12-01</code>.</p>

TDS properties for global attribute and global element integer types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Integer schema types: `byte`, `int`, `long`, `short`, `unsignedByte`, `unsignedInt`, `unsignedLong`, `unsignedShort`

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Numeric representation

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid for fixed length objects. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for global attribute and global element string types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.
Interpret Element Value	Enumerated Type	<p>Specify if values stored within this object must be interpreted as having significance for the parser and, if so, the type of interpretation that must occur. This interpretation is generally standard-specific and is therefore hard coded.</p> <p>The possible values for this property are:</p> <ul style="list-style-type: none"> • None (the default value) • EDIFACT Service String • X12 Service String • Message Key • EDIFACT Syntax Level ID

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> • Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Global attribute group properties

A global attribute group can have the following properties;

- “Global attribute group logical properties” on page 31
- “Global attribute group CWF properties” on page 49
- “Global attribute group XML properties” on page 56
- “Global attribute group TDS properties” on page 67
- “Documentation properties for all message set objects” on page 20

Global attribute group logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>XML</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>

Global attribute group CWF properties:

| There are no properties to show.

Global attribute group XML properties:

| There are no properties to show.

Global attribute group TDS properties:

| There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

| 1. Key, Keyref, and Unique objects do not have documentation properties.

Global element properties

A global element can have the following properties;

- “Global element logical properties” on page 32
- “Global element CWF properties” on page 49
- “Global element XML properties” on page 56
- “Global element TDS properties” on page 67
- “Documentation properties for all message set objects” on page 20

Global element logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • int • string • boolean • hexBinary • dateTime • date • time • decimal • float • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Value

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>
Nilable	Check box	Select this if you want the element to be able to be defined as null. This is distinct from being empty where there is no data in the element.

Substitution settings

Substitution Groups provide a means by which one element may be substituted for another in a message. The element which can be substituted is called the 'head' element, and the substitution group is the list of elements that may be used in its place. An element can be in at most one substitution group.

Property	Type	Meaning
Final	Enumerated type	<p>You use this property to limit the set of elements which may belong to its substitution group.</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type. • extension. Prohibit element substitution by elements whose types are extensions of the head element's type. • all. Prohibit substitution by any method.

Property	Type	Meaning
Block	Enumerated type	<p>You use this property to limit the set of elements which may be substituted for this element in a message.</p> <p>Select from:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type • extension. Prohibit element substitution by elements whose types are extensions of the head element's type • substitution. Prohibit element substitution by members of the element's substitution group. • #all. Prohibit substitution by any method.
Substitution Group	Enumerated type	Use this property to specify the name of a 'head' element. Setting this property indicates that this element is a member of the substitution group for the 'head' element.
Abstract	Check box	Select this if you do not want the element to appear in the message, but require one of the members of its substitution group to appear in its place.

Global element CWF properties:

There are no properties to show.

Global element XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

XML properties for global attribute and global element binary types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Binary schema types: base64Binary, hexBinary

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML properties for global attribute and global element boolean types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Boolean schema types: boolean

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element dateTime types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of dateTime formats.</p>

XML properties for global attribute and global element decimal types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element float types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Float schema types: double, float

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element integer types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

XML properties for global attribute and global element string types:

The XML Wire Format properties described here apply to:

- Objects: Global Attribute, Global Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>

Global element TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - base64Binary - hexBinary	Boolean types - boolean	DateTime types - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time	Decimal types - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

TDS properties for global attribute and global element binary types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Binary schema types: base64Binary, hexBinary

Field identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Physical representation

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p> <p>Regardless of the value of the <i>Data Element Separator</i> property, either the <i>Length</i> or <i>Length Reference</i> property must be set.</p>

TDS properties for global attribute and global element boolean types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Boolean schema types: boolean

Field identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.

TDS properties for global attribute and global element dateTime types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default DateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of date and time formats.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for global attribute and global element decimal types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>

Property	Type	Meaning
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present.
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for global attribute and global element float types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Float schema types: double, float

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>

Property	Type	Meaning
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present. • Exponential Notation: data is written out to the bit stream as a signed value having the format [sign1]a.bbb[e[sign2]ccc] where: <ul style="list-style-type: none"> – [sign1] is the value of Negative Sign if the value is negative – a is a single decimal digit – bbb is one or more decimal digits – [sign2] is the value of Negative Sign if the exponent is negative – ccc is exactly three decimal digits (the exponent) <p>[sign1] and [sign2] are absent if the value and exponent, respectively, are positive.</p> <p>For example, the value -123.456 is represented as -1.23456e002 and the value 0.00012 is represented as 1.2e-004 in the output bit stream (assuming the value of <i>Negative Sign</i> is "-" and <i>Sign Orientation</i> is Leading).</p> <p>The value -0.00012 is represented as 1.2*e*004 if <i>Negative Sign</i> is "*" and <i>Sign Orientation</i> is Trailing.</p>
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	<p>Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.</p>

Property	Type	Meaning
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • <code>NULLPadFill</code>. This is only valid for fixed length objects. This is the default value. • <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • <code>NULLLiteralValue</code>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <code>dateTime</code> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <code>dateTime</code> object to <code>NULLLogicalValue</code>, you must set this property to an ISO8601 <code>dateTime</code> format. These formats are described in "Date Time as STRING data" on page 527. For example, specify a value conforming to <code>yyyy-MM-dd'T'HH:mm:ss</code> such as <code>1970-12-01</code>.</p>

TDS properties for global attribute and global element integer types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- Integer schema types: `byte`, `int`, `long`, `short`, `unsignedByte`, `unsignedInt`, `unsignedLong`, `unsignedShort`

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Numeric representation

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid for fixed length objects. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for global attribute and global element string types:

The TDS Format properties described here apply to:

- Objects: Global Attribute, Global Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.</p>
Interpret Element Value	Enumerated Type	<p>Specify if values stored within this object must be interpreted as having significance for the parser and, if so, the type of interpretation that must occur. This interpretation is generally standard-specific and is therefore hard coded.</p> <p>The possible values for this property are:</p> <ul style="list-style-type: none"> • None (the default value) • EDIFACT Service String • X12 Service String • Message Key • EDIFACT Syntax Level ID

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> • Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Global group properties

A global element can have the following properties;

- “Global group logical properties” on page 34
- “Global group CWF properties” on page 49
- “Global group XML properties” on page 56
- “Global group TDS properties” on page 68
- “Documentation properties for all message set objects” on page 20

Global group logical properties:

Valid children in a global group that depend on both **Composition** and **Content Validation** are shown in “Content Validation properties for complex types” on page 27.

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Composition	Enumerated type	<p>The following only applies to the element content of a complex type and has no effect on the attribute content of a complex type. Select from:</p> <ul style="list-style-type: none"> • Empty • sequence. If you select this option, you can define children that are elements or groups. These children, if present, must appear in the specified order. They can repeat and can be duplicated. • choice. If you select this option, you can define children that are elements or groups. Only one of the defined children of the complex type can be present, but repeating children are allowed. <p>When a complex type whose <i>Composition</i> is set to Choice is present in an output message, the bit stream contains the number of bytes represented by the longest item (padded with 0x00).</p> <p>Use this option if you want to model C unions and COBOL REDEFINES in a Custom Wire Format, or an XML DTD element that uses choice in an XML Wire Format, or some industry standard tagged/delimited messages (for example SWIFT) use this format.</p> <ul style="list-style-type: none"> • all. The elements in an all group can appear in any order. Each element can appear once, or not at all. An all group can only contain elements - groups are not allowed. An all group can only be used at the top level of a complex type - it cannot be a member of another group within a type. • unorderedSet. If you select this option, you can define only elements as children. The elements can repeat but cannot be duplicated. Child elements can appear in any order. • orderedSet. If you select this option, you can define only elements as children. These elements, if present, must appear in the specified order, and they can repeat but cannot be duplicated. This is the default value for new complex types. • message. If you select this option, you can define only messages as children. They can repeat, but they cannot be duplicated. Like choice, only one of the defined children can be present. <p>If the complex type includes more than one message, the bit stream contains the exact length of the embedded message, and is not padded to the length of the longest.</p> <p>Use this option to model multipart messages, which are used in some industry standards, for example, SWIFT. For more information, see the section on multipart messages in Multipart messages.</p>

Property	Type	Meaning
Content Validation	Enumerated type	<p><i>Content Validation</i> controls how the broker responds to undeclared content and specifies where the objects that are included within the complex type are defined, if at all. It is used in combination with the <i>Composition</i> property.</p> <p>Options:</p> <ul style="list-style-type: none"> • Closed. The complex type can only contain the child elements that you have added to it. • Open Defined. The complex type can contain any valid element defined within the message set. • Open. The complex type can contain any valid element, not just those that you have added to this complex type. <p>See “Combinations of Composition and Content Validation” on page 123 for further details of these options.</p>

Global group CWF properties:

| There are no properties to show.

Global group XML properties:

| There are no properties to show.

Global group TDS properties:

Field Identification

Property	Type	Meaning
Data Element Separation	Enumerated Type	<p>Specify the method used to separate the data elements within the type. Select one of the following values:</p> <ul style="list-style-type: none"> • Tagged Delimited. This value indicates that all elements within the complex type are identified by a tag, and separated by the value specified in the optional <i>Delimiter</i> property (if specified). You must set the <i>Tag</i> property for all child elements of simple type, and you may set the <i>Delimiter</i> property to a non-empty value. See “Global element TDS properties” on page 67. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Fixed Length. This value indicates that each element is identified by a tag, and the data has a fixed length. There are no delimiters. You must set the <i>Tag</i> property for each of the child elements of this complex type, and each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Encoded Length. This value indicates that all elements within the complex type are separated by a tag, and a length field follows each tag. There are no delimiters. The tag can be fixed length, as set by <i>Length of Tag</i>, or variable length delimited by the <i>Tag Data Separator</i>. You must also set <i>Length Of Encoded Length</i> so that the parser knows the size of the length field, and set <i>Extra Chars in Encoded Length</i> to tell the parser how many to subtract from the value in <i>Length Of Encoded Length</i> to get the actual length of data that follows the length field. <p>This method provides a more flexible way of handling ACORD AL3 standard messages than Fixed Length AL3, by allowing different parts of the messages to be at different versions of the ACORD AL3 standard.</p> <ul style="list-style-type: none"> • All Elements Delimited. This value indicates that all elements within the complex type are separated by a delimiter. You must set the value in the <i>Delimiter</i> property. • Variable Length Elements Delimited. This value indicates that some of the elements within the complex type might be of variable length: if they are, they must be delimited by the value specified in the <i>Delimiter</i> property. • Use Data Pattern. This value indicates that the parser determines the elements by matching the data with the regular expression set in the element or type member <i>Data Pattern</i> property. See “Message definition file properties” on page 20. • Fixed Length. This value indicates that all elements within the complex type are fixed length. The next data element is accessed by adding the value of the <i>Length</i> property to the offset (see “Global element TDS properties” on page 67). If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length. Each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. • Fixed Length AL3. This value has a similar meaning to the separation type Fixed Length, but also indicates to the parser that a number of predefined rules with regard to missing optional elements, encoded lengths, and versioning must be applied. If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length AL3, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length AL3. • Undefined. This value is set automatically if you set the <i>Type Composition</i> property of a complex type to Message, and you cannot change it to any other value. <p>Do not set the <i>Type Composition</i> property to Empty, Choice, Unordered Set, Ordered Set, Sequence, or Simple Unordered Set. If you do, you will be unable to check in the type.</p>

Property	Type	Meaning
Group Indicator	String	Specify the value of a special character or string that precedes the data belonging to a group or complex type within the bit stream.
Group Terminator	String	Specify the value of a special character or string that terminates data belonging to a group or a complex type within the bit stream.
Delimiter	String	Specify the value of a special character or string that specifies the delimiter used between data elements. This property applies only to the delimited <i>Data Element Separation</i> methods (Tagged Delimited, All Elements Delimited, and Variable Elements Delimited).
Suppress Absent Element Delimiters	Enumerated type	Use this property to select if you want delimiters to be suppressed for elements that are missing within a message. Select from: <ul style="list-style-type: none"> • End Of Type. Use this option to suppress the delimiter when an element is missing. For example, if the model has been defined to have up to 3 elements and only 2 are present, the last delimiter can be omitted from the message. • Never. Use this option to ensure that even if optional elements are not present, all delimiters will be written out. This option should be used when the delimiter used to delimit parent and child objects is the same. For example, if an optional child element is missing, message processing applications could not tell where the child elements in a message ended and the next parent element started if the delimiters are all the same.
Observe Element Length	Check box	Applicable when <i>Data Element Separation</i> is All Elements Delimited and tells the TDS parser to take any <i>Length</i> property of child elements or attributes into account. The default value depends on the setting of the <i>Messaging Standard</i> property (at the message set level) and <i>Data Element Separation</i> properties. <ul style="list-style-type: none"> • Where <i>Data Element Separation</i> is All Elements Delimited and the <i>Messaging Standard</i> is set to TLOG , this property should be set. For all other messaging standards it should not be set. • Where <i>Data Element Separation</i> is Tagged Delimited this property should not be set. • Where <i>Data Element Separation</i> is Tagged Fixed Length, Fixed Length, Fixed Length AL3, or Variable Length Elements Delimited this property will be set and is disabled. • For all other data element separation methods, this property is not set and is disabled. Any other combination will generate a task list warning.
Tag Data Separator	Button and String	Specify the value of a special character or string that separates the Tag from the data. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides <i>Length of Tag</i> . This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).
Length of Tag	Integer	Specify the length of a tag value. When the message is parsed, this allows tags to be extracted from the bit stream if the <i>Tag Data Separator</i> property is not set. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides this value. This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).

Property	Type	Meaning
Length of Encoded Length	Integer	<p>Specifies the number of characters (not bytes) after a tag that are used for the length field. Enter a value from 0 to 2147483647.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length; it is not valid otherwise.</p> <p>The actual number of data characters parsed also depends on the value of the <i>Extra Chars in Encoded Length</i> property.</p>
Extra Chars in Encoded Length	Integer	<p>(Only valid if the <i>Data Element Separation</i> method is set to Tagged Encoded Length.) Specifies the number of extra characters included in the value found in the length field. (For example, the value in the length might include the size of the length field itself as well as the size of the data field, or it might be the total size of the tag, length, and data fields.)</p> <p>Enter a value from 0 to 2147483647. The parser subtracts this number from the number found in the length field to get the number of data characters that follow the length field.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length and the actual number of data characters is less than the value found in the length field.</p>

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Group reference properties

A group reference can have the following properties;

- “Group reference logical properties” on page 36
- “Group reference CWF properties” on page 49
- “Group reference XML properties” on page 56
- “Group reference TDS properties” on page 71
- “Documentation properties for all message set objects” on page 20

Group reference logical properties:

Property	Type	Meaning
Reference Name	Enumerated type	The <i>Reference Name</i> is the name of the object that this object is referring to. The objects available to reference can be selected from the drop down list.

Occurrence properties

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Group reference CWF properties: Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Group reference XML properties:

There are no properties to show.

Group reference TDS properties: Field Identification

Property	Type	Meaning
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Key properties

A key can have the following properties;

- “Key logical properties” on page 38
- “Key CWF properties” on page 50
- “Key XML properties” on page 56
- “Key TDS properties” on page 71
- “Documentation properties for all message set objects” on page 20

Key logical properties:

There are no properties to show.

Key CWF properties:

There are no properties to show.

Key XML properties:

There are no properties to show.

Key TDS properties:

There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Keyref properties

A keyref can have the following properties;

- “Keyref logical properties” on page 38
- “Keyref CWF properties” on page 50
- “Keyref XML properties” on page 57
- “Keyref TDS properties” on page 71
- “Documentation properties for all message set objects” on page 20

Keyref logical properties:

There are no properties to show.

Keyref CWF properties:

There are no properties to show.

Keyref XML properties:

There are no properties to show.

Keyref TDS properties:

There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Local attribute properties

A local attribute can have the following properties;

- “Local attribute logical properties” on page 36
- “Local attribute CWF properties” on page 50
- “Local attribute XML properties” on page 57
- “Local attribute TDS properties” on page 71

- “Documentation properties for all message set objects” on page 20

Local attribute logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • int • string • boolean • hexBinary • dateTime • date • time • decimal • float • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Value properties

The *Value* properties are used in conjunction with the *Usage* property in an Attribute Reference or a Local Attribute.

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>

Usage properties

Property	Type	Meaning
Usage	Enumerated type	<p>The usage property is used in conjunction with the <i>Value</i> property found in an attribute object. The default for the <i>Usage</i> property is optional.</p> <p>Select from;</p> <ul style="list-style-type: none"> • optional. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute can appear once and can have any value. – Where the <i>Value</i> property is set to default, the attribute can appear once. If it does not appear, its value will be the data that has been entered in the <i>Value</i> property. If it does appear it will be the value given. – Where the <i>Value</i> property is set to fixed, the attribute can appear once. If it does appear, its value must match the data that has been entered in the <i>Value</i> property. If it does not appear its value will be the data that has been entered in the <i>Value</i> property. • prohibited. The attribute must not appear. • required. <ul style="list-style-type: none"> – Where the <i>Value</i> property is set to default and no data has been entered in the <i>Value</i> property, the attribute must appear once and can have any value. – Where the <i>Value</i> property is set to fixed, the attribute must appear once and it must match the data that has been entered in the <i>Value</i> property.

Local attribute CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

CWF properties for attribute reference and local attribute binary types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Binary schema types: base64Binary, hexBinary

Physical representation

Property	Type	Meaning
Length Count	Button and Integer	If you have set <i>Length Type</i> to Count, enter the number of length units for the element. The minimum value that you can specify is 1. The maximum value that you can specify is 2147483647. The default value is empty (not set).
Length Reference	Button and Enumerated Type	If you have selected the length to be defined by <i>Length Reference</i> , select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute boolean types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Boolean schema types: boolean

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none">• 1 Bytes. This is the default value.• 2 Bytes• 4 Bytes• 8 Bytes• 16 Bytes
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

CWF properties for attribute reference and local attribute dateTime types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Fixed Length String. The element's length is determined by other length properties below. Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. Packed Decimal. The date<code>Time</code> is coded as a Packed Decimal number. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. Binary. The date<code>Time</code> is encoded as a binary sequence of bytes. If you select this option, the range of symbols that you can specify for the Format String property is less than the range of symbols you can specify if you select a string option (see "Date<code>Time</code> formats" on page 527 for details). Time Seconds. This value supports C <code>time_t</code> and Java Date and Time objects. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. Time Milliseconds. This value supports C <code>time_t</code> and Java Date and Time objects. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. <p>The default value is fixed length string.</p>
Date <code>Time</code> Format	String	<p>Specify a template for date and time.</p> <p>The default date<code>Time</code> format is dependent on the logical type of the object. For information on the defaults for the date<code>Time</code> format according to the logical type see "Date<code>Time</code> defaults by logical type" on page 531.</p> <p>If you set the <i>Physical Type</i> to Binary, the template is restricted to those components defined in "Date<code>Time</code> as STRING data" on page 527. If you set the <i>Physical Type</i> to Packed Decimal, Time Seconds, or Time Milliseconds, the template is restricted to those components that represent numbers. In these cases, you must update this <i>Date<code>Time</code> Format</i> property.</p> <p>See "Date<code>Time</code> formats" on page 527 for details of date and time formats.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String, Packed Decimal, or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1 for all three physical types.</p> <p>The maximum value that you can specify is 256 for Fixed Length String, 10 for Packed Decimal, and 2147483647 for Binary.</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>If you have set the <i>Physical Type</i> property to Packed Decimal, Time Seconds, or Time Milliseconds, select (the default) or unselect <i>Signed</i>. If you have selected another value for <i>Physical Type</i>, this property is invalid.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. Use this option when the value you have set for <i>Encoding Null Value</i> to specify a null date is not a dateTime value, or does not conform to the standard dateTime format yyyy-MM-dd 'T'HH:mm:ss. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>If you set the <i>Encoding Null</i> property to NULLPadFill, this property is disabled (grayed out).</p> <p>If you set the <i>Encoding Null</i> property to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralValue, you can enter any value that is the same length as the field.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralFill, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

CWF properties for attribute reference and local attribute decimal types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	Select one of the following from the drop-down list: <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	Enter the number of bytes to specify the element length: <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. • If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i> .
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a decimal element containing 1234 with a Virtual Decimal value of 3 is 1.234. This is equivalent to 'V' or 'P' in a COBOL picture clause. There is no C equivalent</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute float types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Float schema types: double, float

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. Float. This equates to the data type FLOAT or DOUBLE in C or the COMP-1 or COMP-2 data type in COBOL. This is the default value. Packed Decimal. This equates to the COMP-3 data type in COBOL. External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> Elements that have <i>Physical Type</i> set to Integer, Packed Decimal, and Float are represented in the appropriate WebSphere MQ Encoding value. Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> If you set the <i>Physical Type</i> to Float, select a value from the drop-down list. The default value is 8. If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select or deselect (unsigned, the default) this property. If you have set <i>Physical Type</i> to Float, this is selected. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a float element containing 1234 with a Virtual Decimal value of 3 is 1.234.</p> <p>This is not applicable if you have set <i>Physical Type</i> to Float.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • <i>NULLPadFill</i>. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • <i>NULLLogicalValue</i>. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • <i>NULLLiteralValue</i>. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • <i>NULLLiteralFill</i>. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute integer types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. Packed Decimal. This equates to the COMP-3 data type in COBOL. External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. If you have set <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 6. If you have set <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 11.
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • <i>NULLPadFill</i>. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • <i>NULLLogicalValue</i>. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • <i>NULLLiteralValue</i>. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • <i>NULLLiteralFill</i>. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

CWF properties for attribute reference and local attribute string types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Fixed Length String. The element's length is determined by other length properties below. Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. <p>The default is Fixed Length String.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 0 (zero), the maximum value that you can specify is 2147483647</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Property	Type	Meaning
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • <code>NULLPadFill</code>. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • <code>NULLLiteralValue</code>. The <i>Encoding Null Value</i> is directly substituted as if it is a string. • <code>NULLLiteralFill</code>. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	STRING	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. If specified, its length must be equal to the length of the string element, with the exception of <code>NULLLiteralFill</code>.</p> <p>The default value is empty (not set).</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <code>c</code> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <code>YY</code> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <code>YY</code> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <code>xxxx</code> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Local attribute XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types <ul style="list-style-type: none">- base64Binary- hexBinary	Boolean types <ul style="list-style-type: none">- boolean	DateTime types <ul style="list-style-type: none">- date- dateTime- gDay- gMonth- gMonthDay- gYear- gYearMonth- time	Decimal types <ul style="list-style-type: none">- decimal- integer- negativeInteger- nonNegativeInteger- nonPositiveInteger- positiveInteger
Float types <ul style="list-style-type: none">- double- float	Integer types <ul style="list-style-type: none">- byte- int- long- short- unsignedByte- unsignedInt- unsignedLong- unsignedShort	Interval types <ul style="list-style-type: none">- duration¹	String types <ul style="list-style-type: none">- anyURI- ENTITIES- ENTITY- ID- IDREF- IDREFS- language- Name- NCName- NMTOKEN- NMTOKENS- normalizedString- NOTATION- QName- stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

XML properties for attribute reference, element reference, local attribute, local element binary types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Binary schema types: base64Binary, hexBinary

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML properties for attribute reference, element reference, local attribute, local element boolean types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Boolean schema types: boolean

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element dateTime types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of dateTime formats.</p>

XML properties for attribute reference, element reference, local attribute, local element decimal types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element float types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Float schema types: double, float

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element integer types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Integer schema types: `byte`, `int`, `long`, `short`, `unsignedByte`, `unsignedInt`, `unsignedLong`, `unsignedShort`

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element string types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Local attribute TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

TDS properties for local attribute and local element binary types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Binary schema types: base64Binary, hexBinary

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p> <p>Regardless of the value of the <i>Data Element Separator</i> property, either the <i>Length</i> or <i>Length Reference</i> property must be set.</p>
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS properties for local attribute and local element boolean types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Boolean schema types: boolean

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS properties for local attribute and local element dateTime types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default DateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of date and time formats.</p>
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element decimal types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present.
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element float types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Float schema types: double, float

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present. • Exponential Notation: data is written out to the bit stream as a signed value having the format [sign1]a.bbbe[sign2]ccc where: <ul style="list-style-type: none"> – [sign1] is the value of Negative Sign if the value is negative – a is a single decimal digit – bbb is one or more decimal digits – [sign2] is the value of Negative Sign if the exponent is negative – ccc is exactly three decimal digits (the exponent) <p>[sign1] and [sign2] are absent if the value and exponent, respectively, are positive.</p> <p>For example, the value -123.456 is represented as -1.23456e002 and the value 0.00012 is represented as 1.2e-004 in the output bit stream (assuming the value of <i>Negative Sign</i> is "-" and <i>Sign Orientation</i> is Leading).</p> <p>The value -0.00012 is represented as 1.2*e*004 if <i>Negative Sign</i> is "*" and <i>Sign Orientation</i> is Trailing.</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to <i>None</i>, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property <i>Leading</i>, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to <i>Trailing</i>, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • <i>NULLPadFill</i>. This is only valid for fixed length objects. This is the default value. • <i>NULLLogicalValue</i>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • <i>NULLLiteralValue</i>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <i>dateTime</i> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <i>dateTime</i> object to <i>NULLLogicalValue</i>, you must set this property to an ISO8601 <i>dateTime</i> format. These formats are described in "Date/Time as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element integer types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>

Property	Type	Meaning
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid for fixed length objects. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element string types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.
Interpret Element Value	Enumerated Type	<p>Specify if values stored within this object must be interpreted as having significance for the parser and, if so, the type of interpretation that must occur. This interpretation is generally standard-specific and is therefore hard coded.</p> <p>The possible values for this property are:</p> <ul style="list-style-type: none"> • None (the default value) • EDIFACT Service String • X12 Service String • Message Key • EDIFACT Syntax Level ID
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> • Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none">• <code>NULLPadFill</code>. This is only valid for fixed length objects. This is the default value.• <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field.• <code>NULLLiteralValue</code>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <code>dateTime</code> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <code>dateTime</code> object to <code>NULLLogicalValue</code>, you must set this property to an ISO8601 <code>dateTime</code> format. These formats are described in “Date Time as STRING data” on page 527. For example, specify a value conforming to <code>yyyy-MM-dd'T'HH:mm:ss</code> such as <code>1970-12-01</code>.</p>

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Local element properties

A local element can have the following properties;

- “Local element logical properties” on page 39
- “Local element CWF properties” on page 51
- “Local element XML properties” on page 57
- “Local element TDS properties” on page 72
- “Documentation properties for all message set objects” on page 20

Local element logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <code>xml</code> or any variant (for example <code>Xml</code>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Type	Enumerated type	<p>The Type property constrains the type of data that can be present in the object.</p> <p>There are a limited number of types available directly from the drop down selector. These are;</p> <ul style="list-style-type: none"> • <code>int</code> • <code>string</code> • <code>boolean</code> • <code>hexBinary</code> • <code>dateTime</code> • <code>date</code> • <code>time</code> • <code>decimal</code> • <code>float</code> • (More...) • (New Simple Type) • (New Complex Type) <p>If you select (More...), this starts the Type Selection wizard. From this wizard you can select any of the available types.</p> <p>If you select (New Simple Type), this starts the New Simple Type wizard which allows you to create an Anonymous simple type which will be based on an existing type. This can be created locally or globally.</p> <p>If you select (New Complex Type), this starts the New Complex Type wizard which allows you to create an Anonymous complex type which can be derived from an existing base type. This can be created locally or globally.</p> <p>For further information about these types, and examples of their use see the XML Schema Part 0: Primer which can be found on the World Wide Web Consortium (W3C) Web site.</p>

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Occurrences

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Value

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>
Nilable	Check box	Select this if you want the element to be able to be defined as null. This is distinct from being empty where there is no data in the element.

Substitution settings

Substitution Groups provide a means by which one element may be substituted for another in a message. The element which can be substituted is called the 'head' element, and the substitution group is the list of elements that may be used in its place. An element can be in at most one substitution group.

Property	Type	Meaning
Final	Enumerated type	<p>You use this property to limit the set of elements which may belong to its substitution group.</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type. • extension. Prohibit element substitution by elements whose types are extensions of the head element's type. • all. Prohibit substitution by any method.
Block	Enumerated type	<p>You use this property to limit the set of elements which may be substituted for this element in a message.</p> <p>Select from:</p> <ul style="list-style-type: none"> • Empty • restriction. Prohibit element substitution by elements whose types are restrictions of the head element's type • extension. Prohibit element substitution by elements whose types are extensions of the head element's type • substitution. Prohibit element substitution by members of the element's substitution group. • #all. Prohibit substitution by any method.
Substitution Group	Enumerated type	Use this property to specify the name of a 'head' element. Setting this property indicates that this element is a member of the substitution group for the 'head' element.
Abstract	Check box	Select this if you do not want the element to appear in the message, but require one of the members of its substitution group to appear in its place.

Local element CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

CWF properties for element reference and local element binary types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Binary schema types: base64Binary, hexBinary

Physical representation

Property	Type	Meaning
Length Count	Button and Integer	If you have set <i>Length Type</i> to Count, enter the number of length units for the element. The minimum value that you can specify is 1. The maximum value that you can specify is 2147483647. The default value is empty (not set).
Length Reference	Button and Enumerated Type	If you have selected the length to be defined by <i>Length Reference</i> , select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for element reference and local element boolean types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Boolean schema types: boolean

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.

Property	Type	Meaning
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for element reference and local element dateTime types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Fixed Length String. The element's length is determined by other length properties below. • Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. • Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. • Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. • Packed Decimal. The dateTime is coded as a Packed Decimal number. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. • Binary. The dateTime is encoded as a binary sequence of bytes. If you select this option, the range of symbols that you can specify for the Format String property is less than the range of symbols you can specify if you select a string option (see "DateTime formats" on page 527 for details). • Time Seconds. This value supports C time_t and Java Date and Time objects. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. • Time Milliseconds. This value supports C time_t and Java Date and Time objects. It is valid only if the <i>DateTime Format</i> property represents numeric-only data. <p>The default value is fixed length string.</p>

Property	Type	Meaning
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see “DateTime defaults by logical type” on page 531.</p> <p>If you set the <i>Physical Type</i> to Binary, the template is restricted to those components defined in “DateTime as STRING data” on page 527. If you set the <i>Physical Type</i> to Packed Decimal, Time Seconds, or Time Milliseconds, the template is restricted to those components that represent numbers. In these cases, you must update this <i>DateTime Format</i> property.</p> <p>See “DateTime formats” on page 527 for details of date and time formats.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String, Packed Decimal, or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1 for all three physical types.</p> <p>The maximum value that you can specify is 256 for Fixed Length String, 10 for Packed Decimal, and 2147483647 for Binary.</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message’s CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message’s CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Property	Type	Meaning
Signed	Boolean	If you have set the <i>Physical Type</i> property to Packed Decimal, Time Seconds, or Time Milliseconds, select (the default) or unselect <i>Signed</i> . If you have selected another value for <i>Physical Type</i> , this property is invalid.
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. Use this option when the value you have set for <i>Encoding Null Value</i> to specify a null date is not a dateTime value, or does not conform to the standard dateTime format yyyy-MM-dd 'T'HH:mm:ss. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>If you set the <i>Encoding Null</i> property to NULLPadFill, this property is disabled (grayed out).</p> <p>If you set the <i>Encoding Null</i> property to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralValue, you can enter any value that is the same length as the field.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralFill, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for element reference and local element decimal types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. Packed Decimal. This equates to the COMP-3 data type in COBOL. External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Bytes. Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a decimal element containing 1234 with a Virtual Decimal value of 3 is 1.234. This is equivalent to 'V' or 'P' in a COBOL picture clause. There is no C equivalent</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for element reference and local element float types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Float schema types: double, float

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none">• Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL.• Float. This equates to the data type FLOAT or DOUBLE in C or the COMP-1 or COMP-2 data type in COBOL. This is the default value.• Packed Decimal. This equates to the COMP-3 data type in COBOL.• External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none">• Elements that have <i>Physical Type</i> set to Integer, Packed Decimal, and Float are represented in the appropriate WebSphere MQ Encoding value.• Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none">• If you set the <i>Physical Type</i> to Float, select a value from the drop-down list. The default value is 8.• If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list.• If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10.• If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select or deselect (unsigned, the default) this property. If you have set <i>Physical Type</i> to Float, this is selected. This property is used in conjunction with <i>Sign Orientation</i> .
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a float element containing 1234 with a Virtual Decimal value of 3 is 1.234.</p> <p>This is not applicable if you have set <i>Physical Type</i> to Float.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for element reference and local element integer types:

The Custom Wire Format properties described here apply to:

- Objects: Element Reference, Local Element
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you have set <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 6. • If you have set <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 11.
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>

Property	Type	Meaning
Signed	Boolean	Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i> .
Signed EBCDIC Custom	Boolean	<p>If the <i>Physical Type</i> is set to External Decimal and the <i>Signed EBCDIC Custom</i> property is set, this will indicate that the Sign EBCDIC Custom representation is to be used within an ASCII environment. If this checkbox is not set (the default) then the Sign ASCII representation will be used.</p> <p>The setting of the <i>Sign EBCDIC Custom</i> checkbox is only appropriate if the <i>Sign Orientation</i> property is set to Leading or Trailing (indicating that the element/attribute has an embedded sign representation).</p> <p>The checkbox will be disabled if the element/attribute is unsigned (for example, where the <i>Signed</i> checkbox is not set).</p>
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for element reference and local element string types:

The Custom Wire Format properties described here apply to:

- Objects: Attribute Reference, Local Attribute
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none">• Fixed Length String. The element's length is determined by other length properties below.• Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units.• Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding.• Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. <p>The default is Fixed Length String.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 0 (zero), the maximum value that you can specify is 2147483647</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	STRING	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. If specified, its length must be equal to the length of the string element, with the exception of <code>NULLLiteralFill</code>.</p> <p>The default value is empty (not set).</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <code>c</code> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <code>YY</code> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <code>YY</code> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <code>xxxx</code> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>

Property	Type	Meaning
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Local element XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

<p>Binary types</p> <ul style="list-style-type: none"> - base64Binary - hexBinary 	<p>Boolean types</p> <ul style="list-style-type: none"> - boolean 	<p>DateTime types</p> <ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<p>Decimal types</p> <ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger
<p>Float types</p> <ul style="list-style-type: none"> - double - float 	<p>Integer types</p> <ul style="list-style-type: none"> - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort 	<p>Interval types</p> <ul style="list-style-type: none"> - duration¹ 	<p>String types</p> <ul style="list-style-type: none"> - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
<p>Note:</p> <p>1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.</p>			

XML properties for attribute reference, element reference, local attribute, local element binary types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Binary schema types: base64Binary, hexBinary

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML properties for attribute reference, element reference, local attribute, local element boolean types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Boolean schema types: boolean

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element dateTime types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of dateTime formats.</p>

XML properties for attribute reference, element reference, local attribute, local element decimal types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element float types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Float schema types: double, float

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element integer types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- Integer schema types: `byte`, `int`, `long`, `short`, `unsignedByte`, `unsignedInt`, `unsignedLong`, `unsignedShort`

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML properties for attribute reference, element reference, local attribute, local element string types:

The XML Wire Format properties described here apply to:

- Objects: Attribute Reference, Element Reference, Local Attribute, Local Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Local element TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
<ul style="list-style-type: none"> - base64Binary - hexBinary 	<ul style="list-style-type: none"> - boolean 	<ul style="list-style-type: none"> - date - dateTime - gDay - gMonth - gMonthDay - gYear - gYearMonth - time 	<ul style="list-style-type: none"> - decimal - integer - negativeInteger - nonNegativeInteger - nonPositiveInteger - positiveInteger

Float types - double - float	Integer types - byte - int - long - short - unsignedByte - unsignedInt - unsignedLong - unsignedShort	Interval types - duration ¹	String types - anyURI - ENTITIES - ENTITY - ID - IDREF - IDREFS - language - Name - NCName - NMTOKEN - NMTOKENS - normalizedString - NOTATION - QName - stringtoken
Note: 1. duration: The physical format properties for simple type duration are the same as the physical properties of the String logical types.			

TDS properties for local attribute and local element binary types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Binary schema types: base64Binary, hexBinary

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p> <p>Regardless of the value of the <i>Data Element Separator</i> property, either the <i>Length</i> or <i>Length Reference</i> property must be set.</p>
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS properties for local attribute and local element boolean types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Boolean schema types: boolean

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS properties for local attribute and local element dateTime types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default DateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of date and time formats.</p>
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element decimal types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present.
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element float types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Float schema types: double, float

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> Not Applicable Left Justify Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present. • Exponential Notation: data is written out to the bit stream as a signed value having the format [sign1]a.bbbe[sign2]ccc where: <ul style="list-style-type: none"> – [sign1] is the value of Negative Sign if the value is negative – a is a single decimal digit – bbb is one or more decimal digits – [sign2] is the value of Negative Sign if the exponent is negative – ccc is exactly three decimal digits (the exponent) <p>[sign1] and [sign2] are absent if the value and exponent, respectively, are positive.</p> <p>For example, the value -123.456 is represented as -1.23456e002 and the value 0.00012 is represented as 1.2e-004 in the output bit stream (assuming the value of <i>Negative Sign</i> is "-" and <i>Sign Orientation</i> is Leading).</p> <p>The value -0.00012 is represented as 1.2*e*004 if <i>Negative Sign</i> is "*" and <i>Sign Orientation</i> is Trailing.</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to <i>None</i>, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property <i>Leading</i>, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to <i>Trailing</i>, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • <i>NULLPadFill</i>. This is only valid for fixed length objects. This is the default value. • <i>NULLLogicalValue</i>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • <i>NULLLiteralValue</i>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <i>dateTime</i> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <i>dateTime</i> object to <i>NULLLogicalValue</i>, you must set this property to an ISO8601 <i>dateTime</i> format. These formats are described in "Date/Time as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element integer types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>

Property	Type	Meaning
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid for fixed length objects. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS properties for local attribute and local element string types:

The TDS Format properties described here apply to:

- Objects: Local Attribute, Local Element
- String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.
Interpret Element Value	Enumerated Type	<p>Specify if values stored within this object must be interpreted as having significance for the parser and, if so, the type of interpretation that must occur. This interpretation is generally standard-specific and is therefore hard coded.</p> <p>The possible values for this property are:</p> <ul style="list-style-type: none"> • None (the default value) • EDIFACT Service String • X12 Service String • Message Key • EDIFACT Syntax Level ID
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Physical representation

Property	Type	Meaning
Physical Type	Enumerated type	<p>The <i>Physical Type</i> can be set to Characters and Messaging Standard Alternate. This property tells the TDS parser whether the data in the message is the normal TDS character format, or is another alternate form that has a specific messaging standard such as TLOG. The available values and the default value depend on both <i>Messaging Standard</i> and the logical type.</p> <ul style="list-style-type: none"> • Where the logical type of the object is of a dateTime, float, or integer type, this property is disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to other than TLOG, the <i>Physical Type</i> property will be disabled. • Where the <i>Messaging Standard</i> property (at the message set level) is set to TLOG and the logical type of the object is set to a boolean, decimal or string type, the <i>Physical Type</i> property can be set to Characters or Messaging Standard Alternate.

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none">• <code>NULLPadFill</code>. This is only valid for fixed length objects. This is the default value.• <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field.• <code>NULLLiteralValue</code>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <code>dateTime</code> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <code>dateTime</code> object to <code>NULLLogicalValue</code>, you must set this property to an ISO8601 <code>dateTime</code> format. These formats are described in “Date Time as STRING data” on page 527. For example, specify a value conforming to <code>yyyy-MM-dd'T'HH:mm:ss</code> such as <code>1970-12-01</code>.</p>

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Local group properties

A local group can have the following properties;

- “Local group logical properties” on page 41
- “Local group CWF properties” on page 52
- “Local group XML properties” on page 58
- “Local group TDS properties” on page 73
- “Documentation properties for all message set objects” on page 20

Local group logical properties:

Valid children in a local group that depend on both **Composition** and **Content Validation** are shown in “Content Validation properties for complex types” on page 27.

Property	Type	Meaning
Composition	Enumerated type	<p>The following only applies to the element content of a complex type and has no effect on the attribute content of a complex type. Select from:</p> <ul style="list-style-type: none"> • Empty • sequence. If you select this option, you can define children that are elements or groups. These children, if present, must appear in the specified order. They can repeat and can be duplicated. • choice. If you select this option, you can define children that are elements or groups. Only one of the defined children of the complex type can be present, but repeating children are allowed. <p>When a complex type whose <i>Composition</i> is set to Choice is present in an output message, the bit stream contains the number of bytes represented by the longest item (padded with 0x00).</p> <p>Use this option if you want to model C unions and COBOL REDEFINES in a Custom Wire Format, or an XML DTD element that uses choice in an XML Wire Format, or some industry standard tagged/delimited messages (for example SWIFT) use this format.</p> <ul style="list-style-type: none"> • all. The elements in an all group can appear in any order. Each element can appear once, or not at all. An all group can only contain elements - groups are not allowed. An all group can only be used at the top level of a complex type - it cannot be a member of another group within a type. • unorderedSet. If you select this option, you can define only elements as children. The elements can repeat but cannot be duplicated. Child elements can appear in any order. • orderedSet. If you select this option, you can define only elements as children. These elements, if present, must appear in the specified order, and they can repeat but cannot be duplicated. This is the default value for new complex types. • message. If you select this option, you can define only messages as children. They can repeat, but they cannot be duplicated. Like choice, only one of the defined children can be present. <p>If the complex type includes more than one message, the bit stream contains the exact length of the embedded message, and is not padded to the length of the longest.</p> <p>Use this option to model multipart messages, which are used in some industry standards, for example, SWIFT. For more information, see the section on multipart messages in Multipart messages.</p>
Content Validation	Enumerated type	<p><i>Content Validation</i> controls how the broker responds to undeclared content and specifies where the objects that are included within the complex type are defined, if at all. It is used in combination with the <i>Composition</i> property.</p> <p>Options:</p> <ul style="list-style-type: none"> • Closed. The complex type can only contain the child elements that you have added to it. • Open Defined. The complex type can contain any valid element defined within the message set. • Open. The complex type can contain any valid element, not just those that you have added to this complex type. <p>See “Combinations of Composition and Content Validation” on page 123 for further details of these options.</p>

Occurrences

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Local group CWF properties: Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Local group XML properties:

There are no properties to show.

Local group TDS properties:

Field Identification

Property	Type	Meaning
Data Element Separation	Enumerated Type	<p>Specify the method used to separate the data elements within the type. Select one of the following values:</p> <ul style="list-style-type: none"> • Tagged Delimited. This value indicates that all elements within the complex type are identified by a tag, and separated by the value specified in the optional <i>Delimiter</i> property (if specified). You must set the <i>Tag</i> property for all child elements of simple type, and you may set the <i>Delimiter</i> property to a non-empty value. See “Global element TDS properties” on page 67. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Fixed Length. This value indicates that each element is identified by a tag, and the data has a fixed length. There are no delimiters. You must set the <i>Tag</i> property for each of the child elements of this complex type, and each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. You must also set the <i>Tag Data Separator</i> or the <i>Length of Tag</i> properties. • Tagged Encoded Length. This value indicates that all elements within the complex type are separated by a tag, and a length field follows each tag. There are no delimiters. The tag can be fixed length, as set by <i>Length of Tag</i>, or variable length delimited by the <i>Tag Data Separator</i>. You must also set <i>Length Of Encoded Length</i> so that the parser knows the size of the length field, and set <i>Extra Chars in Encoded Length</i> to tell the parser how many to subtract from the value in <i>Length Of Encoded Length</i> to get the actual length of data that follows the length field. <p>This method provides a more flexible way of handling ACORD AL3 standard messages than Fixed Length AL3, by allowing different parts of the messages to be at different versions of the ACORD AL3 standard.</p> <ul style="list-style-type: none"> • All Elements Delimited. This value indicates that all elements within the complex type are separated by a delimiter. You must set the value in the <i>Delimiter</i> property. • Variable Length Elements Delimited. This value indicates that some of the elements within the complex type might be of variable length: if they are, they must be delimited by the value specified in the <i>Delimiter</i> property. • Use Data Pattern. This value indicates that the parser determines the elements by matching the data with the regular expression set in the element or type member <i>Data Pattern</i> property. See “Message definition file properties” on page 20. • Fixed Length. This value indicates that all elements within the complex type are fixed length. The next data element is accessed by adding the value of the <i>Length</i> property to the offset (see “Global element TDS properties” on page 67). If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length. Each child element must have a <i>Length</i> or <i>Length Reference</i> property assigned to it. • Fixed Length AL3. This value has a similar meaning to the separation type Fixed Length, but also indicates to the parser that a number of predefined rules with regard to missing optional elements, encoded lengths, and versioning must be applied. If you set the <i>Data Element Separation</i> property of a complex type to Fixed Length AL3, you must also set the <i>Data Element Separation</i> property of all complex children of this type to Fixed Length AL3. • Undefined. This value is set automatically if you set the <i>Type Composition</i> property of a complex type to Message, and you cannot change it to any other value. <p>Do not set the <i>Type Composition</i> property to Empty, Choice, Unordered Set, Ordered Set, Sequence, or Simple Unordered Set. If you do, you will be unable to check in the type.</p>

Property	Type	Meaning
Group Indicator	String	Specify the value of a special character or string that precedes the data belonging to a group or complex type within the bit stream.
Group Terminator	String	Specify the value of a special character or string that terminates data belonging to a group or a complex type within the bit stream.
Delimiter	String	Specify the value of a special character or string that specifies the delimiter used between data elements. This property applies only to the delimited <i>Data Element Separation</i> methods (Tagged Delimited, All Elements Delimited, and Variable Elements Delimited).
Suppress Absent Element Delimiters	Enumerated type	Use this property to select if you want delimiters to be suppressed for elements that are missing within a message. Select from: <ul style="list-style-type: none"> • End Of Type. Use this option to suppress the delimiter when an element is missing. For example, if the model has been defined to have up to 3 elements and only 2 are present, the last delimiter can be omitted from the message. • Never. Use this option to ensure that even if optional elements are not present, all delimiters will be written out. This option should be used when the delimiter used to delimit parent and child objects is the same. For example, if an optional child element is missing, message processing applications could not tell where the child elements in a message ended and the next parent element started if the delimiters are all the same.
Observe Element Length	Check box	Applicable when <i>Data Element Separation</i> is All Elements Delimited and tells the TDS parser to take any <i>Length</i> property of child elements or attributes into account. The default value depends on the setting of the <i>Messaging Standard</i> property (at the message set level) and <i>Data Element Separation</i> properties. <ul style="list-style-type: none"> • Where <i>Data Element Separation</i> is All Elements Delimited and the <i>Messaging Standard</i> is set to TLOG , this property should be set. For all other messaging standards it should not be set. • Where <i>Data Element Separation</i> is Tagged Delimited this property should not be set. • Where <i>Data Element Separation</i> is Tagged Fixed Length, Fixed Length, Fixed Length AL3, or Variable Length Elements Delimited this property will be set and is disabled. • For all other data element separation methods, this property is not set and is disabled. Any other combination will generate a task list warning.
Tag Data Separator	Button and String	Specify the value of a special character or string that separates the Tag from the data. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides <i>Length of Tag</i> . This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).
Length of Tag	Integer	Specify the length of a tag value. When the message is parsed, this allows tags to be extracted from the bit stream if the <i>Tag Data Separator</i> property is not set. The <i>Tag Data Separator</i> and <i>Length of Tag</i> properties are mutually exclusive. If you set the property <i>Tag Data Separator</i> , it overrides this value. This property applies only to the tagged <i>Data Element Separation</i> methods (Tagged Delimited, Tagged Fixed Length, and Tagged Encoded Length).

Property	Type	Meaning
Length of Encoded Length	Integer	<p>Specifies the number of characters (not bytes) after a tag that are used for the length field. Enter a value from 0 to 2147483647.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length; it is not valid otherwise.</p> <p>The actual number of data characters parsed also depends on the value of the <i>Extra Chars in Encoded Length</i> property.</p>
Extra Chars in Encoded Length	Integer	<p>(Only valid if the <i>Data Element Separation</i> method is set to Tagged Encoded Length.) Specifies the number of extra characters included in the value found in the length field. (For example, the value in the length might include the size of the length field itself as well as the size of the data field, or it might be the total size of the tag, length, and data fields.)</p> <p>Enter a value from 0 to 2147483647. The parser subtracts this number from the number found in the length field to get the number of data characters that follow the length field.</p> <p>You must set this property if you have set <i>Data Element Separation</i> property to Tagged Encoded Length and the actual number of data characters is less than the value found in the length field.</p>
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Message properties

A message can have the following properties;

- “Message logical properties” on page 43
- “Message CWF properties” on page 52
- “Message XML properties” on page 58
- “Message TDS properties” on page 76
- “Documentation properties for all message set objects” on page 20

Message logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>

Message CWF properties:

There are no properties to show.

Message XML properties:

Namespace schema locations

This property is only active if namespaces have been enabled.

Property	Type	Meaning
Namespace URI	String	<p>A unique string, usually in the form of a URL that identifies the schema for this</p> <p>If namespaces have not been enabled, this property will display <no target namespace>.</p> <p>This property will override the same property at the message set level.</p>
Schema location	String	<p>Enter the location of the schema for the associated namespace name that will be used to validate objects within the namespace.</p>

XML declarations

Property	Type	Meaning
Output Namespace Declaration	Enumerated Type	<p>The <i>Output Namespace Declaration</i> property controls where the namespace declarations will be placed in the output XML document.</p> <p>Select from:</p> <ul style="list-style-type: none"> • At start of document. Declarations for all of the entries in the <i>Namespace schema locations</i> table above will be output as attributes of the message in the output XML document. The disadvantage of this option is that in some cases unnecessary declarations may be output. • As required. Declarations will only be output when required by an element or attribute that is in that namespace. The disadvantage of this option is that the same namespace declaration may need to be output more than once in the output XML document. <p>The default option is At start of document.</p> <p>This property is only active if namespaces are enabled for this message set.</p>

XML document type settings

Property	Type	Meaning
DOCTYPE System ID	String	<p>Specify the System ID for DOCTYPE external DTD subset. It overrides the equivalent message set property setting for this particular message.</p> <p>If the message set property <i>Suppress DOCTYPE</i> is set to Yes, this parameter is ignored and cannot be changed (the field is disabled) .</p> <p>The default value is the value that you specified for the <i>DOCTYPE System ID</i> property for the message set.</p>
DOCTYPE Public ID	String	<p>Specify the Public ID for DOCTYPE external DTD subset. It overrides the equivalent message set property setting for this particular message.</p> <p>If the message set property <i>Suppress DOCTYPE</i> is set to Yes, this parameter is ignored and cannot be changed (the field is disabled) .</p> <p>The default value is the value that you specified for the <i>DOCTYPE Public ID</i> property for the message set.</p>
DOCTYPE Text	String	<p>Enter optional additional text to include within the DOCTYPE. It overrides the message set property for this particular message.</p> <p>If the message set property <i>Suppress DOCTYPE</i> is set to Yes, this parameter is ignored and cannot be changed (the field is disabled) .</p> <p>For more information, see “In-line DTDs and the DOCTYPE text property” on page 12.</p> <p>The default value is the value that you specified for the <i>DOCTYPE Text</i> property for the message set.</p>

Property	Type	Meaning
Root Tag Name*	String	<p>Specify the name of the root tag for a message bit stream XML document. It overrides the message set property set for this message.</p> <p>The default value is the value that you specified for the <i>Root Tag Name</i> property for the message set.</p>

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>

XML Message rendering options:

There are four properties on the XML layer that you can use to affect how the XML messages are rendered. The table below shows examples of the values that you can set for the *Member Render* property. In this table, the member element is referred to as *A*, and has the value value of element. The parent is referred to as *X*.

The effect of rendering options on XML output

To get XML rendered like this:	Set this Member Render property value:	Set these other property values:
<pre><X> <A>value of element </X></pre>	XMLElement (the default)	Member XML Name = A
<pre><X A='value of element' /></pre>	XMLAttribute	Member XML Name = A
<pre><X> <Field id='A'>value of element</Field> </X></pre>	XMLElementAttrID	Member XML Name = Field Member ID Attribute Name = id Member ID Attribute Value = A
<pre><X> </X></pre>	XMLElementAttrVal	Member XML Name = A Member Value Attribute Name = val
<pre><X> <Field id='A' val='value of element' /> </X></pre>	XMLElementAttrIDVal	Member XML Name = Field Member ID Attribute Name = id Member ID Attribute Value = A Member Value Attribute Name = val

Message TDS properties:

Property	Type	Meaning
Message Key	STRING	Specify a unique value that identifies the message in the bit stream. This property is required if the message is embedded within another message. For further information, see Multipart messages.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Simple type properties

A simple type can have the following properties;

- “Simple type logical properties” on page 44
- “Simple type CWF properties” on page 52
- “Simple type XML properties” on page 62
- “Simple type TDS properties” on page 76
- “Documentation properties for all message set objects” on page 20

A simple type can also have “Simple type logical value constraints” on page 44.

Simple type logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none">• - the hyphen• _ the underscore• . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Base Type	Enumerated type	You can use this property to select a base type that is used as the starting point to define a new simple type that is derived by setting additional value constraints.

A simple type can also have “Simple type logical value constraints” on page 44.

Simple type logical value constraints:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types <ul style="list-style-type: none">- base64Binary- hexBinary	Boolean types <ul style="list-style-type: none">- boolean	DateTime types <ul style="list-style-type: none">- date- dateTime- gDay- gMonth- gMonthDay- gYear- gYearMonth- time	Decimal types <ul style="list-style-type: none">- decimal- integer- negativeInteger- nonNegativeInteger- nonPositiveInteger- positiveInteger
Float types <ul style="list-style-type: none">- double- float	Integer types <ul style="list-style-type: none">- byte- int- long- short- unsignedByte- unsignedInt- unsignedLong- unsignedShort	Interval types <ul style="list-style-type: none">- duration	String types <ul style="list-style-type: none">- anyURI- ENTITIES- ENTITY- ID- IDREF- IDREFS- language- Name- NCName- NMTOKEN- NMTOKENS- normalizedString- NOTATION- QName- stringtoken

Logical properties for value constraints for simple type binary types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- Binary schema types: base64Binary, hexBinary

Length constraints

Property	Type	Meaning
Length	Integer	The length property is used to specify the exact length of the simple type in bytes or characters. The value must be greater than 0 and less than 2147483648.
Min	Integer	The Min property is used to specify the minimum length of the simple type in bytes or characters. The value must be greater than 0 and less than 2147483648.

Property	Type	Meaning
Max	Integer	<p>The Max property is used to specify the maximum length of the simple type in bytes or characters.</p> <p>The value must be greater than 0 and less than 2147483648.</p>

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • preserve. If you set the property to preserve, all white space characters including carriage return, line feed and tab are preserved. • replace. If you set the property to replace, all carriage return, line feed and tab characters are replaced with a space character. • collapse. If you set the property to collapse, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Enumerations

Property	Type	Meaning
Enumerations	String	<p><i>Enumerations</i> constrain the values to the list that is specified in this property. For example, you could create a simple type called <i>RainbowColors</i>. You would then add Red, Orange, Yellow, Green, Blue, Indigo, and Violet to the enumerations list.</p> <p>You will need to ensure that you have all variations of the data that you are likely to receive in the message defined in the list. For example, Yellow, yellow, yel, y could be possible variations of a single color.</p> <p>Select Add to add a default enumeration. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see "Using regular expressions to parse data elements" on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Logical properties for value constraints for simple type boolean types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- Boolean schema types: boolean

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • preserve. If you set the property to preserve, all white space characters including carriage return, line feed and tab are preserved. • replace. If you set the property to replace, all carriage return, line feed and tab characters are replaced with a space character. • collapse. If you set the property to collapse, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see “Using regular expressions to parse data elements” on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Logical properties for value constraints for simple type dateTime types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time

Inclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than or equal to.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Min</i> Inclusive Constraint and <i>Min</i> Exclusive Constraint properties together for the same simple type.</p>

Property	Type	Meaning
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than or equal to.</p> <p>When this value is set it cannot be equal to or less than the <i>Min Inclusive</i> Constraint property.</p> <p>You cannot specify both <i>Max Inclusive</i> Constraint and <i>Max Exclusive</i> Constraint properties together for the same simple type.</p>

Exclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max Inclusive</i> Constraint property.</p> <p>You cannot specify both <i>Min Inclusive</i> Constraint and <i>Min Exclusive</i> Constraint properties together for the same simple type.</p>
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than.</p> <p>When this value is set it cannot be equal to or less than the <i>Min Inclusive</i> Constraint property.</p> <p>You cannot specify both <i>Max Inclusive</i> Constraint and <i>Max Exclusive</i> Constraint properties together for the same simple type.</p>

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • <i>preserve</i>. If you set the property to <i>preserve</i>, all white space characters including carriage return, line feed and tab are preserved. • <i>replace</i>. If you set the property to <i>replace</i>, all carriage return, line feed and tab characters are replaced with a space character. • <i>collapse</i>. If you set the property to <i>collapse</i>, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Enumerations

Property	Type	Meaning
Enumerations	String	<p><i>Enumerations</i> constrain the values to the list that is specified in this property. For example, you could create a simple type called <i>RainbowColors</i>. You would then add Red, Orange, Yellow, Green, Blue, Indigo, and Violet to the enumerations list.</p> <p>You will need to ensure that you have all variations of the data that you are likely to receive in the message defined in the list. For example, Yellow, yellow, yel, y could be possible variations of a single color.</p> <p>Select Add to add a default enumeration. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see “Using regular expressions to parse data elements” on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Logical properties for value constraints for simple type decimal types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger

Inclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than or equal to.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Min</i> Inclusive Constraint and <i>Min</i> Exclusive Constraint properties together for the same simple type.</p>

Property	Type	Meaning
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than or equal to.</p> <p>When this value is set it cannot be equal to or less than the <i>Min Inclusive Constraint</i> property.</p> <p>You cannot specify both <i>Max Inclusive Constraint</i> and <i>Max Exclusive Constraint</i> properties together for the same simple type.</p>

Exclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max Inclusive Constraint</i> property.</p> <p>You cannot specify both <i>Min Inclusive Constraint</i> and <i>Min Exclusive Constraint</i> properties together for the same simple type.</p>
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than.</p> <p>When this value is set it cannot be equal to or less than the <i>Min Inclusive Constraint</i> property.</p> <p>You cannot specify both <i>Max Inclusive Constraint</i> and <i>Max Exclusive Constraint</i> properties together for the same simple type.</p>

Property	Type	Meaning
Fraction Digits	Integer	<p>Set this property to limit the number of digits in the fraction part of a numerical value to the number of digits specified in this property.</p> <p>The value must be greater than or equal to 0 and less than 2147483648.</p> <p>The value set for <i>Fraction Digits</i> cannot be greater than the value specified for <i>Total Digits</i>.</p>

Property	Type	Meaning
Total Digits	Integer	<p>Set this property to set the maximum number of digits in a numerical value to the number of digits specified in this property.</p> <p>The value must be greater than or equal to 0 and less than 2147483648.</p> <p>The value set for <i>Total Digits</i> cannot be less than the value specified for <i>Fraction Digits</i>.</p>

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • preserve. If you set the property to preserve, all white space characters including carriage return, line feed and tab are preserved. • replace. If you set the property to replace, all carriage return, line feed and tab characters are replaced with a space character. • collapse. If you set the property to collapse, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Enumerations

Property	Type	Meaning
Enumerations	String	<p><i>Enumerations</i> constrain the values to the list that is specified in this property. For example, you could create a simple type called <i>RainbowColors</i>. You would then add Red, Orange, Yellow, Green, Blue, Indigo, and Violet to the enumerations list.</p> <p>You will need to ensure that you have all variations of the data that you are likely to receive in the message defined in the list. For example, Yellow, yellow, yel, y could be possible variations of a single color.</p> <p>Select Add to add a default enumeration. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see “Using regular expressions to parse data elements” on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Logical properties for value constraints for simple type float types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- Float schema types: double, float

Inclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than or equal to.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Min</i> Inclusive Constraint and <i>Min</i> Exclusive Constraint properties together for the same simple type.</p>
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than or equal to.</p> <p>When this value is set it cannot be equal to or less than the <i>Min</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Max</i> Inclusive Constraint and <i>Max</i> Exclusive Constraint properties together for the same simple type.</p>

Exclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Min</i> Inclusive Constraint and <i>Min</i> Exclusive Constraint properties together for the same simple type.</p>
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than.</p> <p>When this value is set it cannot be equal to or less than the <i>Min</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Max</i> Inclusive Constraint and <i>Max</i> Exclusive Constraint properties together for the same simple type.</p>

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • preserve. If you set the property to preserve, all white space characters including carriage return, line feed and tab are preserved. • replace. If you set the property to replace, all carriage return, line feed and tab characters are replaced with a space character. • collapse. If you set the property to collapse, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Enumerations

Property	Type	Meaning
Enumerations	String	<p><i>Enumerations</i> constrain the values to the list that is specified in this property. For example, you could create a simple type called <i>RainbowColors</i>. You would then add Red, Orange, Yellow, Green, Blue, Indigo, and Violet to the enumerations list.</p> <p>You will need to ensure that you have all variations of the data that you are likely to receive in the message defined in the list. For example, Yellow, yellow, yel, y could be possible variations of a single color.</p> <p>Select Add to add a default enumeration. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see “Using regular expressions to parse data elements” on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Logical properties for value constraints for simple type integer types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Inclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than or equal to.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Min</i> Inclusive Constraint and <i>Min</i> Exclusive Constraint properties together for the same simple type.</p>

Property	Type	Meaning
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than or equal to.</p> <p>When this value is set it cannot be equal to or less than the <i>Min Inclusive Constraint</i> property.</p> <p>You cannot specify both <i>Max Inclusive Constraint</i> and <i>Max Exclusive Constraint</i> properties together for the same simple type.</p>

Exclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max Inclusive Constraint</i> property.</p> <p>You cannot specify both <i>Min Inclusive Constraint</i> and <i>Min Exclusive Constraint</i> properties together for the same simple type.</p>
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than.</p> <p>When this value is set it cannot be equal to or less than the <i>Min Inclusive Constraint</i> property.</p> <p>You cannot specify both <i>Max Inclusive Constraint</i> and <i>Max Exclusive Constraint</i> properties together for the same simple type.</p>

Property	Type	Meaning
Fraction Digits	Integer	<p>Set this property to limit the number of digits in the fraction part of a numerical value to the number of digits specified in this property.</p> <p>The value must be greater than or equal to 0 and less than 2147483648.</p> <p>The value set for <i>Fraction Digits</i> cannot be greater than the value specified for <i>Total Digits</i>.</p>

Property	Type	Meaning
Total Digits	Integer	<p>Set this property to set the maximum number of digits in a numerical value to the number of digits specified in this property.</p> <p>The value must be greater than or equal to 0 and less than 2147483648.</p> <p>The value set for <i>Total Digits</i> cannot be less than the value specified for <i>Fraction Digits</i>.</p>

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • preserve. If you set the property to preserve, all white space characters including carriage return, line feed and tab are preserved. • replace. If you set the property to replace, all carriage return, line feed and tab characters are replaced with a space character. • collapse. If you set the property to collapse, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Enumerations

Property	Type	Meaning
Enumerations	String	<p><i>Enumerations</i> constrain the values to the list that is specified in this property. For example, you could create a simple type called <i>RainbowColors</i>. You would then add Red, Orange, Yellow, Green, Blue, Indigo, and Violet to the enumerations list.</p> <p>You will need to ensure that you have all variations of the data that you are likely to receive in the message defined in the list. For example, Yellow, yellow, yel, y could be possible variations of a single color.</p> <p>Select Add to add a default enumeration. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see “Using regular expressions to parse data elements” on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Logical properties for value constraints for simple type interval types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- Interval schema types: duration

Inclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than or equal to.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Min</i> Inclusive Constraint and <i>Min</i> Exclusive Constraint properties together for the same simple type.</p>
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than or equal to.</p> <p>When this value is set it cannot be equal to or less than the <i>Min</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Max</i> Inclusive Constraint and <i>Max</i> Exclusive Constraint properties together for the same simple type.</p>

Exclusive Constraints

Property	Type	Meaning
Min	Integer	<p>The <i>Min</i> property is used to specify the minimum value for which the data in the message must be greater than.</p> <p>When this value is set it cannot be equal to or greater than the <i>Max</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Min</i> Inclusive Constraint and <i>Min</i> Exclusive Constraint properties together for the same simple type.</p>
Max	Integer	<p>The <i>Max</i> property is used to specify the maximum value for which the data in the message must be less than.</p> <p>When this value is set it cannot be equal to or less than the <i>Min</i> Inclusive Constraint property.</p> <p>You cannot specify both <i>Max</i> Inclusive Constraint and <i>Max</i> Exclusive Constraint properties together for the same simple type.</p>

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • preserve. If you set the property to preserve, all white space characters including carriage return, line feed and tab are preserved. • replace. If you set the property to replace, all carriage return, line feed and tab characters are replaced with a space character. • collapse. If you set the property to collapse, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Enumerations

Property	Type	Meaning
Enumerations	String	<p><i>Enumerations</i> constrain the values to the list that is specified in this property. For example, you could create a simple type called <i>RainbowColors</i>. You would then add Red, Orange, Yellow, Green, Blue, Indigo, and Violet to the enumerations list.</p> <p>You will need to ensure that you have all variations of the data that you are likely to receive in the message defined in the list. For example, Yellow, yellow, yel, y could be possible variations of a single color.</p> <p>Select Add to add a default enumeration. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see “Using regular expressions to parse data elements” on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Logical properties for value constraints for simple type string types:

The simple type value constraint properties described here apply to:

- Objects: Simple types
- Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort

Length constraints

Property	Type	Meaning
Length	Integer	<p>The length property is used to specify the exact length of the simple type in bytes or characters.</p> <p>The value must be greater than 0 and less than 2147483648.</p>
Min	Integer	<p>The Min property is used to specify the minimum length of the simple type in bytes or characters.</p> <p>The value must be greater than 0 and less than 2147483648.</p>
Max	Integer	<p>The Max property is used to specify the maximum length of the simple type in bytes or characters.</p> <p>The value must be greater than 0 and less than 2147483648.</p>

Property	Type	Meaning
White Space	Enumerated type	<p>The <i>White Space</i> controls the processing of white space characters received for this type.</p> <p>The value of <i>White Space</i> must be one of;</p> <ul style="list-style-type: none"> • preserve. If you set the property to preserve, all white space characters including carriage return, line feed and tab are preserved. • replace. If you set the property to replace, all carriage return, line feed and tab characters are replaced with a space character. • collapse. If you set the property to collapse, all carriage return, line feed and tab characters are replaced with a space character. Any adjacent white space characters are then collapsed to a single space character and any leading or trailing spaces are stripped from the data.

Enumerations

Property	Type	Meaning
Enumerations	String	<p><i>Enumerations</i> constrain the values to the list that is specified in this property. For example, you could create a simple type called <i>RainbowColors</i>. You would then add Red, Orange, Yellow, Green, Blue, Indigo, and Violet to the enumerations list.</p> <p>You will need to ensure that you have all variations of the data that you are likely to receive in the message defined in the list. For example, Yellow, yellow, yel, y could be possible variations of a single color.</p> <p>Select Add to add a default enumeration. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Patterns

Property	Type	Meaning
Patterns	String	<p><i>Patterns</i> are a regular expression or a series of regular expressions used to constrain the data within the simple type. For further information on patterns and their syntax see "Using regular expressions to parse data elements" on page 521.</p> <p>Select Add to add a default pattern. Overtyping the default with the data you require.</p> <p>If you need to change an entry, select the entry, then click on the entry a second time (as distinct from double-click). The selected entry can then be updated.</p>

Simple type CWF properties:

| There are no properties to show.

Simple type XML properties:

| There are no properties to show.

Simple type TDS properties:

There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Unique properties

A unique can have the following properties;

- “Unique logical properties” on page 45
- “Unique CWF properties” on page 52
- “Unique XML properties” on page 62
- “Unique TDS properties” on page 76
- “Documentation properties for all message set objects” on page 20

Unique logical properties:

There are no properties to show.

Unique CWF properties:

There are no properties to show.

Unique XML properties:

There are no properties to show.

Unique TDS properties:

There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Wildcard attribute properties

A wildcard attribute can have the following properties;

- “Wildcard attribute logical properties” on page 45
- “Wildcard attribute CWF properties” on page 53
- “Wildcard attribute XML properties” on page 62
- “Wildcard attribute TDS properties” on page 76

- “Documentation properties for all message set objects” on page 20

Wildcard attribute logical properties:

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Property	Type	Meaning
Process Content	Enumerated type	<p>If a message contains an attribute that corresponds to a wildcard in the message model, <i>Process Content</i> defines how the attribute is validated.</p> <p>Select from;</p> <ul style="list-style-type: none"> • strict. The parser can only match against attributes declared in the specified namespace. • lax. The parser attempts to match against attributes declared in any accessible namespace. If the specified namespace cannot be found, an error will not be generated. • skip. If you select skip the parser does not perform any validation on the attribute.

Wildcard attribute CWF properties:

| There are no properties to show.

Wildcard attribute XML properties:

| There are no properties to show.

Wildcard attribute TDS properties:

| There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

- | 1. Key, Keyref, and Unique objects do not have documentation properties.

Wildcard element properties

A wildcard element can have the following properties;

- “Wildcard element logical properties” on page 46

- “Wildcard element CWF properties” on page 53
- “Wildcard element XML properties” on page 62
- “Wildcard element TDS properties” on page 76
- “Documentation properties for all message set objects” on page 20

Wildcard element logical properties:

Property	Type	Meaning
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <no target namespace> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>

Property	Type	Meaning
Process Content	Enumerated type	<p>If a message contains an attribute that corresponds to a wildcard in the message model, <i>Process Content</i> defines how the attribute is validated.</p> <p>Select from;</p> <ul style="list-style-type: none"> • strict. The parser can only match against attributes declared in the specified namespace. • lax. The parser attempts to match against attributes declared in any accessible namespace. If the specified namespace cannot be found, an error will not be generated. • skip. If you select skip the parser does not perform any validation on the attribute.

Occurrences

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Wildcard element CWF properties:

There are no properties to show.

Wildcard element XML properties:

There are no properties to show.

Wildcard element TDS properties:

There are no properties to show.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Deprecated message model object properties

Some objects in the message model are deprecated. Examples of such deprecation are objects that are only created as a result of migrating a message set from WebSphere MQ Integrator Broker Version 2.1.

There are two ways of accessing the reference information for the properties of deprecated message model objects. The following topics allow you to access the property information by property kind:

- “Logical properties for deprecated message model objects”
- “Physical properties for deprecated message model objects” on page 394
- “Documentation properties for all message set objects” on page 20

Alternatively, you can access the property information by object, starting from the following topic:

- “Deprecated message model object properties by object” on page 397

Logical properties for deprecated message model objects

Logical property information is available for the following deprecated objects:

- “Compound element logical properties” on page 392
- “Embedded simple type logical properties” on page 393

Compound element logical properties

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none"> • - the hyphen • _ the underscore • . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <i>xml</i> or any variant (for example <i>Xml</i>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <code><no target namespace></code> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>
Nilable	Check box	Select this if you want the element to be able to be defined as null. This is distinct from being empty where there is no data in the element.
Abstract	Check box	Select this if you do not want the element to appear in the message, but require one of the members of its substitution group to appear in its place.

Value

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>

Occurrences

Property	Type	Meaning
Min Occurs	Integer	Specify the minimum number of times that the object can repeat. The default is 1. If the value is set to 0, then the object is optional. If a value is set, it must be less than or equal to the value in <i>Max Occurs</i> .
Max Occurs	Integer	Specify the maximum number of times that the object can repeat. The default is 1. If this property is not set, then the object can not occur more than once. It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.

Compound element complex type logical properties:

Only the complex type properties shown in the tables below are applicable to compound elements.

Property	Type	Meaning
Name	String	This property is set to **ANONYMOUS** and cannot be changed.

Content:

Property	Type	Meaning
Group Reference	Button	This radio button is already selected and cannot be changed.
Group Name	Enumerated type	The Group Name is the name of the group that this complex type is referring to. The groups available to be referenced can be selected from the drop down list.

Compound element value constraint properties:

The properties for compound element value constraints are identical to simple type value constraints. See “Simple type logical value constraints” on page 44 for details.

Embedded simple type logical properties

Occurrences

Property	Type	Meaning
Min Occurs	Integer	Specify the minimum number of times that the object can repeat. The default is 1. If the value is set to 0, then the object is optional. If a value is set, it must be less than or equal to the value in <i>Max Occurs</i> .

Property	Type	Meaning
Max Occurs	Integer	Specify the maximum number of times that the object can repeat. The default is 1. If this property is not set, then the object can not occur more than once. It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.

Physical properties for deprecated message model objects

Property information is available for deprecated objects within:

- “Custom Wire Format properties for deprecated message model objects”
- “XML wire format physical properties for deprecated message model objects” on page 395
- “Tagged/delimited string format physical properties for deprecated objects” on page 396

Custom Wire Format properties for deprecated message model objects

Custom wire format physical property information is available for the following deprecated objects:

- “Compound element CWF properties”
- “Embedded simple type CWF properties”

Compound element CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _BaseValueBinary	Boolean types - ComIbmMrm _BaseValueBoolean	DateTime types - ComIbmMrm _BaseValueDateTime	Decimal types - ComIbmMrm _BaseValueDecimal
Float types - ComIbmMrm _BaseValueFloat	Integer types - ComIbmMrm _BaseValueInt	String types - ComIbmMrm _BaseValueString	

Compound element complex type CWF properties:

There are no properties to show.

Embedded simple type CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _AnonBinary	Boolean types - ComIbmMrm _AnonBoolean	DateTime types - ComIbmMrm _AnonDate - ComIbmMrm _AnonDateTime - ComIbmMrm _AnonGDay - ComIbmMrm _AnonGMonth - ComIbmMrm _AnonGMonthDay - ComIbmMrm _AnonGYear - ComIbmMrm _AnonGYearMonth - ComIbmMrm _AnonTime	Decimal types - ComIbmMrm _AnonDecimal
Float types - ComIbmMrm _AnonFloat	Integer types - ComIbmMrm _AnonInt	String types - ComIbmMrm _AnonString	

XML wire format physical properties for deprecated message model objects

XML wire format physical property information is available for the following deprecated objects:

- “Compound element XML properties”
- “Embedded simple type XML properties” on page 396

Compound element XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _BaseValueBinary	Boolean types - ComIbmMrm _BaseValueBoolean	DateTime types - ComIbmMrm _BaseValueDateTime	Decimal types - ComIbmMrm _BaseValueDecimal
Float types - ComIbmMrm _BaseValueFloat	Integer types - ComIbmMrm _BaseValueInt	String types - ComIbmMrm _BaseValueString	

Compound element complex type XML properties:

l There are no properties to show.

Embedded simple type XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _AnonBinary	Boolean types - ComIbmMrm _AnonBoolean	DateTime types - ComIbmMrm _AnonDate - ComIbmMrm _AnonDateTime - ComIbmMrm _AnonGDay - ComIbmMrm _AnonGMonth - ComIbmMrm _AnonGMonthDay - ComIbmMrm _AnonGYear - ComIbmMrm _AnonGYearMonth - ComIbmMrm _AnonTime	Decimal types - ComIbmMrm _AnonDecimal
Float types - ComIbmMrm _AnonFloat	Integer types - ComIbmMrm _AnonInt	String types - ComIbmMrm _AnonString	

Tagged/delimited string format physical properties for deprecated objects

TDS format physical property information is available for the following deprecated objects:

- “Compound element TDS properties”
- “Embedded simple type TDS properties” on page 397

Compound element TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _BaseValueBinary	Boolean types - ComIbmMrm _BaseValueBoolean	DateTime types - ComIbmMrm _BaseValueDateTime	Decimal types - ComIbmMrm _BaseValueDecimal
Float types - ComIbmMrm _BaseValueFloat	Integer types - ComIbmMrm _BaseValueInt	String types - ComIbmMrm _BaseValueString	

Compound element complex type TDS properties:

The TDS properties for compound element complex types are identical to the TDS properties for normal complex types. See “Complex type TDS properties” on page 63 for details.

Embedded simple type TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _AnonBinary	Boolean types - ComIbmMrm _AnonBoolean	DateTime types - ComIbmMrm _AnonDate - ComIbmMrm _AnonDateTime - ComIbmMrm _AnonGDay - ComIbmMrm _AnonGMonth - ComIbmMrm _AnonGMonthDay - ComIbmMrm _AnonGYear - ComIbmMrm _AnonGYearMonth - ComIbmMrm _AnonTime	Decimal types - ComIbmMrm _AnonDecimal
Float types - ComIbmMrm _AnonFloat	Integer types - ComIbmMrm _AnonInt	String types - ComIbmMrm _AnonString	

Documentation properties for all message set objects

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Deprecated message model object properties by object

The following deprecated objects have properties that can be viewed or set:

- “Compound element properties” on page 398
- “Embedded simple type properties” on page 456

Compound element properties

A compound element can have the following properties;

- “Compound element logical properties” on page 392
- “Compound element CWF properties” on page 394
- “Compound element XML properties” on page 395
- “Compound element TDS properties” on page 396
- “Documentation properties for all message set objects” on page 20

Compound element logical properties:

Property	Type	Meaning
Name	String	<p>Specify a name for the object when you create it.</p> <p>Names can consist of virtually any alphanumeric character including the letters <i>A</i> through <i>Z</i>, <i>a</i> through <i>z</i> and the digits <i>0</i> through <i>9</i>.</p> <p>They may also include the following punctuation characters;</p> <ul style="list-style-type: none">• - the hyphen• _ the underscore• . the period <p>Names can only start with the letters, ideograms or the underscore character and not with a number, hyphen or period.</p> <p>Names beginning with <code>xml</code> or any variant (for example <code>Xml</code>) are reserved by the XML standards specification.</p> <p>Further details of naming conventions and allowable characters can be found in the Extensible Markup Language (XML) specification that can be found on the World Wide Web Consortium (W3C) Web site.</p>
Namespace	Enumerated type	<p>Namespaces are a simple method for qualifying element and attribute names by associating them with namespaces identified by URI references.</p> <p>If <code><no target namespace></code> is displayed, a namespace has not been set for this object.</p> <p>If the property is inactive, the message set has not been configured to support namespaces.</p> <p>Where the property is active, namespaces that are available for selection will be displayed in the drop down list.</p>
Nilable	Check box	Select this if you want the element to be able to be defined as null. This is distinct from being empty where there is no data in the element.
Abstract	Check box	Select this if you do not want the element to appear in the message, but require one of the members of its substitution group to appear in its place.

Value

Property	Type	Meaning
Default	Button and String	<p>This is the default setting for the Value properties.</p> <p>If a <i>Default</i> value is set for an attribute and a message is received, the value of the attribute is set to the data received from the attribute in the message.</p> <p>If no attribute information is received in the message then the default value that you set here is used to populate the attribute property.</p>
Fixed	Button and String	<p>If a <i>Fixed</i> value is set for an attribute and a message is received, if the attribute exists in the message, the data in the attribute property of the message must match that held in the fixed value of the attribute definition.</p> <p>If the attribute does not exist in the message, the broker will create the attribute and populate it with the data held in the fixed value of the attribute definition.</p>

Occurrences

Property	Type	Meaning
Min Occurs	Integer	<p>Specify the minimum number of times that the object can repeat. The default is 1.</p> <p>If the value is set to 0, then the object is optional.</p> <p>If a value is set, it must be less than or equal to the value in <i>Max Occurs</i>.</p>
Max Occurs	Integer	<p>Specify the maximum number of times that the object can repeat. The default is 1.</p> <p>If this property is not set, then the object can not occur more than once.</p> <p>It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.</p>

Compound element value constraint properties:

The properties for compound element value constraints are identical to simple type value constraints. See “Simple type logical value constraints” on page 44 for details.

Compound element CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types	Boolean types	DateTime types	Decimal types
- ComIbmMrm _BaseValueBinary	- ComIbmMrm _BaseValueBoolean	- ComIbmMrm _BaseValueDateTime	- ComIbmMrm _BaseValueDecimal
Float types	Integer types	String types	
- ComIbmMrm _BaseValueFloat	- ComIbmMrm _BaseValueInt	- ComIbmMrm _BaseValueString	

CWF properties for compound element binary types:

The Custom Wire Format properties described here apply to:

- Objects: Compound elements

Physical representation

Property	Type	Meaning
Length Count	Button and Integer	<p>If you have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1.</p> <p>The maximum value that you can specify is 2147483647.</p> <p>The default value is empty (not set).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for compound element boolean types:

The Custom Wire Format properties described here apply to:

- Objects: Compound elements

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for compound element dateTime types:

The Custom Wire Format properties described here apply to:

- Objects: Compound elements

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Fixed Length String. The element's length is determined by other length properties below. Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. Packed Decimal. The date<code>Time</code> is coded as a Packed Decimal number. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. Binary. The date<code>Time</code> is encoded as a binary sequence of bytes. If you select this option, the range of symbols that you can specify for the Format String property is less than the range of symbols you can specify if you select a string option (see "Date<code>Time</code> formats" on page 527 for details). Time Seconds. This value supports C <code>time_t</code> and Java Date and Time objects. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. Time Milliseconds. This value supports C <code>time_t</code> and Java Date and Time objects. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. <p>The default value is fixed length string.</p>
Date <code>Time</code> Format	String	<p>Specify a template for date and time.</p> <p>The default date<code>Time</code> format is dependent on the logical type of the object. For information on the defaults for the date<code>Time</code> format according to the logical type see "Date<code>Time</code> defaults by logical type" on page 531.</p> <p>If you set the <i>Physical Type</i> to Binary, the template is restricted to those components defined in "Date<code>Time</code> as STRING data" on page 527. If you set the <i>Physical Type</i> to Packed Decimal, Time Seconds, or Time Milliseconds, the template is restricted to those components that represent numbers. In these cases, you must update this <i>Date<code>Time</code> Format</i> property.</p> <p>See "Date<code>Time</code> formats" on page 527 for details of date and time formats.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String, Packed Decimal, or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1 for all three physical types.</p> <p>The maximum value that you can specify is 256 for Fixed Length String, 10 for Packed Decimal, and 2147483647 for Binary.</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>If you have set the <i>Physical Type</i> property to Packed Decimal, Time Seconds, or Time Milliseconds, select (the default) or unselect <i>Signed</i>. If you have selected another value for <i>Physical Type</i>, this property is invalid.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. Use this option when the value you have set for <i>Encoding Null Value</i> to specify a null date is not a dateTime value, or does not conform to the standard dateTime format yyyy-MM-dd 'T'HH:mm:ss. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>If you set the <i>Encoding Null</i> property to NULLPadFill, this property is disabled (grayed out).</p> <p>If you set the <i>Encoding Null</i> property to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralValue, you can enter any value that is the same length as the field.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralFill, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for compound element decimal types:

The Custom Wire Format properties described here apply to:

- Objects: Compound elements

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	Select one of the following from the drop-down list: <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.

Property	Type	Meaning
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. • If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i> .
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>

Property	Type	Meaning
Virtual Decimal Point	Integer	Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a decimal element containing 1234 with a Virtual Decimal value of 3 is 1.234. This is equivalent to 'V' or 'P' in a COBOL picture clause. There is no C equivalent
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to <i>External Decimal</i> , select <i>Left Justify</i> or <i>Right Justify</i> (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • <code>NULLPadFill</code>. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • <code>NULLLiteralValue</code>. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • <code>NULLLiteralFill</code>. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <i>c</i> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <i>YY</i> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <i>YY</i> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <i>xxxx</i> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set <i>Byte Alignment Pad</i> property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set <i>Byte Alignment Pad</i> property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for compound element float types:

The Custom Wire Format properties described here apply to:

- Objects: Compound elements

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Float. This equates to the data type FLOAT or DOUBLE in C or the COMP-1 or COMP-2 data type in COBOL. This is the default value. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer, Packed Decimal, and Float are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Float, select a value from the drop-down list. The default value is 8. • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. • If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select or deselect (unsigned, the default) this property. If you have set <i>Physical Type</i> to Float, this is selected. This property is used in conjunction with <i>Sign Orientation</i> .
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a float element containing 1234 with a Virtual Decimal value of 3 is 1.234.</p> <p>This is not applicable if you have set <i>Physical Type</i> to Float.</p>
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for compound element integer types:

The Custom Wire Format properties described here apply to:

- Objects: Compound elements

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you have set <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 6. • If you have set <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 11.
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for compound element string types:

The Custom Wire Format properties described here apply to:

- Objects: Compound elements

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none">• Fixed Length String. The element's length is determined by other length properties below.• Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units.• Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding.• Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. <p>The default is Fixed Length String.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 0 (zero), the maximum value that you can specify is 2147483647</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	STRING	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. If specified, its length must be equal to the length of the string element, with the exception of <code>NULLLiteralFill</code>.</p> <p>The default value is empty (not set).</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <code>c</code> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <code>YY</code> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <code>YY</code> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <code>xxxx</code> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>

Property	Type	Meaning
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Compound element XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _BaseValueBinary	Boolean types - ComIbmMrm _BaseValueBoolean	DateTime types - ComIbmMrm _BaseValueDateTime	Decimal types - ComIbmMrm _BaseValueDecimal
Float types - ComIbmMrm _BaseValueFloat	Integer types - ComIbmMrm _BaseValueInt	String types - ComIbmMrm _BaseValueString	

XML wire format properties for compound element binary types:

The XML Wire Format properties described here apply to:

- Objects: Compound elements

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> You must also set <i>Value Attribute Name</i> to the same value for the two objects. Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML wire format properties for compound element boolean types:

The XML Wire Format properties described here apply to:

- Objects: Compound elements

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML wire format properties for compound element dateTime types:

The XML Wire Format properties described here apply to:

- Objects: Compound elements

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see "DateTime defaults by logical type" on page 531.</p> <p>See "DateTime formats" on page 527 for details of dateTime formats.</p>

XML wire format properties for compound element decimal types:

The XML Wire Format properties described here apply to:

- Objects: Compound elements

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML wire format properties for compound element float types:

The XML Wire Format properties described here apply to:

- Objects: Compound elements

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML wire format properties for compound element integer types:

The XML Wire Format properties described here apply to:

- Objects: Compound elements

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrIDVal</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. XMLElementAttrIDVal. This option combines the two options, <i>XMLElementAttrID</i> and <i>XMLElementAttrVal</i>. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value <i>XMLElementAttrID</i> for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is <code>id</code>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <code>XMLElement</code>, <code>XMLAttribute</code>, or <code>XMLElementAttrVal</code>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <code>val</code>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

XML wire format properties for compound element string types:

The XML Wire Format properties described here apply to:

- Objects: Compound elements

Field identification

A number of the following properties will only become active depending on the value that *Render* property is set to.

Property	Type	Meaning
Render	Enumerated Type	<p>Specify how the instantiated object or type is rendered (output) in the resulting XML document. Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • XMLElement. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the content of the child element. If you select this value for more than one object, and set their <i>XML Name</i> property to the same value, both objects must refer to the same element. This is the default value for element objects. • XMLAttribute. If you select this value, the object (or type) is rendered as an attribute of the parent XML object. The identity of the child is determined by the attribute name. The value is the attribute value. This is only valid for simple elements. If you select this value for more than one object, you must set their <i>XML Name</i> property to different values. This is the default value for attribute objects. • XMLElementAttrID. If you select this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the value of a specified attribute of the child. The value is the content of the child element. You must add an attribute to the child element with an attribute name as specified in <i>ID Attribute Name</i> and a value as specified in <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrIDVal for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. • XMLElementAttrVal. If you specify this value, the object (or type) is rendered as a child XML element of the parent complex type. The identity of the child is determined by the tag name of the child. The value is the value of a specified attribute. The name of the attribute is specified in <i>Value Attribute Name</i>. • XMLElementAttrIDVal. This option combines the two options, XMLElementAttrID and XMLElementAttrVal. The object is rendered as a child of the parent complex type. The identity of the child is determined by the value of <i>ID Attribute Name</i>. The value is the value of <i>ID Attribute Value</i>. If you select this value for one object, and set this same value or the value XMLElementAttrID for a second object, and set <i>XML Name</i>, <i>ID Attribute Name</i>, <i>ID Attribute Value</i> to the same values: <ul style="list-style-type: none"> – You must also set <i>Value Attribute Name</i> to the same value for the two objects. – Both objects must refer to the same element. <p>“XML Message rendering options” on page 61 shows some examples of how these rendering options affect the XML output.</p>

Property	Type	Meaning
XML Name	String	<p>Enter a value for the XML element name. This property specifies the name for the XML start tag or attribute for the element (or attribute) in an XML document (message).</p> <p>This can be used to provide name mapping when the MRM identifier needs to be different from the XML name, for example because of different namespace rules. It must be a valid XML name.</p> <p>You cannot specify a name that is already used for another element (or attribute) , or for a message. No two elements (or attribute) or messages can have the same XML name.</p> <p>If you do not set a value, it defaults to that of the element's identifier. If the element's identifier is a prefixed identifier, it defaults to the identifier with the caret character (^) replaced by an underscore (_).</p>
ID Attribute Name	String	<p>Specify the name of the attribute used to identify the child. This must be a valid XML Attribute Name. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is <i>id</i>.</p>
Namespace	String	Enter the namespace associated with the <i>ID Attribute</i> .
ID Attribute Value	String	<p>Specify the value of the attribute used to identify the child. This property is ignored and cannot be changed (the field is disabled) if <i>Render</i> is set to <i>XMLElement</i>, <i>XMLAttribute</i>, or <i>XMLElementAttrVal</i>.</p> <p>The default value is the identifier of the child.</p>
Value Attribute Name	String	<p>Specify the name of the attribute used for the value of the child. This must be a valid XML Attribute Name. This is only used if required by the setting of <i>Render</i>.</p> <p>The default value is <i>val</i>.</p>
Namespace	String	Enter the namespace associated with the <i>Value Attribute</i> .

Compound element TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _BaseValueBinary	Boolean types - ComIbmMrm _BaseValueBoolean	DateTime types - ComIbmMrm _BaseValueDateTime	Decimal types - ComIbmMrm _BaseValueDecimal
Float types - ComIbmMrm _BaseValueFloat	Integer types - ComIbmMrm _BaseValueInt	String types - ComIbmMrm _BaseValueString	

TDS format properties for compound element binary types:

The TDS Format properties described here apply to:

- Objects: Compound elements

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p> <p>Regardless of the value of the <i>Data Element Separator</i> property, either the <i>Length</i> or <i>Length Reference</i> property must be set.</p>
Repeating Element Delimiter	String	<p>Specify the delimiter to be used between repeating elements.</p>
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS format properties for compound element boolean types:

The TDS Format properties described here apply to:

- Objects: Compound elements

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS format properties for compound element datetime types:

The TDS Format properties described here apply to:

- Objects: Compound elements

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default DateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see “DateTime defaults by logical type” on page 531.</p> <p>See “DateTime formats” on page 527 for details of date and time formats.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none">• NULLPadFill. This is only valid for fixed length objects. This is the default value.• NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field.• NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for compound element decimal types:

The TDS Format properties described here apply to:

- Objects: Compound elements

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>

Property	Type	Meaning
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present.
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for compound element float types:

The TDS Format properties described here apply to:

- Objects: Compound elements

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present. • Exponential Notation: data is written out to the bit stream as a signed value having the format [sign1]a.bbbe[sign2]ccc where: <ul style="list-style-type: none"> – [sign1] is the value of Negative Sign if the value is negative – a is a single decimal digit – bbb is one or more decimal digits – [sign2] is the value of Negative Sign if the exponent is negative – ccc is exactly three decimal digits (the exponent) <p>[sign1] and [sign2] are absent if the value and exponent, respectively, are positive.</p> <p>For example, the value -123.456 is represented as -1.23456e002 and the value 0.00012 is represented as 1.2e-004 in the output bit stream (assuming the value of <i>Negative Sign</i> is "-" and <i>Sign Orientation</i> is Leading).</p> <p>The value -0.00012 is represented as 1.2*e*004 if <i>Negative Sign</i> is "*" and <i>Sign Orientation</i> is Trailing.</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to <i>None</i>, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property <i>Leading</i>, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to <i>Trailing</i>, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • <i>NULLPadFill</i>. This is only valid for fixed length objects. This is the default value. • <i>NULLLogicalValue</i>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • <i>NULLLiteralValue</i>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <i>dateTime</i> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <i>dateTime</i> object to <i>NULLLogicalValue</i>, you must set this property to an ISO8601 <i>dateTime</i> format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for compound element integer types:

The TDS Format properties described here apply to:

- Objects: Compound elements

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.</p>
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p> <p>Regardless of the value of the <i>Data Element Separator</i> property, either the <i>Length</i> or <i>Length Reference</i> property must be set.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid for fixed length objects. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for compound element string types:

The TDS Format properties described here apply to:

- Objects: Compound elements

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.</p>
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Interpret Element Value	Enumerated Type	<p>Specify if values stored within this object must be interpreted as having significance for the parser and, if so, the type of interpretation that must occur. This interpretation is generally standard-specific and is therefore hard coded.</p> <p>The possible values for this property are:</p> <ul style="list-style-type: none"> • None (the default value) • EDIFACT Service String • X12 Service String • Message Key • EDIFACT Syntax Level ID
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Property	Type	Meaning
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i> : <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. For full information about using these options, see “TDS Null handling options” on page 517.
Encoding Null Value	String	The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero. If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Embedded simple type properties

An embedded simple type can have the following properties;

- “Embedded simple type logical properties” on page 393
- “Embedded simple type CWF properties” on page 394
- “Embedded simple type XML properties” on page 396
- “Embedded simple type TDS properties” on page 397
- “Documentation properties for all message set objects” on page 20

Embedded simple type logical properties:

Occurrences

Property	Type	Meaning
Min Occurs	Integer	Specify the minimum number of times that the object can repeat. The default is 1. If the value is set to 0, then the object is optional. If a value is set, it must be less than or equal to the value in <i>Max Occurs</i> .
Max Occurs	Integer	Specify the maximum number of times that the object can repeat. The default is 1. If this property is not set, then the object can not occur more than once. It can also be set to -1 to indicate that the limit is unbounded and there are no maximum number of occurrences.

Embedded simple type CWF properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _AnonBinary	Boolean types - ComIbmMrm _AnonBoolean	DateTime types - ComIbmMrm _AnonDate - ComIbmMrm _AnonDateTime - ComIbmMrm _AnonGDay - ComIbmMrm _AnonGMonth - ComIbmMrm _AnonGMonthDay - ComIbmMrm _AnonGYear - ComIbmMrm _AnonGYearMonth - ComIbmMrm _AnonTime	Decimal types - ComIbmMrm _AnonDecimal
Float types - ComIbmMrm _AnonFloat	Integer types - ComIbmMrm _AnonInt	String types - ComIbmMrm _AnonString	

CWF properties for embedded simple type binary types:

The Custom Wire Format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
Length Count	Button and Integer	<p>If you have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1.</p> <p>The maximum value that you can specify is 2147483647.</p> <p>The default value is empty (not set).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for embedded simple type boolean types:

The Custom Wire Format properties described here apply to:

- Objects: Embedded simple types

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	Specify how the object is aligned from the start of the message. Select one of: <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.

Property	Type	Meaning
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for embedded simple type dateTime types:

The Custom Wire Format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> Fixed Length String. The element's length is determined by other length properties below. Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units. Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding. Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. Packed Decimal. The date<code>Time</code> is coded as a Packed Decimal number. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. Binary. The date<code>Time</code> is encoded as a binary sequence of bytes. If you select this option, the range of symbols that you can specify for the Format String property is less than the range of symbols you can specify if you select a string option (see "Date<code>Time</code> formats" on page 527 for details). Time Seconds. This value supports C <code>time_t</code> and Java Date and Time objects. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. Time Milliseconds. This value supports C <code>time_t</code> and Java Date and Time objects. It is valid only if the <i>Date<code>Time</code> Format</i> property represents numeric-only data. <p>The default value is fixed length string.</p>
Date <code>Time</code> Format	String	<p>Specify a template for date and time.</p> <p>The default date<code>Time</code> format is dependent on the logical type of the object. For information on the defaults for the date<code>Time</code> format according to the logical type see "Date<code>Time</code> defaults by logical type" on page 531.</p> <p>If you set the <i>Physical Type</i> to Binary, the template is restricted to those components defined in "Date<code>Time</code> as STRING data" on page 527. If you set the <i>Physical Type</i> to Packed Decimal, Time Seconds, or Time Milliseconds, the template is restricted to those components that represent numbers. In these cases, you must update this <i>Date<code>Time</code> Format</i> property.</p> <p>See "Date<code>Time</code> formats" on page 527 for details of date and time formats.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String, Packed Decimal, or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 1 for all three physical types.</p> <p>The maximum value that you can specify is 256 for Fixed Length String, 10 for Packed Decimal, and 2147483647 for Binary.</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>If you have set the <i>Physical Type</i> property to Packed Decimal, Time Seconds, or Time Milliseconds, select (the default) or unselect <i>Signed</i>. If you have selected another value for <i>Physical Type</i>, this property is invalid.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. Use this option when the value you have set for <i>Encoding Null Value</i> to specify a null date is not a dateTime value, or does not conform to the standard dateTime format yyyy-MM-dd 'T'HH:mm:ss. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>If you set the <i>Encoding Null</i> property to NULLPadFill, this property is disabled (grayed out).</p> <p>If you set the <i>Encoding Null</i> property to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralValue, you can enter any value that is the same length as the field.</p> <p>If you set the <i>Encoding Null</i> property to NULLLiteralFill, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes

Property	Type	Meaning
Leading Skip Count	Integer	Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to the first instance only.
Trailing Skip Count	Integer	Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property. For repeating objects, this property is applied to all instances.

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message. For information about reordering elements, see Reordering objects.

CWF properties for embedded simple type decimal types:

The Custom Wire Format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	Select one of the following from the drop-down list: <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.

Property	Type	Meaning
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. • If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i> .
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>

Property	Type	Meaning
Virtual Decimal Point	Integer	Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a decimal element containing 1234 with a Virtual Decimal value of 3 is 1.234. This is equivalent to 'V' or 'P' in a COBOL picture clause. There is no C equivalent
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to <i>External Decimal</i> , select <i>Left Justify</i> or <i>Right Justify</i> (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • <code>NULLPadFill</code>. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • <code>NULLLogicalValue</code>. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • <code>NULLLiteralValue</code>. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • <code>NULLLiteralFill</code>. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <i>c</i> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <i>YY</i> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <i>YY</i> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <i>xxxx</i> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set <i>Byte Alignment Pad</i> property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set <i>Byte Alignment Pad</i> property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for embedded simple type float types:

The Custom Wire Format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Float. This equates to the data type FLOAT or DOUBLE in C or the COMP-1 or COMP-2 data type in COBOL. This is the default value. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer, Packed Decimal, and Float are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Float, select a value from the drop-down list. The default value is 8. • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you set the <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 10. • If you set the <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 256. (Numbers greater than the maximum COBOL PICTURE clause of 18 are assumed to be 18.)

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	Select or deselect (unsigned, the default) this property. If you have set <i>Physical Type</i> to Float, this is selected. This property is used in conjunction with <i>Sign Orientation</i> .
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
Virtual Decimal Point	Integer	<p>Specify the number of places to the left (for a positive value) or right (for a negative value) that a decimal point should be moved from its assumed position. For example, a float element containing 1234 with a Virtual Decimal value of 3 is 1.234.</p> <p>This is not applicable if you have set <i>Physical Type</i> to Float.</p>
String Justification	Enumerated Type	If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i> , this property is inactive.

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for embedded simple type integer types:

The Custom Wire Format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Integer. This equates to the data type SHORT or LONG in C, or the COMP, COMP-4, COMP-5, or BINARY numeric data type in COBOL. • Packed Decimal. This equates to the COMP-3 data type in COBOL. • External Decimal. This equates to the data type PIC 9 USAGE DISPLAY in COBOL. <p>The representation of numeric elements can be affected by the Encoding and CodedCharSetId attributes that are set for the WebSphere MQ queue manager:</p> <ul style="list-style-type: none"> • Elements that have <i>Physical Type</i> set to Integer and Packed Decimal are represented in the appropriate WebSphere MQ Encoding value. • Elements that have <i>Physical Type</i> set to External Decimal are represented in the WebSphere MQ CodedCharSetId value.
Length Count	Integer	<p>Enter the number of bytes to specify the element length:</p> <ul style="list-style-type: none"> • If you set the <i>Physical Type</i> to Integer, select 1, 2, or 4 (the default) from the drop-down list. • If you have set <i>Physical Type</i> to Packed Decimal, enter a value between 1 and 6. • If you have set <i>Physical Type</i> to Extended Decimal, enter a value between 1 and 11.
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i>. <p>The default is bytes.</p>
Signed	Boolean	<p>Select (the default) or deselect this property. This property is used in conjunction with <i>Sign Orientation</i>.</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>If you have set <i>Physical Type</i> to External Decimal and you have selected <i>Signed</i>, choose from the following options that represent the COBOL options for displaying numeric data:</p> <ul style="list-style-type: none"> • Included Leading. This sets a bit in the first byte on if the number is negative. No setting is made if the number is positive. For example, the ASCII hexadecimal representation of the number 22 is x'3232'. Using this option, the number +22 would be x'3232' and the number -22 would be x'7232'. This is the default value. • Separate Leading. This sets the first byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. • Included Trailing. This sets a bit in the last byte on if the number is negative. No setting is made if the number is positive. Using this option, the number +22 would be x'3232' and the number -22 would be x'3272'. • Separate Trailing. This sets the last byte of the element to '+' if the number is positive and to '-' if the number is negative. For this option, the length must include the sign byte. <p>If you have set <i>Physical Type</i> to any other value, the value Not Applicable is set for you.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to External Decimal, select Left Justify or Right Justify (the default value) from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to <i>Extended Decimal</i>, and the <i>String Justification</i> property is either <i>Left Justify</i> or <i>Right Justify</i>, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is <i>External Decimal</i>. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. You can specify a nonnumeric value for <i>Encoding Null Value</i>. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above, with the exception of <code>NULLLiteralFill</code>. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select SPACE, NUL, 0x00 or 0xFF from the drop-down list • Enter a character between quotation marks, for example 'c' or "c", where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. • Enter a decimal character code in the form YY where YY is a decimal value. • Enter a Unicode value in the form U+xxxx where xxxx is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

CWF properties for embedded simple type string types:

The Custom Wire Format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
Physical Type	Enumerated Type	<p>Select one of the following from the drop-down list:</p> <ul style="list-style-type: none">• Fixed Length String. The element's length is determined by other length properties below.• Length Encoded String 1. The element's first byte contains the length of the string following the length byte in length units. The maximum length of a Length Encoded String 1 element is 255 length units.• Length Encoded String 2. The element's first two bytes contain the length of the string following the 2 length bytes in length units. The maximum length of a Length Encoded String 2 element is 65535 length units. The two length bytes are in the format of the WebSphere MQ queue manager Encoding.• Null Terminated String. The string ends with the hexadecimal NULL character, X'00'. <p>The default is Fixed Length String.</p>
Length Count	Button and Integer	<p>If you have selected a <i>Physical Type</i> of Fixed Length String or Binary, and have set <i>Length Type</i> to Count, enter the number of length units for the element.</p> <p>The minimum value that you can specify is 0 (zero), the maximum value that you can specify is 2147483647</p> <p>The default value is 0 (zero).</p>
Length Reference	Button and Enumerated Type	<p>If you have selected the length to be defined by <i>Length Reference</i>, select the name of the Integer object that specifies the length of this object. Make your selection from the drop-down list of Integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Property	Type	Meaning
Length Units	Enumerated Type	<p>Subject to the <i>Physical Type</i> that has been set, select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • Bytes. This specifies that X bytes are processed where X is the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • Characters. The meaning of this option depends on the value that you have set in the message's CCSID or that you have set for the message set property <i>Default CCSID</i>. <ul style="list-style-type: none"> – If you have specified an SBCS CCSID, X bytes are processed where X is the value of <i>Length Count</i> or of the INTEGER specified by <i>Length Reference</i>. – If you have specified a DBCS CCSID, Y bytes are processed, where Y is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i> multiplied by two. – If you have specified an MBCS CCSID, the parser reads 1 character at a time and determines whether the character comprises 1 or more bytes. The parser performs Z reads, where Z is the value of <i>Length Count</i> or of the INTEGER specified in <i>Length Reference</i>. • Character Units. This specifies that the size of character unit is determined by the value that you have set in the message's CCSID, or for the message set property <i>Default CCSID</i>. The number of bytes processed is the size of character unit multiplied by the value of <i>Length Count</i>, or of the INTEGER specified in <i>Length Reference</i>. • End of Bitstream. All data until end of bit stream is processed. This option is valid only if the element is the last in the message. If you select this value, you do not need to enter a value in <i>Length Count</i> or <i>Length Reference</i>. <p>The default is bytes.</p>
String Justification	Enumerated Type	<p>If you have set the <i>Physical Type</i> property to Fixed Length String, select Left Justify (the default value) or Right Justify from the drop-down list. If you have selected another value for <i>Physical Type</i>, this property is inactive.</p>

Property	Type	Meaning
Padding Character	String	<p>The padding character is used to fill out the remaining character positions when the string length is less than the specified string size. If you have set the <i>Physical Type</i> property to Fixed Length String, and the <i>String Justification</i> property is either Left Justify or Right Justify, set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid if <i>Physical Type</i> is Fixed Length String. The field is filled with the value specified by the <i>Padding Character</i>. <i>Encoding Null Value</i> must be set to an empty string. • NULLLogicalValue. The <i>Encoding Null Value</i> is transformed to match the required format for the field. This is the default value. • NULLLiteralValue. The <i>Encoding Null Value</i> is directly substituted as if it is a string. • NULLLiteralFill. The field is filled with the value specified by the <i>Encoding Null Value</i>. <i>Encoding Null Value</i> must resolve to a single character.

Property	Type	Meaning
Encoding Null Value	STRING	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. If specified, its length must be equal to the length of the string element, with the exception of <code>NULLLiteralFill</code>.</p> <p>The default value is empty (not set).</p> <p>If you set the <i>Encoding Null</i> property to <code>NULLLiteralFill</code>, the value must resolve to a single character. Set the character in one of the following ways:</p> <ul style="list-style-type: none"> • Select <code>SPACE</code>, <code>NUL</code>, <code>0x00</code> or <code>0xFF</code> from the drop-down list • Enter a character between quotation marks, for example <code>'c'</code> or <code>"c"</code>, where <code>c</code> is any alphanumeric character. • Enter a hexadecimal character code in the form <code>0xYY</code> where <code>YY</code> is a hexadecimal value. • Enter a decimal character code in the form <code>YY</code> where <code>YY</code> is a decimal value. • Enter a Unicode value in the form <code>U+xxxx</code> where <code>xxxx</code> is a Unicode value specified in hexadecimal format.

Byte alignment

Property	Type	Meaning
Byte Alignment	Enumerated Type	<p>Specify how the object is aligned from the start of the message. Select one of:</p> <ul style="list-style-type: none"> • 1 Bytes. This is the default value. • 2 Bytes • 4 Bytes • 8 Bytes • 16 Bytes
Leading Skip Count	Integer	<p>Specify the number of bytes to skip before reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a field defined by C or COBOL data which requires alignment on a 2, 4, 8 or 16 byte boundary. Specify the number of bytes to skip before reading or writing this object. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to the first instance only.</p>
Trailing Skip Count	Integer	<p>Specify the number of bytes to skip after reading or writing this object. The default is 0, the minimum value is 0, and the maximum value is 999999. You can use this value to ignore unwanted fields in a structure, or to model a repeating structure containing fields which require alignment on a 2, 4, 8 or 16 byte boundary. When an output message is written, Skip Count bytes are assigned the value of the message set Byte Alignment Pad property.</p> <p>For repeating objects, this property is applied to all instances.</p>

Repeat

Property	Type	Meaning
Repeat Count	Button and Integer	<p>If you have selected the <i>Repeat Count</i> property, enter the number of occurrences for this object. The minimum value is 0 (zero and one mean that a single occurrence is expected), the maximum value is 2147483647.</p>

Property	Type	Meaning
Repeat Reference	Button and Enumerated Type	<p>If you have selected the <i>Repeat Reference</i> property, select the name of the integer object the value of which specifies the number of occurrences of this object from the drop-down list of integer objects that are defined as siblings of the current object, and occur before it in the structure of the message.</p> <p>For information about reordering elements, see Reordering objects.</p>

Embedded simple type XML properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

<p>Binary types</p> <ul style="list-style-type: none"> - ComIbmMrm _AnonBinary 	<p>Boolean types</p> <ul style="list-style-type: none"> - ComIbmMrm _AnonBoolean 	<p>DateTime types</p> <ul style="list-style-type: none"> - ComIbmMrm _AnonDate - ComIbmMrm _AnonDateTime - ComIbmMrm _AnonGDay - ComIbmMrm _AnonGMonth - ComIbmMrm _AnonGMonthDay - ComIbmMrm _AnonGYear - ComIbmMrm _AnonGYearMonth - ComIbmMrm _AnonTime 	<p>Decimal types</p> <ul style="list-style-type: none"> - ComIbmMrm _AnonDecimal
<p>Float types</p> <ul style="list-style-type: none"> - ComIbmMrm _AnonFloat 	<p>Integer types</p> <ul style="list-style-type: none"> - ComIbmMrm _AnonInt 	<p>String types</p> <ul style="list-style-type: none"> - ComIbmMrm _AnonString 	

XML Wire Format properties for embedded simple type binary types:

The XML wire format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
Encoding	String	<p>Select one of the following values from the drop-down list: :</p> <ul style="list-style-type: none"> • CDatahex (the default) • hex • base64

XML wire format properties for embedded simple type boolean types:

The XML wire format properties described here apply to:

- Objects: Embedded simple types

There are no properties to show.

XML wire format properties for embedded simple type dateTime types:

The XML wire format properties described here apply to:

- Objects: Embedded simple types

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a format string that specifies the rendering of the value for dateTime elements.</p> <p>The default dateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see “DateTime defaults by logical type” on page 531.</p> <p>See “DateTime formats” on page 527 for details of dateTime formats.</p>

XML wire format properties for embedded simple type decimal types:

The XML Wire Format properties described here apply to:

- Objects: Embedded simple types

There are no properties to show.

XML wire format properties for embedded simple type float types:

The XML wire format properties described here apply to:

- Objects: Embedded simple types

There are no properties to show.

XML wire format properties for embedded simple type integer types:

The XML wire format properties described here apply to:

- Objects: Embedded simple types

There are no properties to show.

XML wire format properties for embedded simple type string types:

The XML wire format properties described here apply to:

- Objects: Embedded simple types

There are no properties to show.

Embedded simple type TDS properties:

The properties displayed on the object page and the values that those properties can take, can vary according to the type of the object. For example, the properties for type string are different to those of type boolean. Select the link for the object type from the table below.

Binary types - ComIbmMrm _AnonBinary	Boolean types - ComIbmMrm _AnonBoolean	DateTime types - ComIbmMrm _AnonDate - ComIbmMrm _AnonDateTime - ComIbmMrm _AnonGDay - ComIbmMrm _AnonGMonth - ComIbmMrm _AnonGMonthDay - ComIbmMrm _AnonGYear - ComIbmMrm _AnonGYearMonth - ComIbmMrm _AnonTime	Decimal types - ComIbmMrm _AnonDecimal
Float types - ComIbmMrm _AnonFloat	Integer types - ComIbmMrm _AnonInt	String types - ComIbmMrm _AnonString	

TDS format properties for embedded simple type binary types:

The TDS format properties described here apply to:

- Objects: Embedded simple types

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p> <p>Regardless of the value of the <i>Data Element Separator</i> property, either the <i>Length</i> or <i>Length Reference</i> property must be set.</p>
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS format properties for embedded simple type boolean types:

The TDS format properties described here apply to:

- Objects: Embedded simple types

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

TDS format properties for embedded simple type dateTime types:

The TDS format properties described here apply to:

- Objects: Embedded simple types

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none">• Not Applicable• Left Justify• Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Physical representation

Property	Type	Meaning
DateTime Format	String	<p>Specify a template for date and time.</p> <p>The default DateTime format is dependent on the logical type of the object. For information on the defaults for the dateTime format according to the logical type see “DateTime defaults by logical type” on page 531.</p> <p>See “DateTime formats” on page 527 for details of date and time formats.</p>

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • NULLPadFill. This is only valid for fixed length objects. This is the default value. • NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for embedded simple type decimal types:

The TDS format properties described here apply to:

- Objects: Embedded simple types

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>

Property	Type	Meaning
Data Pattern	String	Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present.
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to None, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property Leading, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to Trailing, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see “TDS Null handling options” on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for embedded simple type float types:

The TDS format properties described here apply to:

- Objects: Embedded simple types

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See “Regular expression syntax” on page 523 for further details.</p>

Property	Type	Meaning
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Virtual Decimal Point	Button and Integer	<p>Specify a nonzero integer representing the position of an implied decimal point within a number, or specify 0 (zero, the default) to use the formatting of Float or Decimal numbers as specified by the <i>Precision</i> property.</p> <p>If you specify a positive integer, the position of the decimal point is moved left from the right hand side of the number. For example, if you specify 3, the decimal value 1234 represents 1.234</p> <p>If you specify a negative integer, the position of the decimal point is moved right from the right hand side of the number. For example, if you specify -3, the decimal value 1234 represents 1234000</p>
Precision	Button and Integer	<p>This value is used if the <i>Virtual Decimal Point</i> property value is 0, indicating that the decimal point is present in the data. It deals with truncation, and specifies the number of digits that should follow the decimal point.</p> <p>Either specify a number of digits:</p> <ul style="list-style-type: none"> • If you set <i>Precision</i> to 0, data is truncated so that the fractional part is lost. For example, the value 123.45 is truncated to 123. • If you set <i>Precision</i> to a number less than the number of fractional digits, data is truncated. For example, the value 123.4567 is truncated to 123.45 if you set <i>Precision</i> to 2. • If you set <i>Precision</i> to a number greater than the number of fractional digits, the value is padded with extra zeros. For example, the value 12.345 is padded to 12.34500 if you set <i>Precision</i> to 5. <p>Or select one of the following from the drop-down list:</p> <ul style="list-style-type: none"> • All Significant Digits (the default): all the significant digits are written to the output bit stream, and there is no decimal separator if there are no fractional digits. • Explicit Decimal Separator: all the significant digits are written to the output bit stream and the decimal separator is always included, even if there are no fractional digits. The decimal separator must be present in the input bitstream, even if no fractional digits are present. • Exponential Notation: data is written out to the bit stream as a signed value having the format [sign1]a.bbbe[sign2]ccc where: <ul style="list-style-type: none"> – [sign1] is the value of Negative Sign if the value is negative – a is a single decimal digit – bbb is one or more decimal digits – [sign2] is the value of Negative Sign if the exponent is negative – ccc is exactly three decimal digits (the exponent) <p>[sign1] and [sign2] are absent if the value and exponent, respectively, are positive.</p> <p>For example, the value -123.456 is represented as -1.23456e002 and the value 0.00012 is represented as 1.2e-004 in the output bit stream (assuming the value of <i>Negative Sign</i> is "-" and <i>Sign Orientation</i> is Leading).</p> <p>The value -0.00012 is represented as 1.2*e*004 if <i>Negative Sign</i> is "*" and <i>Sign Orientation</i> is Trailing.</p>

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to <i>None</i>, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property <i>Leading</i>, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to <i>Trailing</i>, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • <i>NULLPadFill</i>. This is only valid for fixed length objects. This is the default value. • <i>NULLLogicalValue</i>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • <i>NULLLiteralValue</i>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <i>dateTime</i> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <i>dateTime</i> object to <i>NULLLogicalValue</i>, you must set this property to an ISO8601 <i>dateTime</i> format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for embedded simple type integer types:

The TDS format properties described here apply to:

- Objects: Embedded simple types

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.</p>
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.
Length Reference	Enumerated type	<p>Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure.</p> <p>For information about reordering elements, see Reordering objects.</p>

Numeric representation

Property	Type	Meaning
Sign Orientation	Enumerated Type	<p>Select the value that indicates the positioning of a sign symbol within a numeric value, from the drop-down list:</p> <ul style="list-style-type: none"> • None • Leading • Trailing <p>If you set the value for this property to <i>None</i>, this is interpreted as having no sign, and an exception is thrown if a negative number is processed (on either input or output).</p> <p>If you set the value for this property <i>Leading</i>, this indicates that the sign is positioned ahead of the number, for example, -1234. Similarly, if you set this property to <i>Trailing</i>, the sign follows the number, for example, 1234-.</p> <p>If there is no explicit sign set, the number is assumed to be positive.</p>
Positive Sign	String	Specify the value that represents the positive symbol. If no value is set, "+" is assumed. The positive sign is not written when creating an output message, it is only used to recognize the positive sign when parsing a message bitstream.
Negative Sign	String	Specify the value that represents the negative symbol. If no value is set, "-" is assumed.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	<p>Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i>:</p> <ul style="list-style-type: none"> • <i>NULLPadFill</i>. This is only valid for fixed length objects. This is the default value. • <i>NULLLogicalValue</i>. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. • <i>NULLLiteralValue</i>. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For <i>dateTime</i> elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. <p>For full information about using these options, see "TDS Null handling options" on page 517.</p>
Encoding Null Value	String	<p>The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero.</p> <p>If you set the <i>Encoding Null</i> property for a <i>dateTime</i> object to <i>NULLLogicalValue</i>, you must set this property to an ISO8601 <i>dateTime</i> format. These formats are described in "DateTime as STRING data" on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.</p>

TDS format properties for embedded simple type string types:

The TDS format properties described here apply to:

- Objects: Embedded simple types

Field Identification

Property	Type	Meaning
Tag	String	<p>Specify the value used to identify the object in a message bit stream. If the object is simple and the <i>Data Element Separation</i> property of the complex type or types in which the object is a child is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length, this property must contain a non-empty value.</p> <p>Should the object be a complex element and the <i>Data Element Separation</i> property of its parent is Tagged Delimited, Tagged Fixed Length, or Tagged Encoded Length then the property can contain an empty value.</p> <p>The value for this property must be unique for every element in the message set, that is, no two elements in the message set can contain the same value for this property.</p>
Data Pattern	String	<p>Specify the regular expression to be used by the parser to identify the data in the message to be assigned to the object. Used when the <i>Data Element Separation</i> method has been set to Use Data Pattern in the complex type. See "Regular expression syntax" on page 523 for further details.</p>
Length	Integer	<p>Specify the expected length of the object in characters (except in the case of binary objects, in which case the length value represents the length in bytes).</p> <p>This property applies to simple objects and to complex elements with a base type.</p> <p>If you give this property a value of 0, the <i>Length Reference</i> property is checked for a value.</p> <p>If you set the <i>Data Element Separator</i> property for the type to Fixed Length or Fixed Length AL3, either this property, or the <i>Length Reference</i> property, must contain a non 0 (or non NULL) value.</p>
Justification	Enumerated Type	<p>Specify the justification of the object where the data being written or parsed is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Select one of the following values from the drop-down list:</p> <ul style="list-style-type: none"> • Not Applicable • Left Justify • Right Justify

Property	Type	Meaning
Padding Character	String	<p>Specify the padding character to be inserted or interpreted on the writing or parsing of a fixed length object where the data is less than the fixed length value. This property is only used when a value is output as a fixed length string.</p> <p>Set this character in one of the following ways:</p> <ul style="list-style-type: none"> • Select NUL, '0', or SPACE from the drop-down list. • Enter a character between quotes, for example "c" or 'c', where c is any alphanumeric character. • Enter a hexadecimal character code in the form 0xYY where YY is a hexadecimal value. <p>The choice of which of these padding character forms is used for an MRM element will depend on the padding character required and whether the padding character is to be subject to data conversion. In most cases the specification of a padding character will be sufficient and when this padding character is used, it will be converted to the target codepage of the output MRM message being generated. If a padding character is required that cannot easily be entered in the padding character field, then the Unicode mnemonic format can be used to specify the required character. When used, this Unicode value will also be converted to the target codepage of the MRM message being generated. If a padding character is required that is not subject to data conversion, then the hexadecimal format can be used. This gives the option of specifying a padding character that is inserted directly in the output message. If this format is used then you must ensure that this hexadecimal is valid for the codepage of any output messages that are created using these MRM definitions.</p> <p>If you are converting a message from one code page to another, then you must ensure that the converted value of the padding character is valid for this codepage. For example, when converting from ASCII to the codepage 500, if you have specified the numeric 8 as your padding character then this is converted from 0x08 to 0x15, the ASCII and EBCDIC representations of 'back space'.</p> <p>There is a currently a restriction that the value of your padding character should not be greater than U+007F. You should note that if you enter a Unicode mnemonic or numeric value, it is considered to be the character represented by that number in UTF-8.</p> <ul style="list-style-type: none"> • Enter a Unicode value in the form U+xxxxxx where xxxxxx is a Unicode value specified in hexadecimal. The maximum length of the string you can enter is 10. • Select the default value of an empty string.
Interpret Element Value	Enumerated Type	<p>Specify if values stored within this object must be interpreted as having significance for the parser and, if so, the type of interpretation that must occur. This interpretation is generally standard-specific and is therefore hard coded.</p> <p>The possible values for this property are:</p> <ul style="list-style-type: none"> • None (the default value) • EDIFACT Service String • X12 Service String • Message Key • EDIFACT Syntax Level ID
Repeating Element Delimiter	String	Specify the delimiter to be used between repeating elements.

Property	Type	Meaning
Length Reference	Enumerated type	Specify the identifier of a sibling Integer object, the value of which dictates the length of the object in question. The sibling object must be defined before the current object within the message structure. For information about reordering elements, see Reordering objects.

Representation of null values

Property	Type	Meaning
Encoding Null	Enumerated Type	Select one of the following options from the drop-down list. The option that you select determines the value that you must set for the property <i>Encoding Null Value</i> : <ul style="list-style-type: none"> NULLPadFill. This is only valid for fixed length objects. This is the default value. NULLLogicalValue. The <i>Encoding Null Value</i> property is first converted to an actual value, and rendered in the way specified for the field. NULLLiteralValue. This specifies that <i>Encoding Null Value</i> contains a value that is directly substituted as if it is a string. For dateTime elements, use this option if you want to use the <i>Encoding Null Value</i> to test or compare the content of the field in the message. For full information about using these options, see “TDS Null handling options” on page 517.
Encoding Null Value	String	The use of this property depends on the <i>Encoding Null</i> property, described above. The default value is zero. If you set the <i>Encoding Null</i> property for a dateTime object to NULLLogicalValue, you must set this property to an ISO8601 dateTime format. These formats are described in “DateTime as STRING data” on page 527. For example, specify a value conforming to yyyy-MM-dd'T'HH:mm:ss such as 1970-12-01.

Documentation properties for all message set objects:

The documentation property of an object¹ is where you can add information to enhance the understanding of that objects function.

It is a string field and any standard alphanumeric characters can be used.

Note:

1. Key, Keyref, and Unique objects do not have documentation properties.

Additional MRM domain information

This section provides additional information in relation to the MRM domain. This information is categorized into:

- “Additional logical information” on page 502
- “Additional CWF information” on page 503
- “Additional XML information” on page 504
- “Additional TDS information” on page 506
- “DateTime formats” on page 527

Additional logical information

This section provides additional information in relation to the logical model. This information has been categorized into:

- “MRM model restrictions”

MRM model restrictions

This topic provides information on areas where the WebSphere MQ Integrator Broker does not exactly follow the XML Schema specification.

XML Schema features supported only in the message editor:

The following features can be created and edited using the message editor, but are not honored by WebSphere Business Integration Message Broker:

- Pattern facet on non-string data types. The message broker will only validate pattern facets which are applied to simple types based on `xsd:string`.
- Whitespace facet. The message broker does not make use of the whitespace facet at all (although whitespace facets can be included in the message model if necessary). It is possible to accurately control the processing of whitespace using the settings on the physical formats.
- ID attribute. The message model can contain attributes with the name ‘id’, but these will not be checked for uniqueness.

XML Schema exceptions:

The following features may be created and edited using the message editor, but the message broker will process them in a way which differs from the XML Schema specification, as follows:

- Default and fixed values. The processing of default and fixed values depends on the physical format in which the message is parsed. For details on how each physical format uses these fields, refer to the concept topic Relationship to the logical model for the relevant physical format.
- `xsi:type` attribute. The `xsi:type` attribute is not automatically processed by the message broker. An attribute with the name ‘`xsi:type`’ can be included in the message model, and can be processed using a message flow.

Differences in validation:

If validation is enabled in a message flow, the following features/scenarios will not be validated in exactly the same way as a validating XML parser would validate them:

- Any Element/Any Attribute. If the message model contains a wildcard (‘any element’ or ‘any attribute’) the message broker will validate the ‘`processContents`’ field as follows:
 - skip. No checking will be done, any element/attribute will be allowed.
 - lax. No checking will be done, any element/attribute will be allowed.
 - strict. Any element/attribute in the same message set will be allowed.

Note: If all of the definitions for a namespace are included within the same message set, the validation of ‘strict’ will be the same as a validating XML parser.

- Element substitution and 'all' groups. If an element is substitutable, and it occurs within an 'all' group, the following exceptions apply to the validation of the element:
 - A substitutable element within an 'all' group will always be validated as if it were optional.
 - An input message will not be rejected if more than one of the substitutions is used in the same 'all' group.

Additional CWF information

This section provides additional information in relation to the CWF physical format. This information has been categorized into:

- "CWF data conversion"
- "CWF Null handling options" on page 504

CWF data conversion

You can convert an MRM message to a different code page or encoding, or both. To do this, you should set the `CodedCharSetId` and `Encoding` fields in the appropriate output WebSphere MQ header to the target value. The appropriate WebSphere MQ header is the header that precedes and is adjacent to the output message body.

The data conversion performed is dependent on the simple type of each element:

- Binary schema types: `base64Binary`, `hexBinary` objects are not converted.
- Boolean schema types: `boolean` objects are not converted.
- DateTime schema types: `date`, `dateTime`, `gDay`, `gMonth`, `gMonthDay`, `gYear`, `gYearMonth`, time objects are handled as either binary, string, or packed decimals. If a `dateTime` element is defined as binary, it is not converted. If it is defined as string, it is converted as a string element (described below). If it is defined as a packed decimal value, it is converted as decimal with *Physical Type* set to `Packed Decimal` (described below).
- Decimal schema types: `decimal`, `integer`, `negativeInteger`, `nonNegativeInteger`, `nonPositiveInteger`, `positiveInteger` objects with *Physical Type* set to `External Decimal` are converted to the target `CodedCharSetId`. Elements with other *Physical Type* settings are converted to the target Encoding.
- Float schema types: `double`, `float` objects with *Physical Type* set to `External Decimal` are converted to the target `CodedCharSetId`. Elements with other *Physical Type* settings are converted to the target Encoding.
- Integer schema types: `byte`, `int`, `long`, `short`, `unsignedByte`, `unsignedInt`, `unsignedLong`, `unsignedShort` objects with *Physical Type* set to `External Decimal` are converted to the target `CodedCharSetId`. Elements with other *Physical Type* settings are converted to the target Encoding.
- String schema types: `anyURI`, `duration`, `ENTITIES`, `ENTITY`, `ID`, `IDREF`, `IDREFS`, `language`, `Name`, `NCName`, `NMTOKEN`, `NMTOKENS`, `normalizedString`, `NOTATION`, `QName`, `string`, `token` objects are converted to the target `CodedCharSetId` (the length of an object that has *Physical Type* of `Length Encoded String 2` is converted to the target Encoding).

CWF Null handling options

The Custom Wire Format (CWF) supports handling of null values within messages. The Boolean Null Value that you set for the message set is applicable for all the defined objects within the message set.

For more information about the use of nulls, refer to the properties *Encoding Null* and *Encoding Null Value* for objects of each simple type, for example, “CWF properties for element reference and local element dateTime types” on page 133.

Additional XML information

This section provides additional information in relation to the XML physical format. This information has been categorized into:

- “XML Null handling options”

XML Null handling options

The XML wire format layer supports handling of null values within messages. NULL properties for XML are set for a message set only and apply to all the defined objects within the message set.

You can use the following two NULL encoding properties to represent the numeric and non-numeric encoding for NULL within the XML layer:

- *Encoding Null Num*
- *Encoding Null Non-Num*

These represent the numeric and non numeric encoding for NULL within the XML layer:

- The numeric data types are:
 - Decimal schema types: decimal, integer, negativeInteger, nonNegativeInteger, nonPositiveInteger, positiveInteger
 - Float schema types: double, float
 - Integer schema types: byte, int, long, short, unsignedByte, unsignedInt, unsignedLong, unsignedShort
- The non numeric data types are:
 - Binary schema types: base64Binary, hexBinary
 - Boolean schema types: boolean
 - DateTime schema types: date, dateTime, gDay, gMonth, gMonthDay, gYear, gYearMonth, time
 - String schema types: anyURI, duration, ENTITIES, ENTITY, ID, IDREF, IDREFS, language, Name, NCName, NMTOKEN, NMTOKENS, normalizedString, NOTATION, QName, string, token

Each of these encodings has five enumerated values:

- NULLEmpty (Default)
- NULLValue
- NULLElement
- NULLValAttr
- NULLXMLSchema

The table below defines the XML options for encoding null values.

Encoding Null Num Encoding Null Non-Num	Encoding Null Num Val Encoding Null Non-Num Val	Example XML
NULLEmpty (default)	not applicable	<child></child>
NULLValue	zzz	<child>zzz</child>
NULLElement	null	<child><null/></child>
NULLValueAttribute	not applicable	<child></child> ² <child id="X"></child> ³
NULLXMLSchema	null	<child null='true'> ¹
Notes: <ol style="list-style-type: none"> 1. The value of <i>Boolean True</i> is used. 2. This is only valid for <i>XMLElementAttrVal</i> element rendering, as specified in “XML Message rendering options” on page 61. Marking an element as being rendered in this way, and setting it to null, is equivalent to removing the attribute of the element that detailed the element’s value. 3. This is only valid for <i>XMLElementAttrIDVal</i> element rendering, as specified in “XML Message rendering options” on page 61. Marking an element as being rendered in this way, and setting it to null, is equivalent to removing the attribute of the element that detailed the element’s value, but not removing the attribute id. 		

You do not have to supply additional clarification for NULLEmpty and NULLValAttr, but if you select NULLValue, NULLAttribute, or NULLElement, you must define further values to be assigned to represent the NULL condition in the *Encoding Null Num Value* and *Encoding Null Non-Num Value* message set properties (see the table above).

XML Null value:

Unlike the TDS and CWF format, when you set the *Encoding Null Num* property to NULLValue in XML, the value is taken as a literal. A direct comparison is done with the text string, and no logical data conversion is performed.

For example, if you set the message set property *Encoding Null Num* to the value NULLValue, and you set *Encoding Null Num Val* to 0, a FLOAT value of 0.0 or a DECIMAL value of +0 does not match NULL.

If you set *Encoding Null Num* to NULLEmpty, this is equivalent to setting *Encoding Null Num* to NULLValue and *Encoding Null Num Val* to "".

XML Null element and NullValAttr:

In XML there are two conventions for storing a value:

1. It can be stored as an XML attribute with a local element or element reference property *Render* set to XMLAttribute, XMLElement, XMLElementAttrID, XMLElementAttrVal, or XMLElementAttrIDVal. For example, <element1 val="12"></element1>.
2. It can be stored as XML content with a local element or element reference property *Render* set to XMLElement. For example, <element1>12</element1>.

If you set the message set property *Encoding Null Num* to NULLElement, there is no way to represent a null value for an attribute value. If a null value is present in the tree (from ESQl or another format), an attribute with an empty string is written in the output message.

Conversely, if you have set the message set property *Encoding Null Num* or *Encoding Null Non-Num* to NULLValAttr, there is no way to represent a null value

for a value rendered as XML content. If a null value is present in the tree, when writing an empty string, an element with no character content is written out instead.

XML Null representation for Binary data:

If you use the *Encoding Null Non-Num Val* field in conjunction with a binary object in XML, you need to type the desired hex value. Do not insert the word CDATA in this field. If CDATAHex is specified in the *Encoding XML* property, CDATA rendering is used when writing the message.

Additional TDS information

This section provides additional information in relation to the TDS physical format. It has been categorized into:

- “TDS Industry standard formats”
- “Message characteristics” on page 510
- “TDS Null handling options” on page 517
- “TDS message model integrity” on page 517
- “Using regular expressions to parse data elements” on page 521

TDS Industry standard formats

WebSphere Business Integration Message Broker supports the ACORD AL3 , EDIFACT, SWIFT, TLOG, and X12 standards. The default property values for each of these standards are defined in “Default TDS Message set properties” on page 17. If you use these defaults, or override some of these defaults where necessary, you can model all these industry standard formats.

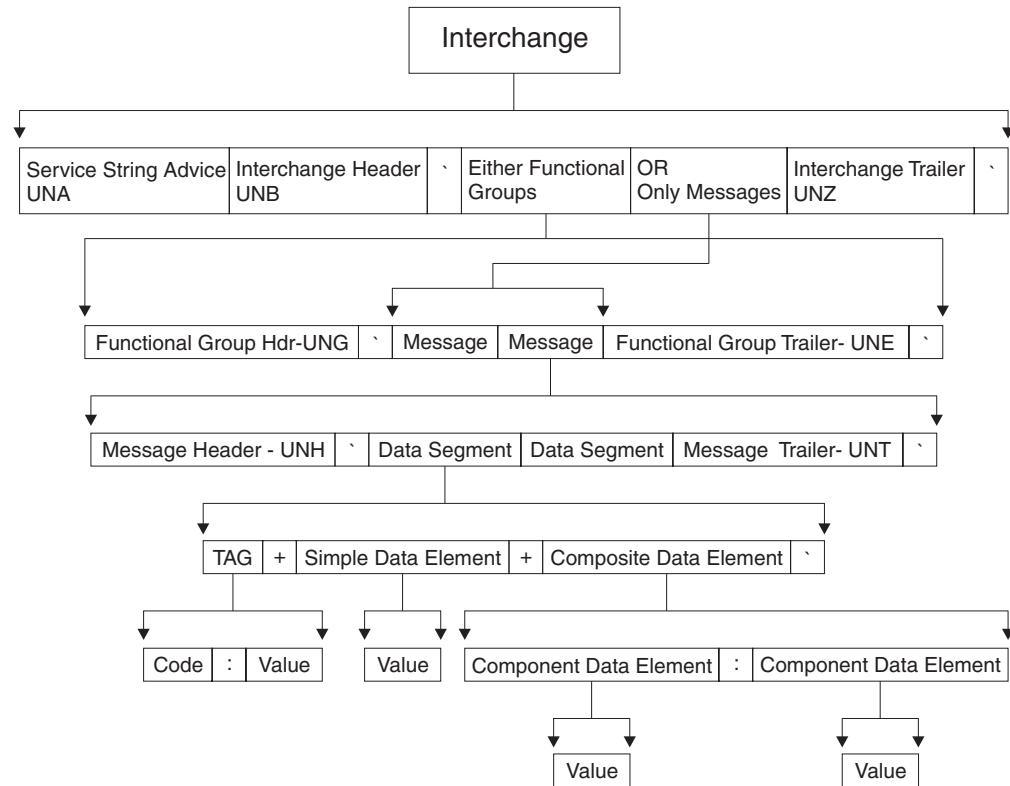
WebSphere Business Integration Message Broker supplies a number of SupportPacs, including those that supply predefined message sets for the SWIFT and EDIFACT standards. For details of these and other SupportPacs available, refer to WebSphere MQ website.

For more details about each of these industry standards see:

- “EDIFACT”
- “SWIFT” on page 507
- TLOG
- “X12” on page 508
- “ACORD AL3” on page 508

EDIFACT:

The high level structure of an EDIFACT message is shown below.



You can model the top level interchange of an EDIFACT message by setting the following properties for the complex type on which the message is based:

Composition = Sequence
Content Validation = Closed
Tag Data Separator = <EDIFACT_TAGDATA_SEP>
Data Element Separation = Tagged Delimited
Delimiter = <EDIFACT_CS>

Within an EDIFACT message, you can define the delimiters to be used in the message itself using the optional Service String Advice element. To enable this element to be recognized as an EDIFACT Service String, you must set the element property *Interpret Element Value* to EDIFACT Service String. You must also set the delimiter values to the mnemonic values that are defaulted when you set the *Message Standard* property to EDIFACT.

SWIFT:

The high-level block structure of a SWIFT message is shown in the table below.

SWIFT message high level block structure

Block name	Format
Basic header	{1:...}
Application header	{2:...}
User header	{3:...}
Text	{4:...}
Trailer	{5:...}

When they are concatenated in a message, the blocks appear as:
{1:...}{2:...}{3:...}{4:...}{5:...}

You can model this setting the following type properties for the message:

```
Data Element Separation = Tagged Delimited
Group Indicator = {
Delimiter = }{
Group Terminator = }
Tag Data Separator = :
```

Each block is modeled as a complex element with element *Tag* property values of 1,2,3,4, and 5 respectively.

The text body of the message has the following format:

```
{4:
:20:X
:32A:940930USD1,
.....
:72:/A/
-}
```

You can model the complex type of the Text body by setting the following type properties:

```
Data Element Separation = Tagged Delimited
Group Indicator = <CR><LF>:
Delimiter = <CR><LF>:
Group Terminator = <CR><LF>-
Tag Data Separator = :
```

The *Tag* property of the elements within the body has values of 20, 32A, 72, and so on.

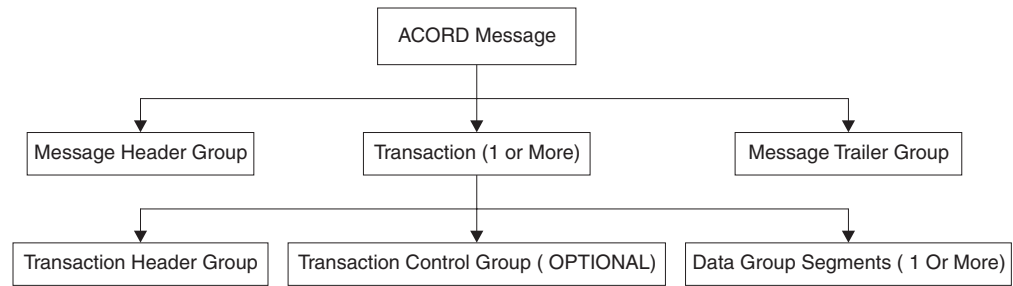
If you need to validate message content to the level of SWIFT Gold Certification, you can request the MRM parser to perform additional message validation. See [Validating messages](#) for details of how to set this up using MQInput and Compute node properties.

X12:

If you are working with X12 messages, you can define the delimiters to be used in the message itself using the mandatory Interchange Control Header element. To enable this element to be recognized as an X12 Service String, you must set the element property *Interpret Element Value* to X12 Service String. You must also set the delimiter values to the mnemonic values defaulted by setting the *Message Standard* property to X12.

ACORD AL3:

The basic structure of an ACORD AL3 message is shown below.



Each group with an ACORD AL3 message has a header consisting of a one-digit number, three letters, plus a three-digit total length count. These first seven characters can be modeled as a tag. The data within the headers is fixed length. Therefore the header type used for the overall message can be modeled as follows:

Data Element Separation = Tagged Fixed Length
Length of Tag = 7

The Transaction Group contains other groups, and is therefore modeled in the same way as the overall message. The Message Header Group and the Message Trailer group just consist of fixed length elements, therefore the type used can be modeled as:

Data Element Separation = Fixed Length

There are two *Data Element Separation* methods particularly suited to handling ACORD AL3 messages:

- Fixed Length AL3 supports basic handling of ACORD AL3 messages, including situations where the message groups conform to a different version of the ACORD AL3 standard. This is deprecated and will be removed in a future version of the product and an alternative will be provided.
- Tagged Encoded Length supports handling of more sophisticated situations, including messages containing message groups unknown to the message dictionary.

The following sections describe their use:

- “Using Fixed Length AL3”
- “Using Tagged Encoded Length to support re-versioning” on page 510

Using Fixed Length AL3:

This is deprecated and will be removed in a future version of the product and an alternative will be provided.

You can select the value Fixed Length AL3 for the *Data Element Separation* property for complex types within a message that conforms to the ACORD AL3 standard. This allows different versions of the ACORD AL3 standard to be supported using the same message set. This value is similar to the value Fixed Length except for the following:

- A question mark (?) in the left-most position of an element means that it is skipped.
- A sequence of question marks is inserted for all missing optional elements.
- Unused trailing optional elements are truncated.
- Any <CR><LF> after the last element is ignored.

- The length field is extracted on input (and *not* put to the tree), and automatically recalculated on output.

Using Tagged Encoded Length to support re-versioning:

The incoming message might contain a group that is no longer in use within the current ACORD AL3 standards, and has therefore been deleted from the later version of the standards. Similarly, the incoming bit stream might be from a later version of the ACORD AL3 standards, and might contain a new group that was not defined in earlier versions.

In order to correctly parse this self defining tag, the TDS parser needs to know the length of the group it is parsing and skip to the end of all data associated with that self defining tag.

Use the *Data Element Separation* method Tagged Encoded Length to handle these situations. You will also need to set these properties:

- Length of Tag or Tag Data Separator, so that the TDS parser knows where tags end.
- Length of Encoded Length, so that the TDS parser knows the size of the length field.
- Extra Chars in Encoded Length, are used to indicate to the TDS parser how many characters, apart from the data itself, are counted in the encoded length field.

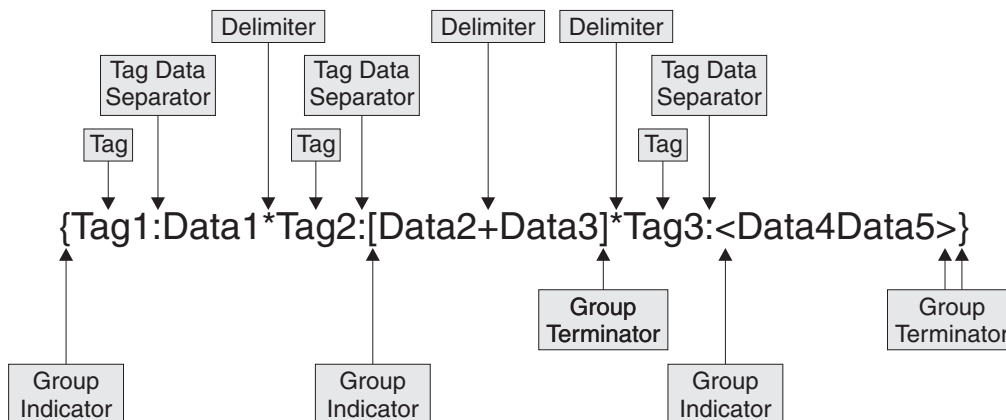
Message characteristics

There are a number of features of text string messages that are common across many formats. The following sections give an overview of the main features that are supported by the TDS wire format:

- The text strings in the message can have a tag or a label preceding the data value. The tag is a string that uniquely identifies the data value. The TDS format allows you to associate a tag with each element when you define the element in the workbench.
- The message can contain various special characters or strings in addition to the tags and text string data values. The TDS format supports a number of different types of special characters or strings. Some messages have a special character or string that separates each data value from the next. In the TDS format this is known as a delimiter. In formats that have a tag before each data value, the tag can be separated from its data value by a special character or string. In the TDS format this is known as a tag data separator.
- A message can be split into a number of substructures in a similar manner to a COBOL or C structure. You can model each of these substructures separately by defining complex types or elements for each one. Complex types and elements are described in Message model objects. A substructure can have a special character or string that indicates its start within the data. This is known in the TDS format as a group indicator. A substructure can also have a special character or string that indicates its end in the data. In the TDS format, this is known as a group terminator. A group indicator and group terminator can also be defined for the whole message. Group indicators and group terminators are optional for the message and each substructure.
- Some text strings within a message can be of fixed length, so a delimiter between each data value is not necessary. This is supported by the TDS format. If you use a fixed length tag, a tagged data separator is not required.

- The TDS property that controls the way text strings are separated is *Data Element Separation*. It has several options that let you choose, for example, if tags are used, if strings lengths are fixed or variable, and what types of text strings are permitted. See “Specifying data element separation methods to model a message.”
- The substructures within a message can use different types of *Data Element Separation* and use different special characters. Therefore the TDS format allows you to define different types of data element separation and special characters for each complex type within the message.
- If you use the Use Data Pattern method of *Data Element Separation*, you can use regular expressions to identify parts of the message data to be assigned to sub-fields. This is done by setting the regular expression in the Data Pattern property. See “Using regular expressions to parse data elements” on page 521 for further details.

The figure below illustrates the tags and special characters in a TDS message, showing an example data message with each of its components labeled.



- At the top level, each data value has a tag associated with it, each tag is separated from its data value using a tag data separator of colon (:), and the data values are separated from each other using the asterisk delimiter (*).
- The group indicator for the message is the left brace ({) and the group terminator is the right brace (}).
- The data values Data2 and Data3 are in a substructure in which there are no tags, and each data element is separated from the next using the delimiter plus (+). The group indicator for this substructure is the left bracket ([) and the group terminator is the right bracket (]).
- The data values Data4 and Data5 are in a substructure in which the values are fixed length, and are therefore not separated by a delimiter. The group indicator for this substructure is the less than symbol (<) and the group terminator is the greater than symbol (>).

The following sections describe data element separation and the special characters in more detail:

- “Specifying data element separation methods to model a message”
- “Specifying special characters to model a message” on page 514

Specifying data element separation methods to model a message:

Elements of data in a TDS message are identified according to the data element separation method that you must specify for the *Data Element Separation* property for a complex type. Depending on the value that you have set for *Data Element Separation*, the properties *Tag Data Separator* and *Delimiter* (for a message set and a complex type) might also be required to identify each element.

The methods that you can specify for each complex type are described below. The examples given are all based on a complex type that contains three elements of type STRING. The *Tag Data Separator*, where used, is the colon (:), and the *Delimiter*, where used, is the asterisk (*).

Tagged Delimited

Each data value is preceded by a tag that is specified as an element property. If the tag has an associated *Length of Tag*, indicating that the tag has a fixed length, each data value follows immediately after the tag. If the tag is not specified as fixed length, the tag is separated from the next element by a *Tag Data Separator*. Each data value is separated from the next by a *Delimiter*. There is no *Delimiter* after the last element in the complex type.

The following example shows tags of fixed length:

```
tag1data1*tag2data2*tag3data3
```

The following example shows tags of variable length:

```
tag1:data1*tag11:data2*tag111:data3
```

Tagged Fixed Length

This method is similar to Tagged Delimited, but the data values are always fixed length. Therefore, no delimiter is required after each data value. The tags themselves can be fixed length or variable length, depending on the setting of *Tag Data Separator* and *Length of Tag*.

The following example shows tags of fixed length:

```
tag1data1tag2data2tag3data3
```

The following example shows tags of variable length:

```
tag1:data1tag11:data2tag111:data3
```

Tagged Encoded Length

This method has a tag and a length field before the data. It indicates to the parser that following each tag in the bit stream there is data defining the length of data to be associated with that tag. You must set the *Length of Encoded Length* parameter. If the value in *Length of Encoded Length* includes extra characters, you must also set the *Extra Chars in Encoded Length* parameter.

The following example shows a tag of fixed length of four characters (*Length of Tag* has been set to four), a three-character length field (*Length of Encoded Length* has been set to three), and several characters of data. *Extra Chars in Encoded Length* has been set to zero:

```
tagA007dataAAAtagB006dataBBtagC009dataCCCCC
```

Given the bit stream above, the parser finds the tag "tagA" and extracts the length value 7. Because *Extra Chars in Encoded Length* is set to zero, the next seven (7 - 0) characters are the data. Then follow the characters for the next tag "tagB" and the length value of 6, and so on for tag "tagC". In each case in this example, the value in the length field is exactly the length of data.

The following example shows tags with a fixed length of four characters (*Length of Tag* has been set to four), a three-character length field (*Length of Encoded Length* has been set to three), and several characters of data. *Extra Chars in Encoded Length* has been set to three (because in this example the length field value includes the three-character length field as well as the data field):

```
tagA012dataAAAAAtagB010dataBBBtagC016dataCCCCCCCC
```

Given the bit stream above, after "tagA" the parser extracts the length value 12. But because *Extra Chars in Encoded Length* is set to three, only the next nine (12 - 3) characters are the data. Then follow the characters for "tagB" and length value 10, and so on. In each case in this example, the value in the length field is three more than the actual length of data.

All Elements Delimited

The data values have no tag, but each data value is separated from the next by a delimiter.

The following example shows this:

```
data1*data2*data3
```

Variable Length Elements Delimited

If a data element is fixed length, the next data value follows immediately after it. If the data element is variable length, the next data value is separated from it by the delimiter. There are no tags.

The following example shows element 2 as fixed length, and elements 1 and 3 as variable length:

```
data1*data2data3
```

Use Data Pattern

The data associated with each element is determined by the parser matching the data with the regular expression in the *Data Pattern* property for that element. The TDS parser uses the regular expression in the *Data Pattern* to:

- Determine the length of data to associate with each element.
- Determine if, in the case of a repeating element, another occurrence of an element is present in the bit stream.
- Determine the presence (if the pattern is matched) or absence (if the pattern is not matched) of an element in the bit stream.

There are no delimiters or tags, other than those coded as part of the regular expression patterns. See "Regular expression syntax" on page 523 for an explanation of how pattern matching works.

The following example shows three elements, each having the regular expression *Data Pattern* shown:

```
First Data Pattern = [A-Z]{1,3}
```

```
Second Data Pattern = [0-9]+
```

```
Third Data Pattern = [a-z]*
```

```
Message data = 'DT31758934information for you'
```

```
First element data: 'DT'
```

```
Second element data: '31758934'
```

```
Third element data: 'information'
```

The first *Data Pattern* means "from one to three characters in the range A to Z", the second means "one or more characters in the range 0 to 9", and the

third means "zero or more characters in the range a to z". Notice how each element's data was terminated by the first character that did not match the element's *Data Pattern*.

Fixed Length

All elements are fixed length, and each data value immediately follows the next with no delimiter. There are no tags.

The following example shows this:

```
data1data2data3
```

Fixed Length AL3

This method is the same as Fixed Length, but it also notifies the parser to implement a number of rules in relation to missing elements, length encoding, and versioning that are predefined in the ACORD AL3 standard.

Undefined

This value is set automatically when you set the *Type Composition* property of a complex type to Message, and you cannot set it to any other value. You are also unable to set values for the TDS Type properties *Group Indicator*, *Group Terminator*, *Tag Data Separator*, *Length of Tag*, and *Delimiter*.

If you set the *Data Element Separation* method to Undefined, you must not set the *Type Composition* property to Empty, Choice, Unordered Set, Ordered Set, Sequence, or Simple Unordered Set. If you do, you will be unable to check in the type.

For more information about *Type Composition* set to Message, see Multipart messages.

Specifying special characters to model a message:

You can specify a number of different types of special character in the workbench. You can specify special character values for message sets, types, and type members. The values that you set for a type override the corresponding values set for the message set in which it is defined.

You can specify a special character value in one of the following ways:

1. As a literal string of one or more characters.
2. As a mnemonic value.
3. As a combination of both mnemonics and literals.

The types of special character are described in the table below.

Special character type	Description	Set as a property of...
Group Indicator	This is a string that indicates the start of a group or complex type within a message	Message set, type
Group Terminator	This is a string that indicates that the end of a group or complex type within a message	Message set, type
Tag Data Separator	This is the string that is used to separate a tag from its data.	Message set, type
Delimiter	This is the string used to separate data elements from one another	Message set, type
Repeating Element Delimiter	This is the string used to separate repeating data elements from one another	Type member

Therefore, if you create a complex type and set *Data Element Separation* property to Tagged Delimited, the *Group Indicator* property to left brace ({}), the *Group Terminator* to right brace ({}), the *Tag Data Separator* to colon (:), and the *Delimiter* to asterisk (*), the bit stream has the following format:

```
{tag1:data1*tag2:data2*tag3:data3}
```

In some message formats, a special character is specified before each element or after each element, as shown in the following two examples:

```
:data1:data2:data3
```

```
data1:data2:data3:
```

You can model these formats by using a combination of the *Data Element Separation* method, the *Delimiter* value, the *Group Indicator* value, and the *Group Terminator* value.

For the first example, specify *Data Element Separation* as All Elements Delimited, *Delimiter* as colon (:), and *Group Indicator* as colon (:).

For the second example, specify *Data Element Separation* as All Elements Delimited, *Delimiter* as colon (:), and *Group Terminator* as colon (:).

Using mnemonics as special characters:

A mnemonic is a tag delimited by < and >. The broker translates the mnemonic to obtain the actual value of the special character. There are two types of mnemonic:

- Control code mnemonics which map to the common non-printing characters. These are mapped using the local codepage for your system. This is typically an ASCII codepage on distributed platforms and an EBCDIC codepage on other platforms.

This means that characters are generally mapped to the 'expected' values for your system. This does depend on your codepage setting and if in doubt you should refer to your system documentation. In the event that a specific mnemonic is not mapped to the value you need then you can use the explicit <U+xxxx> representation described below.

- Message mnemonics for use with specific industry message standards such as X12.

These are mapped according to their associated message standard. Each mnemonic has a default mapping, but in message standards such as EDIFACT and X12 this default can be overridden by a 'service string' specified in the message itself.

Mnemonics can be specified in one of two ways:

1. <Mnemonic_Name> where Mnemonic_Name can comprise alphanumeric characters and underscore (_) characters.
2. <U+xxxx> where xxxx are hexadecimal numbers. The delimiter is interpreted as the Unicode character that corresponds to the hexadecimal value.

In addition to special characters, mnemonics can also be used in the message set properties Decimal Point, Escape Character, and Reserved Characters.

For more details about the supported mnemonics, see “TDS Mnemonics” on page 16.

TDS Mnemonics:

The Tagged/Delimited String Format (TDS) uses mnemonics for a number of properties for a message set, complex type, or both. These TDS mnemonics and their associated properties are listed in the table below.

Mnemonic string	Meaning	Default value	Associated property
<EDIFACT_CS>	Component data separator in EDIFACT	:	Message set and type, <i>Delimiter</i>
<EDIFACT_DS>	Data element separator in EDIFACT	+	Message set and type, <i>Delimiter</i>
<EDIFACT_TAGDATA_SEP>	Tag data separator in EDIFACT This is overridden with the same value as that which overrides <EDIFACT_DS>	+	Message set and type, <i>Tag Data Separator</i>
<EDIFACT_DEC_NOTATION>	Decimal notation in EDIFACT	.	Message set, <i>Decimal Point</i>
<EDIFACT_ESC_CHAR>	Escape character in EDIFACT	?	Message set, <i>Escape Character</i>
<EDIFACT_GROUP_TERM>	Tag terminator in EDIFACT	'	Message set, <i>Tag Terminator</i>
<X12_GROUP_TERM>	Tag terminator in X12	!	Message set level, <i>Tag Terminator</i>
<X12_DS>	Data element separator for X12	*	Message set and type, <i>Delimiter</i>
<X12_CS>	Component data element separator for X12	:	Message set and type, <i>Delimiter</i>
<LT>	Represents the less than character, which is a reserved character	<	
<GT>	Represents the greater than character, which is a reserved character	>	
<CR>	Represents the carriage return character	%X0D	
<LF>	Represents the line feed character	%X0A	

Mnemonics are also supported for the following control characters:

<ACK> (x'06')	<BEL> (x'07')	<BS> (x'08')	<CAN> (x'18')
<CR> (x'0D')	<DC1> (x'11')	<DC2> (x'12')	<DC3> (x'13')
<DC4> (x'14')	<DLE> (x'10')	 (x'19')	<ENQ> (x'05')
<EOT> (x'04')	<ESC> (x'1B')	<ETB> (x'17')	<ETX> (x'03')
<FF> (x'0C')	<FS> (x'1C')	<GS> (x'1D')	<GT> (x'3E')
<HT> (x'09')	<LF> (x'0A')	<LT> (x'3C')	<NAK> (x'15')
<NUL> (x'00')	<RS> (x'1E')	<SI> (x'0F')	<SO> (x'0E')
<SOH> (x'01')	<SP> (x'20')	<STX> (x'02')	<SUB> (x'1A')
<SYN> (x'16')	<US> (x'1F')	<VT> (x'0B')	

You can enter a mnemonic in the form <U+xxxx> where xxxx are hexadecimal numbers up to a value of FFFF. These numbers represent a Unicode character, not a

character in your local code page or the code page in which message data is formatted. None of the characters in this structure are case sensitive. Do not enclose spaces inside the angle brackets.

TDS Null handling options

TDS supports handling of null values within messages. You can use the message set property *Boolean Null Representation* to specify the value to be used for Boolean Null representation. You can use the object properties *Encoding Null* and *Encoding Null Value* to control how null handling is represented for individual objects.

You can select the *Encoding Null* property from the three enumerated values `NULLPadFill`, `NULLLogicalValue`, and `NULLLiteralValue`:

- You should use the `NULLPadFill` option only for fixed length objects. If you select this option for an object of simple type `dateTime`, a null `dateTime` is written out, which is an empty tag with a delimiter. (This is equivalent to selecting `NULLLiteralValue`, with the *Encoding Null Value* property set to the empty string `""`.) If you select this option for an object of another simple type, the object is filled with the value specified by the *Padding Character* property. If you select this option, the *Encoding Null Value* property is disabled.

If you use this option for a variable length object, the parser does not know how many padding characters to write out, so it does not write any. Instead, the parser writes an explicit null, with tag and delimiter but no data value. For example:

```
tag1:,
```

is written out, where `tag1` is the tag for the variable length element with `NULLPadFill` set, `":"` is the tag data separator, and `","` is the delimiter.

- If you select the `NULLLogicalValue` option, the value entered for the *Encoding Null Value* property is converted to its logical value. For writing, the logical value is written in the same way as any other value. For parsing, the converted logical value is compared against the converted message data.
- If you select the `NULLLiteralValue` option, the value entered for the *Encoding Null Value* property is directly substituted as if it were a string value. The value is case insensitive. For fixed length objects, the literal value must be no longer than the length of the object.

If the literal value is shorter, the *Encoding Null Value* is padded (using *Padding Character*) on output. On input, if the `NULLLiteralValue`'s length does not match the *Length* field, you should set the message set level *Trim Fix Len String* property so that padded nulls are correctly parsed.

The use of the *Encoding Null Value* property is dependent on the value that you select for the *Encoding Null* property described above. Null values are not defined for binary types. The properties *Encoding Null* and *Encoding Null Value* are therefore not set for binary types.

TDS message model integrity

When you use the TDS wire format, you must conform to a number of rules that apply to the setting of values of properties. These rules are checked any time the project is saved. If an inconsistency is found, the error is displayed in the task list of the workbench.

The following sections cover the rules for TDS wire format properties:

- “General rules: TDS message model integrity” on page 518

- “Restrictions for nesting complex types” on page 519
- “Omission and truncation of elements” on page 520

General rules: TDS message model integrity:

This section describes the general rules for each value that you can set for the *Data Element Separation* property of a type.

Tagged Delimited

- The *Tag* property for every simple child element must contain a non-empty value.

Tagged Encoded Length

- The *Tag* property for every simple child element must contain a non-empty value.
- The *Length Of Encoded Length* property must contain a positive integer greater than zero.

Variable Elements Delimited

- The *Delimiter* property must contain a non-empty value.

Use Data Pattern

- Each simple element that is a child of the complex type must have a regular expression specified for *Data Pattern*. See “Regular expression syntax” on page 523.

All Elements Delimited

- The *Delimiter* property must contain a non-empty value.

Fixed Length

- All simple child elements must specify a length, unless their data type is boolean (or derived from boolean).
- All compound child elements must specify a length, unless their data type is boolean (or derived from boolean).
- The length can be specified using either the *Length* property, or the *Length Value Of member* property.

Fixed Length AL3

- All complex child elements with a non-boolean compound element and non-boolean simple child elements must have either a nonzero value in their *Length* property, or a non-empty value for their *Length Value Of* type member property.

Tagged Fixed Length

- All complex child elements with a non-boolean compound element and non-boolean simple child elements must have either a nonzero value in their *Length* property or a non-empty value for their *Length Value Of* type member property.
- The *Tag* property for each and every simple child element must contain a non-empty value.

The following rules also apply:

- If you have set the parent *Type Composition* to Choice, and the parent *Data Element Separation* property to Variable Elements Delimited, All Elements Delimited, Fixed Length, or Fixed Length AL3:
 - You must not set the *Type Composition* to Message for any child complex types.

- You must not set the *Data Element Separation* method to Tagged Delimited or Tagged Fixed Length for any child complex types.

If you do so, the message set will not deploy successfully.

- If you have set the type's *Data Element Separation* property to Fixed Length, Fixed Length AL3, or Tagged Fixed Length, you must set either the *Length* or *Length Value Of* property for all simple elements under this parent, and also for all complex elements with a simple content and compound elements.
- For a Choice in a fixed length environment (*Data Element Separation* set to Fixed Length, Tagged Fixed Length, or Fixed Length AL3), length references are not valid, and element lengths should be used.
- Elements specified in a *Length Value Of* property must be simple elements of type INTEGER, they must exist in the same structure as the referring element, and they must appear before the referring element in that structure.
- Complex types with simple content and Compound elements must have an empty *Length Value Of* type member property. This is because the *Length Value Of* element would occur after the referring element in the parent structure, which is disallowed by the previous rule.
- Complex types with simple content cannot have a separation type of Use Data Pattern.
- Compound elements cannot have a separation type of Use Data Pattern.
- Regardless of the setting of the type's *Data Element Separation* property, if the type of a simple element is BINARY, you must set either the *Length* or *Length Value Of* property.
- For fixed length elements, the *Justification* property must be set to something other than Not Applicable, and the *Padding Character* property cannot be an empty value.
- If any element within a message has its *Interpret Element Value* property set to Message Key, the *Message Key* property must be set for all messages within the message set.
- If you have set the *Repeat* property in the type member to Yes, you must set a value for the *Max Occurs* property in the following two situations:
 - If you have defined an element as a member of a complex type that has the property *Data Element Separation* set to Fixed Length.
 - If you have defined a fixed length element as a member of a complex type that has the property *Data Element Separation* set to Variable Elements Delimited.

When it is invoked by the broker to interpret an input message, the parser assumes that the number of occurrences of the element is equal to the value that you set for *Max Occurs*. When the parser constructs an output message, if there are fewer elements than the value of *Max Occurs*, the missing elements are inserted with default values.

Restrictions for nesting complex types:

If you include a group within another group or complex type, the *Data Element Separation* property for the nested group must be compatible with the *Data Element Separation* property of the parent group or complex type. For example, you cannot set the parent property to Fixed Length and the child property to Tagged Delimited, because the length of the Tagged Delimited structure would not be known, and would therefore conflict with the parent definition. If groups are nested to three or more levels, the *Data Element Separation* property for each nested group must be compatible with all of its parent groups.

The rules for compatibility are listed in the table of permitted options for nested complex types shown below.

	Parent				
Child	Tagged Delimited, Tagged Encoded Length	All Elements Delimited, Variable Elements Delimited	Fixed Length, Fixed Length AL3	Tagged Fixed Length	Use Data Pattern
Tagged Delimited, Tagged Encoded Length	Allowed	Allowed	Not allowed	Not allowed	Allowed
All Elements Delimited, Variable Elements Delimited	Allowed	Allowed	Not allowed	Not allowed	Allowed
Fixed Length, Fixed Length AL3	Allowed	Allowed	Allowed	Allowed	Allowed
Tagged Fixed Length	Allowed	Allowed	Not allowed ¹	Allowed	Allowed
Use Data Pattern	Allowed	Allowed	Allowed	Allowed	Allowed
Note: 1. Tagged Fixed Length cannot exist at the inner level if any outer level has a <i>Data Element Separation</i> method of Fixed Length or Fixed Length AL3. This is because an item of Tagged Fixed Length can repeat a variable number of times. Fixed Length and Fixed Length AL3 are parsed by moving a set number of bytes: with a variable number of repeats, it is not possible to calculate the number of bytes that need to be parsed.					

Omission and truncation of elements:

The omission and truncation of elements is dependent on the setting of the property *Suppress Absent Element Delimiters*. A description of this can be found in “Complex type TDS properties” on page 63, “Global group TDS properties” on page 68, or “Local group TDS properties” on page 73.

If you have created a message in which some elements are optional, an input message might not contain all defined elements. If the elements are in a complex type that you have defined with the *Data Element Separation* property of the type set to All Elements Delimited or Variable Elements Delimited (in which the elements are separated by a delimiter and have no tag), any elements that are missing from the end of the complex type must be indicated by the application that creates the message in one of two ways. These both provide techniques to avoid unnecessarily long sequences of delimiters, and to preserve consistent representation of missing elements.

1. If you have set the *Delimiter* property for the complex type to a value that does not match the value that you have set for the *Delimiter* property for any of the complex type’s parent types, the elements at the end of the message can be indicated by the occurrence of a *Delimiter* of one of its parents after the last actual element in the complex type data.

This is known as the truncation method, in which missing elements are treated as not expected, and both data and delimiters are omitted in the bit stream.

For example, you define a complex element C that has four optional elements. You set the *Delimiter* property to the character plus (+). You define complex element P, and set the *Delimiter* property of P to asterisk (*). You add three elements to P, the first is a string, the second is complex element C, and the third is a string.

When a particular instance of the message is received by the broker, all the elements of P are present, but only the first two elements of C are present. The data in the message appears as follows if the truncation method is used (where P_n are the values of the elements of P and C_n the values of the elements of C):

P1*C1+C2*P3

When the parser encounters the second asterisk delimiter, it determines that the last two elements of complex element C are not present, and the next element is the third element of P.

You can use truncation successfully only when both omission and truncation cause the parser to exhibit the same behavior, unless the elements truncated are fixed length.

2. If the *Delimiter* of the complex type matches that of one of its parents, the truncation method cannot be used. This is because the parser cannot determine whether the delimiter after the last element is for the current complex type, or for one of its parents. Therefore a delimiter must be included in the message data for each missing element to ensure that the parser can match the elements with the model.

This is known as the omission method, in which missing simple elements are represented by an empty sequence of characters between two delimiters.

For example, you define P and C as in the previous example, but set the *Delimiter* property for P to plus (+). When the same message is received by the broker (all elements of P are present, the first two elements of C are present), the data in the message appears as follows:

P1+C1+C2++P3

Two delimiter characters have been inserted in the message data for the missing elements of complex element C. If the truncation method had been used, the parser would have interpreted the data value P3 as the value of the third element of complex element C and not the third element of complex element P.

Using regular expressions to parse data elements

If your input messages can contain optional sub-fields whose presence or absence can only be determined by examining the actual value of the data (for example an optional field of numeric digits followed by one or more alphabetic characters) you need to use the *Data Element Separation* method Use Data Pattern. This is particularly relevant to messages conforming to the SWIFT industry standard. To use this method, you must provide regular expressions to identify those portions of an input message that are to be associated with sub-fields. You need to provide a regular expression value for the *Data Pattern* property of each child of the complex type.

When parsing, data is matched in turn to each child of the complex type. The parser does this by using the regular expression for the child to determine the number of characters from the message that apply for that child. This number of characters is the length of the longest string, starting from the current position in the message, that matches the regular expression. If the longest string that matches the regular expression is of length zero, the element is present in the message, and

the empty string is used for the value. If no string matches the regular expression, the element is not present. This might cause a subsequent validation error if the element is required.

After the number of characters from the input message has been determined, normal data conversion, or further parsing in the case of a complex element, is performed on the text of the input message to assign values to elements. This might lead to data overrun or underrun errors if the length identified by the pattern is not appropriate for the definition of the child.

“Regular expression syntax” on page 523 explains the full syntax rules and how to apply them, but the table below gives a few simple examples of parsing using data patterns. A more complex example appears after the table.

Input message	Data Pattern	Value matched
"123456ABC"	[0-9]*	"123456"
"123"	[A-Z]*	" "
"123"	[A-Z]+	Not present
"ABCD123"	[A-Z]{1,3} first field [A-Z]{2,4} second field	"ABC" - first field (the longest string matching the pattern) Not present - second field (minimum length of two alphabetic characters is not present)
"ABCDEFGHJIJ1234"	[A-Z]{1,3} first field, repeat [0-9]+ second field	"ABC" - first field [1] "DEF" - first field [2] "GHI" - first field [3] "J" - first field [4] "1234" - second field (the repeating field is terminated when the data "1234" no longer matches the data pattern specified for the first field.)

The example below shows three-field pattern matching.

```

Message definition:
Complex type: Data Element Separation=Use Data Pattern
Field1: xsd:string minOccurs=1, maxOccurs=1, Length=5, Pad=SPACE,
      Data Pattern=".{5}"
Field2: xsd:int minOccurs=0, maxOccurs=1,
      Data Pattern="[0-9]{0,6}"
Field3: xsd:string minOccurs=1, maxOccurs=1, minLength=3, maxLength=4,
      Data Pattern="[A-Z][A-Za-z0-9]{2,3}"

Input1: "ABCDE123F12"
Result1: Field1="ABCDE", Field2="123", Field3="F12"

Input2: "ABCDEF12"
Result2: Field1="ABCDE", Field2=not present, Field3="F12"

Input3: "ABCDE123456XXXX"
Result3: Field1="ABCDE", Field2="123456", Field3="XXXX"

Input4: "ABCDE1234567"
Result4: Field1="ABCDE", Field2="123456", Field3=not present,
      which causes an exception if validation is enabled. One
      character ("7") remains unassigned to any element, which
      also causes an exception.

```

In the case of a repeating child, instances of the child are parsed for as many times as the pattern is matched. This is applied even if *Max Occurs* is specified for the repeating element and the number of occurrences exceeds the upper bound. Therefore some terminating condition must be determinable from the regular expression pattern for the element. The table above includes an example of a repeating element.

When parsing, the data from the input message that matches the *Data Pattern* and is assigned to an element is *not* further scanned for delimiters of a higher level complex type. This behavior is similar to that of *Data Element Separation* method Fixed Length. However, you can code a regular expression that will match data to one of a number of possible delimiters.

When writing, if a length is specified for a child, the value is padded as appropriate to that length. This behavior is similar to that of *Data Element Separation* method Variable Elements Delimited, but without delimiters.

If the message includes a complex type that has *Composition* set to Choice, you can set the *Data Element Separation* method to Use Data Pattern. In this case, the *Data Pattern* values of the children are used to resolve the choice. Starting with the first child, the first pattern to provide a match determines which child is present. Therefore the order of children in a choice might be important.

A complex type can contain repeating children with *Max Occurs* unbounded. Length, and other associated properties such as justification and padding, can optionally be specified for the children.

See “TDS message model integrity” on page 517 for rules that you must follow when using the *Data Element Separation* method Use Data Pattern, and refer to “Combinations of Composition and Content Validation” on page 123 for details of valid settings of *Composition* and *Content Validation*.

Regular expression syntax:

A regular expression is a coded string. It defines a set of strings that match the expression. A regular expression can be made up of one or more branches (choices), each of which can be a string made up of characters, character classes, or parenthesized expressions with modifiers to specify repetition rules.

The regular expression syntax supported is a subset of XML Schema regular expressions. For the full syntax, see Appendix F in XML Schema Part 2: Datatypes that can be found on the World Wide Web Consortium (W3C) Web site.

The following table lists the supported regular expression syntax elements.

Metacharacter	Meaning
\	escape
.	any single character
*	preceding character 0 or more times
+	preceding character 1 or more times
?	preceding character 0 or 1 time
{...}	occurrences of preceding ¹
[...]	match one of the class contained
[^...]	match one of the class not contained ¹
(...)	group the expressions ¹
	match either preceding or following
Escape sequence	Meaning
\n	new line
\r	carriage return
\t	tab
\e	escape
Class code	Meaning
\d	digit [0-9]
\D	non-digit [^0-9] ²
\s	whitespace [\t\n\r]
\S	non-whitespace [^ \t\n\r] ²
\p{L}	all letters ³
\p{N}	all numbers, similar to \d ⁴
[\p{N}\p{L}]	all numbers and all letters , similar to \w ⁴
\P{L}	not letters, equivalent to [^\p{L}]
\P{N}	not numbers, equivalent to [^\p{N}]
Range	Meaning
{n}	exactly n times
{n,}	at least n times
{n,m}	at least n but no more than m times
{0,m}	zero to m times

Notes:

1. The ellipsis (...) is used to indicate anything inside the { }, or [], or () characters.
2. The caret (^) means "not" when inside the [] characters.
3. Consult Appendix F of the document XML Schema Part 2: Datatypes for other characters that can be used in place of L and N.
4. Consult Appendix F of the document XML Schema Part 2: Datatypes for the precise differences.

The following table gives some examples of the syntax rules for regular expression syntax. See "Using regular expressions to parse data elements" on page 521 for some examples of their use.

Regular expression data pattern	Meaning
a	Match character "a"
.	Match any one character
a+	Match a string of one or more "a"
a*	Match a string of zero or more "a"
a?	Match zero or one "a"
a{3}	Match a string of exactly three "a", that is "aaa"
a{3,}	Match a string of three or more "a"
a{2,4}	Match a string with a minimum of two and a maximum of four occurrences of "a"
[abc]	Match any one of the characters "a", "b", or "c"
[a-zA-Z]	Match any one character in the range "a" to "z", or in the range "A" to "Z". Note that the range of characters matched is based on the Unicodes of the characters specified.
[^abc]	Match any character except one of "a", "b", or "c"
(ab)+	Match one or more repetitions of the string "ab"
(ab) (cd)	Match either of the strings "ab" or "cd"

Using multiple delimiters:

If you set *Data Element Separation* to the method Use Data Pattern, messages in which fields are delimited by one of a set of characters or strings can be parsed.

For example, consider a simple message with two numeric fields that can have either of the characters ';' or '/' delimiting them. There are two approaches that you could use:

1. Model the delimiter as a data element which is added to the message tree. If the message is rewritten, it looks like the input message.

Consider this model:

```

Composition = Sequence
Data Element Separation = Use Data Pattern
  FieldA   Data Pattern = [0-9]*
  Delim    Data Pattern = [;/] optionally with a default value.
  FieldB   Data Pattern = [0-9]*

```

After parsing, the elements FieldA and FieldB each contain any number of the digits 0 to 9, and the element Delim contains either ";" or "/".

2. Recognize the delimiter as a delimiter, which is not added to the tree. If the message is rewritten, a preferred delimiter (as specified in the model) is used.

Consider this model:

```

Composition = Choice
Data Element Separation = Use Data Pattern
  SubType1 Data Pattern = [0-9]*;[0-9]*
  (Composition = Sequence
   Data Element Separation = All Elements Delimited
   Delimiter = ';')
  FieldA
  FieldB
  SubType2 Data Pattern = [0-9]*/[0-9]*
  (Composition = Sequence
   Data Element Separation = All Elements Delimited
   Delimiter = '/')
  FieldA
  FieldB

```

The regular expressions differentiate between the two options that can occur in the message, which are then parsed as a normal delimited structure. After parsing, the elements FieldA and FieldB each contain any number of the digits 0 to 9. The delimiter found in the input message is not saved in an element.

You could refine this approach by using different names for the children, or elements for SubType1 and SubType2, to provide the knowledge of which delimiter is used, or to control which delimiter is included in the output message.

Using a variable number of repeats:

You can use the *Data Element Separation* method Use Data Pattern to support a variable number of repetitions in an otherwise fixed length environment. However, it relies on the ability to recognize the end of the repetitions based on the data content.

In its simplest form, you can do this by specifying a regular expression Data Pattern that matches a fixed number of characters that is terminated by reaching the end of the message bit stream.

For example, consider a message with one fixed length field (length 10), followed by another fixed length field (length 20) that repeats indefinitely to the end of the bit stream:

```

Message  Data Element Separation=Use Data Pattern
  FieldA Data Pattern=. {10}
  FieldB Repeat, Min Occurs=1, no Max Occurs, Data Pattern=. {20}

```

The following example message contains a fixed length field (length 20) that repeats a variable number of times, and is separated from a second field by the string ";". The pattern specifies a string of 20 characters starting with anything except a semicolon:

```
Message Data Element Separation=All Elements Delimited, Delimiter=;
SubType1 Data Element Separation=Use Data Pattern
FieldA Repeat, Min Occurs=1, no Max Occurs, Data Pattern=[^;].{19}
FieldB
```

Performance considerations when using regular expressions:

You should take care when specifying regular expressions: some forms of regular expression can involve a large amount of work to find the best match, adversely impacting performance. Other expressions might produce a result that you did not expect.

For example, to match text up to and including a delimiter character ';' do *not* use the pattern ".*;" because this matches up to the *last* ';' character in the message, including any prior ';' characters in the matched text. Instead, you should use the pattern "[^;]*;".

Similarly, avoid using the pattern ".*" because this will always force a search to the end of the message to try and find the best match, and this might result in poor performance. However, you *should* use the pattern ".*" if you intend to match all remaining data in a message.

For best performance, avoid expressions with redundant nested repeats, such as "([0-9]+)*". Try to keep the expressions simple, with precise matching criteria. This avoids the need to perform multiple searches for the best match.

DateTime formats

When you create a `dateTime` object in the MRM, you must specify a format string in the object's *Format String* property for each physical format layer (CWF, TDS, XML). You can use the symbols defined in the information below to control the format in which the `dateTime` appears in the message data.

You can only use `dateTime` for Gregorian calendar dates.

`DateTime` information can appear in a message as:

- String data. This includes TDS, XML, and all CWF physical formats except those mentioned below. This is described further in "DateTime as STRING data."
- Binary data. This is for the CWF Binary physical format. See "DateTime as CWF BINARY data" on page 530 for more information.
- An offset from an epoch in seconds or milliseconds. This is used if you have set the CWF *Physical Type* property to Time Seconds or Time Milliseconds respectively. See "DateTime as CWF encoded values" on page 531 for details of this option.

The defaults that are set for each message set property that relates to `dateTime`, for each physical representation (CWF, TDS, XML), are defined in "Message set defaults" on page 532.

DateTime as STRING data

If the `dateTime` object is CWF Packed Decimal, you can only use the symbols that are presented as numbers. For all other *Physical Type* options, you can use all symbols.

You can specify the `dateTime` format using a string of pattern letters. The count of pattern letters determines the format. The table of `dateTime` formatting symbols below defines the letters that are reserved as pattern letters:

Symbol	Meaning	Presentation	Example
a	am or pm marker	Text	pm
d	day in month	Number	10
D	day in year	Number	189
e	day in week (1-7)	Number	2
E	day in week	Text	Tuesday
F	day of week in month	Number	2 (2nd Wed in July) ³
G	Era	Text	AD
h	hour in am or pm (1-12)	Number	12
H	hour in day (0-24)	Number	0
I	Parse as Date/Time (ISO8601)		(note 6)
k	hour in day (1-24)	Number	24
K	hour in am or pm (0-11)	Number	0
m	minute in hour	Number	30
M	month in year	Text and Number	July and 07
w	week in year	Number	27 ²
W	week in month	Number	2
s	second in minute	Number	55
S	millisecond	Number	978
T	Parse as Time (ISO8601)		(note 6)
y	year	Number	1996 ¹
Y	year: use with week in year only	Number	1996 ²
z	TimeZone	Text	zzzz = Pacific Standard Time
Z	time zone	Text	+05:30
'	escape for text		'User text'
"	single quote within escaped text		'o''clock'

The presentation of the `dateTime` object depends on what symbols you specify as follows:

- **Text.** If you specify four or more of the symbols, the full form is presented. If you specify less than four, the short or abbreviated form, if it exists, is presented. For example, `EEEE` produces `Monday`, `EEE` produces `Mon`.
- **Number.** Repeat the symbol to specify the minimum number of digits that you want. Shorter numbers are padded with zeros to this length. For example, if you

specify `m`, the number 6 is presented. If you specify `mm`, the number 06 is presented. A year is a special case, see note 1 in the list below.

- **Text and number.** If you specify three or more symbols, text is displayed. If you specify less, the number is presented. For example, if you specify `M`, it produces 1. If you specify `MM`, it produces 01. If you specify `MMM`, it produces Jan. If you specify `MMMM`, it produces January.

Any characters in the pattern that are not in the ranges of `['a'..'z']` and `['A'..'Z']` are treated as quoted text. For example, characters like colon (:), comma (,), period (.), the number sign (hash or pound, #), the at sign (@) and space appear in the resulting time text even if they are not enclosed within single quotes.

Notes:

The following points explain the notes in the table above:

1. Year is handled as a special case:
 - On output, if the count of `y` is 2, the year is truncated to 2 digits. For example, if `yyyy` produces 1997, `yy` produces 97.
 - On input, for 2 digit years the CWF message set property of *Century Window* is used to determine the century. For example, if *Century Window* is set to 53, year 97 is 1997, year 52 is 2052, and year 53 is 1953.
2. The first day of a year does not usually fall on a week boundary, therefore dates expressed using week in year might refer to dates in neighboring years. For example, day 1 of week 1 in 2002 (2002 01 Monday) using format string `YYYY ww EEEE` is in fact 31st December 2001. If you use `Y`, the day of week (`E`) and week in year (`w`) are adjusted if necessary to indicate that the date falls in the previous year. If you use the `y` symbol, the adjustment is not done and unpredictable results could occur for dates around year end.
For example, if the string 2002 01 Monday is formatted:
 - day 1 of week 1 in 2002 using format string `YYYY ww EEEE` is correctly interpreted as 31st December 2001
 - day 1 of week 1 in 2002 using format string `yyyy ww EEEE` is incorrectly interpreted as 30th December 2002`Y` should only be used in conjunction with `w`. If you specify `Y` without `w`, the year is ignored. For example, if you specify `YYYY-mm-dd` to format 1996-03-01 the result is 2002-03-01 because the year input is ignored and the current year is assumed.
3. The 11th July 2001 is the second Wednesday in July and can be expressed as 2001 July Wednesday 2 using format string `yyyy MMMM EEEE F`. This is not the same as Wednesday in week 2 of July 2001, which is 4th July 2001.
4. The first and last week in a month might include days from neighboring months. For example, Tuesday 31st July 2001 could be expressed as *Tuesday in week one of August 2001*, which is 2001 08 1 Tuesday using format string `yyyy MM W EEEE`.
5. You can only express a time zone as an offset in hours and minutes from GMT (`+/-hh:mm`). The number of `Z` formatting symbols affects the output:
 - `-Z` (short form) produces `-5`
 - `-ZZ` (medium form) produces `-05`
 - `-ZZZ` (long form) produces `-05:00`
 - `-ZZZZ` (long form) produces `GMT-05:00`
6. You can use the formatting symbol `I` (uppercase letter "i") to match any ISO8601 date`Time` strings. Examples of these are shown in the ISO8601 date`Time` examples below. An `I` should only be used alone.

On input to a message flow, a format string of I allows any ISO8601 compliant date`Time` to be parsed. On output from a message flow, the date`Time` is always expressed in the fullest form `yyyy-MM-dd'T'HH:mm:ss.SSS`.

If you use the `T` formatting symbol, format strings can be constructed to match ISO8601 date`Times` where `T` precedes the time portion of a date`Time` entity.

Examples:

The following table shows examples of date`Time` formats:

Format pattern	Result
"yyyy.MM.dd'at'HH:mm:ss ZZZ"	1996.07.10 at 15:08:56 -05:00
EEE, MMM d, "yy"	Wed, July 10, '96
"h:mm a"	8:08 PM
"hh 'o'clock' a, ZZZZ"	09 o'clock AM, GMT+09:00
"K:mm a, ZZZ"	9:34 AM, -05:00
"yyyy.MMMMM.dd hh:mm aaa"	1996.July.10 12:08 PM

The following are ISO8601 date`Time` examples:

Year
 yyyy
 Year and month
 yyyy-MM
 Complete date
 yyyy-MM-dd
 Complete date plus hours and minutes
 yyyy-MM-ddTHH:mm
 Complete date plus hours, minutes, and seconds
 yyyy-MM-ddTHH:mm:ss
 Complete date plus hours, minutes, seconds, and a decimal fraction of a second
 yyyy-MM-ddTHH:mm:ss.S

You can create formatting strings that produce unpredictable results, so you must use these symbols with care. For example, if you specify `dMyyyy`, it is impossible to distinguish between day, month, and year. `dMyyyy` tells the broker that a minimum of one character represents the day, a minimum of one character represents the month, and four characters represent the year. Therefore `3111999` could be interpreted as `3/11/1999` and `31/1/1999`.

Date`Time` as CWF BINARY data

The count of pattern letters determines the number of bytes used to represent a value. The symbol used in the pattern of letters can only be used in groups of 1, 2, or 4, for example, `y`, `yy`, or `yyyy`.

The following table shows the date`Time` symbols for CWF binary data:

Symbol	Meaning	Example
y	year	1996
M	month in year	7
d	day in month	10
H	hour in day (0-23)	13

Symbol	Meaning	Example
m	minute in hour	30
s	second in minute	55
S	millisecond	978
X	Ignore on input Pad with zeros on output	

The following example shows the C language structure `tm` with an integer of four bytes:

```
struct tm
{ int tm_sec;      /* seconds after the minute - [0,59]*/
{ int tm_min;      /* minutes after the hour   - [0,59]*/
{ int tm_hour;     /* hours since midnight    - [0,23]*/
{ int tm_mday;     /* day of the month       - [1,31]*/
{ int tm_mon;      /* months since January   - [0,11]*/
{ int tm_year;     /* years since 1900       */
{ int tm_wday;     /* days since Sunday      - [0,6]*/
{ int tm_yday;     /* days since January 1   - [0,365]*/
{ int tm_isdst;    /* daylight savings time flag */
};
```

You can format this structure by specifying the string `"ssssmmmmHHHHdddMMMM+1yyyy+1900XXXXXXXXXXXX"`. The number of pattern letters determines the number of bytes. There are 36 A-Z characters specified in this pattern, which match the 36 byte structure `tm`. A field followed by a plus sign (+) has the succeeding numeric characters added to it. Therefore `MMMM+1` adds one to the month, `yyyy+1900` adds 1900 to the year. `X` expects one byte of input, but ignores its value. On output, it outputs the byte as 0.

DateTime as CWF encoded values

You can represent a `dateTime` object with the following physical types:

- `TimeSeconds`. This is a 4 byte integer that represents the number of seconds since the epoch.
- `TimeMilliSeconds`. This is an 8 byte integer that represents the number of milliseconds since the epoch.

These types provide a way for c `time_t` and Java `dateTime` representations to be parsed.

The epoch (time 0) is specified by the format string. This has the default value of `1970-01-01T00:00 +00:00`. You can specify other epoch strings, which must conform to the format pattern `"yyyy-MM-dd'T'HH:mm ZZZ"`.

DateTime defaults by logical type

The default values assigned to `dateTime` property are dependant on the logical type of the property. The following table lists the defaults for each of the logical `dateTime` types:

Logical Type	Default Format
date	yyyy-MM-dd
dateTime	yyyy-MM-dd'T'HH:mm:ss
gDay	- - -dd

Logical Type	Default Format
gMonth	- -MM- -
gMonthDay	- -MM-dd
gYear	yyyy
gYearMonth	yyyy-MM
time	HH:mm:ssZZZ

DateTime component defaults

Default values are assumed if any part of a dateTime object is not present on input. For example, the formatting string yyyy-MM'T'HH:mm does not contain any information about day in month (d), seconds (s), or milliseconds (S). The table below shows the defaults for all dateTime components:

Component	Default value
Year	1970
Month	First month of year
Day	First day of month
Hour	First hour of day
Minute	Minute 0 of hour
Second	Second 0 of minute
Millisecond	Millisecond 0 of second

Message set defaults

The table below shows the default dateTime formatting symbols for the different physical message representations:

Message set property	CWF default	TDS default	XML default
<i>Default DateTime Format</i>	See Note 1.	See Note 1.	See Note 1.
<i>Default Time Zone ID</i>	Use Broker Locale ²	Use Broker Locale ²	Use Broker Locale ²
<i>Century Window</i>	53	53 (80 for SWIFT)	53
<i>Days in First Week of Year</i>	4	Use Broker Locale ²	Use Broker Locale ²
<i>First Day of Week</i>	Monday	Use Broker Locale ²	Use Broker Locale ²
Note: 1. You can either set the default dateTime format to be derived from its logical type (the default), or specify the dateTime format to be used. This is set at the message set level for each physical format that has been added. 2. The key phrase Use Broker Locale causes the broker to get the information from the underlying platform.			

You can update all these default values. The CWF defaults are set for all values of the *Physical Type* property. If you change the CWF *Physical Type* to Binary, Packed Decimal, TimeSeconds, or TimeMilliseconds, you must update the *Default DateTime Format* property manually to ensure consistent results.

For more information about these message set properties, see “Custom Wire Format message set properties” on page 4, “TDS Format message set properties” on page 12, or “XML Wire Format message set properties” on page 7.

Generated model representations

This section provides information on the possible generated model representations. Details are provided for:

- “Document generation”
- “WSDL generation”
- “XML Schema generation” on page 538

Document generation

Output Files

The document generator produces a set of HTML pages and any necessary files (for example, images) that are required to display the pages correctly.

There is one page for each message definition file in the message set, and one additional index page linking these pages together.

The index page (index.html), is intended to be the “entry point” into the documentation.

WSDL generation

The table below defines the general properties that you must set to generate a Web Service Definition from a message set:

Property	Type	Meaning
Company Domain Name	String	Specify the company domain name for the service. The wizard generates a default value for this, which you can change if required.
Definition Name	String	Specify the name of the Web Service Definition that you are creating. The default is the selected message set name, which you can change if required.
Port Type Name	String	Specify the name of the PortType that is used to define the set of operations within the service interface. The default is the selected message set name followed by PortType, for example OrderMessageSetPortType. You can change this if required.
Operation Type	Enumerated type	Select the type of operation that the service supports from the drop down list. The available types of operation are: <ul style="list-style-type: none">• One-way Operation. The endpoint receives a message.• Request-response Operation. The endpoint receives a message, and sends a correlated message.• Solicit-response Operation. The endpoint sends a message, and receives a correlated message.• Notification Operation. The endpoint sends a message.

In addition to the above general properties, there are a number of binding and service properties that you need to define. The required properties depend on the type of binding you select. For further information, see the following topics:

- “SOAP (over JMS) properties” on page 534
- “SOAP (over HTTP) properties” on page 536
- “JMS (TextMessage) properties” on page 536

SOAP (over JMS) properties

The following properties apply to sending SOAP messages over the Java Message Service.

Binding properties

Property	Type	Meaning
Binding Name	String	<p>Specify the name of the binding to be used by the service for sending SOAP messages over the Java Message Service.</p> <p>Binding names can start with a letter, or the underscore character only.</p> <p>The default binding name consists of the message set name followed by JMS_SOAP_Binding, for example OrderMessageSetJMS_SOAP_Binding.</p> <p>You can change the default if required.</p>
Optional JMS Property Values (for each property you define)		
Name	String	<p>Specify the name of the JMS property.</p> <p>Names can start with a letter, or the underscore character only.</p>
XSD Type	Enumerated type	Select the data type of the value of the JMS property from the drop down list.
Value	String	Specify the value of the JMS property.

Service properties

The following properties apply in all cases:

Property	Type	Meaning
Service Name	String	<p>Specify the name of the service being created for sending SOAP messages over JMS.</p> <p>Service names can start with a letter, or the underscore character only.</p> <p>The default service name consists of the message set name followed by JMS_SOAP_Service, for example OrderMessageSetJMS_SOAP_Service.</p> <p>You can change the default if required.</p>
Port Name	String	<p>Specify the name of the port used by the service. The default is JMSOAPPort.</p> <p>You can change this name if required.</p>

The remaining properties depend on which of the following JMS Address Provider options you select:

- JMS Address Provider Neutral
- JMS Address Provider Neutral with JNDI Reference
- JMS Address Provider Specific

JMS Address Provider Neutral

Property	Type	Meaning
Destination Style	String	This property defines the type of destination for the service. This is set to queue. You cannot change this value.
JMS Vendor URI	String	Specify the vendor URI which uniquely identifies the JMS implementation. The wizard generates a default value for this, which you can change if required.
Initial Context Factory	String	Specify the initial context factory that creates an initial context for the service. The wizard generates a default value for this, which you can change if required.
JMS Provider Destination Name	String	Specify the name of the destination (queue) to which messages are sent. The wizard generates a default value for this, which you can change if required.

JMS Address Provider Neutral with JNDI Reference

Property	Type	Meaning
Destination Style	String	This property defines the type of destination for the service. This is set to queue. You cannot change this value.
JMS Vendor URI	String	Specify the vendor URI which uniquely identifies the JMS implementation. The wizard generates a default value for this, which you can change if required.
Initial Context Factory	String	Specify the initial context factory that creates an initial context to be used to locate the JNDI name of the connection factory. The wizard generates a default value for this, which you can change if required.
JNDI Provider URL	String	Specify the provider URL containing information that the initial context factory can use to obtain an initial context. The wizard generates a default value for this, which you can change if required.
JNDI Connection Factory	String	Specify the JNDI connection factory used to create connections with the JMS provider for the specified destination. Each connection factory encapsulates the configuration parameters needed to create a connection to this destination. The wizard generates a default value for this, which you can change if required.
JNDI Destination Name	String	Specify the name of the destination (queue) to which messages are sent. The wizard generates a default value for this, which you can change if required.

JMS Address Provider Specific

Property	Type	Meaning
JMS Vendor URI	String	Specify the vendor URI which uniquely identifies this JMS implementation. The wizard generates a default value for this, which you can change if required.
JMS Provider Server Name	String	Specify the name of the server which provides the JMS provider functions for the service. The wizard generates a default value for this, which you can change if required.
Port	Integer	Specify the actual location (endpoint) of the service.
Queue Manager Name	String	Specify the name of the queue manager associated with the service. The wizard generates a default value, which you can change if required.

Property	Type	Meaning
Queue Name	String	Specify the name of the queue to which messages are sent. The wizard generates a default value, which you can change if required.

SOAP (over HTTP) properties

The following properties apply to sending SOAP messages over the HTTP.

Binding properties

Property	Type	Meaning
Binding Name	String	Specify the name of the binding to be used by the service for sending SOAP messages over HTTP. Binding names can start with a letter, or the underscore character only. The default binding name consists of the message set name followed by HTTP_SOAP_Binding, for example OrderMessageSetHTTP_SOAP_Binding. You can change the default if required.
Style	String	This property indicates the type of operation for the service. In all cases this property is set to rpc to indicate that the operation is RPC-oriented (messages containing parameters and return values). You cannot change this value.
Transport	String	Specify the transport to which the binding corresponds. For a SOAP binding this field defaults to the URI value that corresponds to the HTTP binding in the SOAP specification. You can change this value if required.
SOAP Action	String	Specify the value of the SOAPAction header for the operation. The wizard generates a default value, which you can change if required.

Service properties

Property	Type	Meaning
Service Name	String	Specify the name of the service being created for sending SOAP messages over the HTTP. Service names can start with a letter, or the underscore character only. The default service name consists of the message set name followed by HTTP_SOAP_Service, for example OrderMessageSetHTTP_SOAP_Service. You can change the default if required.
Port Name	String	Specify the name of the TCP/IP port for the SOAP service. The default is HTTPSOAPPort. You can change this name if required.
SOAP Port Address	String	Specify the target address of the SOAP service. This is the URI for the endpoint of the service, for example "MyService/servlet/rcprouter". The wizard generates a default value, which you can change if required.

JMS (TextMessage) properties

The following properties apply to sending text messages over the Java Message Service.

Binding properties

Property	Type	Meaning
Binding Name	String	<p>Specify the name of the binding to be used by the service for sending text messages over the Java Message Service.</p> <p>Binding names can start with a letter, or the underscore character only.</p> <p>The default binding name consists of the message set name followed by JMSBinding, for example OrderMessageSetJMSBinding.</p> <p>You can change the default if required.</p>
Optional JMS Property Values (for each property you define)		
Name	String	<p>Type the name of the JMS property.</p> <p>Names can start with a letter, or the underscore character only.</p>
XSD Type	Enumerated type	Select the data type of the value of the JMS property from the drop down list.
Value	String	Enter the value of the JMS property.

Service properties

The following properties apply in all cases:

Property	Type	Meaning
Service Name	String	<p>Specify the name of the service being created for sending text messages over JMS.</p> <p>Service names can start with a letter, or the underscore character only.</p> <p>The default service name consists of the message set name followed by JMSTextMessage_Service, for example OrderMessageSetJMSTextMessage_Service.</p> <p>You can change the default if required.</p>
Port Name	String	Specify the name of the port used by the service. The default is JMSPort. You can change this name if required.

The remaining properties depend on which of the following JMS Address Provider options you select:

- JMS Address Provider Neutral
- JMS Address Provider Neutral with JNDI Reference
- JMS Address Provider Specific

JMS Address Provider Neutral

Property	Type	Meaning
Destination Style	String	This property defines the type of destination for the service. This is set to queue. You cannot change this value.
JMS Vendor URI	String	Specify the vendor URI which uniquely identifies the JMS implementation. The wizard generates a default value for this, which you can change if required.
Initial Context Factory	String	Specify the initial context factory that creates an initial context for the service. The wizard generates a default value for this, which you can change if required.

Property	Type	Meaning
JMS Provider Destination Name		Specify the name of the destination (queue) to which messages are sent. The wizard generates a default value for this, which you can change if required.

JMS Address Provider Neutral with JNDI Reference

Property	Type	Meaning
Destination Style	String	This property defines the type of destination for the service. This is set to queue. You cannot change this value.
JMS Vendor URI	String	Specify the vendor URI which uniquely identifies the JMS implementation. The wizard generates a default value for this, which you can change if required.
Initial Context Factory	String	Specify the initial context factory that creates an initial context to be used to locate the JNDI name of the connection factory. The wizard generates a default value for this, which you can change if required.
JNDI Provider URL	String	Specify the provider URL containing information that the initial context factory can use to obtain an initial context. The wizard generates a default value for this, which you can change if required.
JNDI Connection Factory	String	Specify the JNDI connection factory used to create connections with the JMS provider for the specified destination. Each connection factory encapsulates the configuration parameters needed to create a connection to this destination. The wizard generates a default value for this, which you can change if required.
JNDI Destination Name	String	Specify the name of the destination (queue) to which messages are sent. The wizard generates a default value for this, which you can change if required.

JMS Address Provider Specific

Property	Type	Meaning
JMS Vendor URI	String	Specify the vendor URI which uniquely identifies this JMS implementation. The wizard generates a default value for this, which you can change if required.
JMS Provider Server Name	String	Specify the name of the server which provides the JMS provider functions for the service. The wizard generates a default value for this, which you can change if required.
Port	Integer	Specify the actual location (endpoint) of the service.
Queue Manager Name	String	Specify the name of the queue manager associated with the service. The wizard generates a default value, which you can change if required.
Queue Name	String	Specify the name of the queue to which messages are sent. The wizard generates a default value, which you can change if required.

XML Schema generation

This topic covers the behavior of XML Schema generation. For example, you could use the schema generated from a message definition file to subsequently validate XML instance documents written by WebSphere Business Integration Message Broker.

Lax generation

Lax generation affects how complex types that have *Content Validation* set to Open or OpenDefined or have *Composition* set to UnorderedSet are rendered in the generated schema. Note that such a validating schema will permit a wider range of messages than MRM parser validation.

Content Validation is set to Open or OpenDefined

Here a complex type (global or anonymous) has its content replaced by a single element of type anyType. The following generation pattern is used for complex types with *Content Validation* set to Open:

```
<element name="xmlNameOfMessage">
  <complexType>
    <sequence>
      <any processContent="lax"
        minOccurs="0" maxOccurs="unbounded" />
    </sequence>
  </complexType>
</element>
```

Where *Content Validation* is set to OpenDefined, the following pattern is used. (The namespaces listed are all those defined in the containing message set.)

```
<element name="xmlNameOfMessage">
  <complexType>
    <sequence>
      <any processContent="lax"
        minOccurs="0" maxOccurs="unbounded"
        namespace="http://www.ns1 http://www.ns2" />
    </sequence>
  </complexType>
</element>
```

Composition is set to UnorderedSet

Where *Composition* is set to UnorderedSet, to mimic the unordered aspect, a choice is inserted with appropriate cardinality. This is shown below.

```
<element name="xmlNameOfMessage">
  <complexType>
    <sequence maxOccurs="unbounded"
      minOccurs=" (minOccurs of original sequence) *"
      (items in original sequence)">
      <choice>
        .. sequence contents ..
      </choice>
    </sequence>
  </complexType>
</element>
```

Strict generation

Strict generation affects how complex types that have *Content Validation* set to Open or OpenDefined or have *Composition* set to UnorderedSet are rendered in the generated schema. Note that such a validating schema will permit a narrower range of messages than MRM parser validation.

Strict is the default generation option and generates a schema that matches the schema held in the message definition file, without the model extensions.

Content Validation set to Open/OpenDefined

A complex type (global or anonymous) will lose the ability to contain self-defining elements and becomes closed.

Composition set to UnorderedSet

A complex type (global or anonymous) will lose the ability to be unordered and becomes a sequence.

Rendering of xsd:elements

If an XML physical format is specified when generating the schema, the wire format customization is applied to the logical model. These properties control how an element in the model is actually rendered when it appears in a message for an XML wire format. See “XML Message rendering options” on page 61 for the different render options available. A generated schema example is given below showing what is generated for the different render options available for local elements; note these examples do not modify the Namespace of any ID Attribute Name or Value Attribute Name properties and assume that all elements specified in the complexType1 are of schema built-in type string.

```
<xsd:complexType name="complexType1">
  <xsd:sequence>
    <!-- Local element Render = 'XMLElement' -->
    <xsd:element name="localElement1" type="xsd:string"/>
    <!-- Local element Render = 'XMLElementAttrID'
      ID Attribute Name = 'id' -->
    <xsd:element name="localElement2">
      <xsd:complexType>
        <xsd:simpleContent>
          <xsd:extension base="xsd:string">
            <xsd:attribute name="id" type="xsd:string"/>
          </xsd:extension>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
    <!-- Local element Render = 'XMLElementAttrVal'
      Val Attribute Name = 'val' -->
    <xsd:element name="localElement3">
      <xsd:complexType>
        <xsd:simpleContent>
          <xsd:extension base="xsd:string">
            <xsd:attribute name="val" type="xsd:string"/>
          </xsd:extension>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
    <!-- Local element Render = 'XMLElementAttrIDVal'
      ID Attribute Name = 'id' Val Attribute Name = 'val' -->
    <xsd:element name="localElement4">
      <xsd:complexType>
        <xsd:simpleContent>
          <xsd:extension base="xsd:string">
            <xsd:attribute name="val" type="xsd:string"/>
            <xsd:attribute name="id" type="xsd:string"/>
          </xsd:extension>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <!-- Local element Render = 'XMLAttribute' -->
  <xsd:attribute name="localElement5" type="xsd:string"/>
</xsd:complexType>
```

Rendering of xsd:attributes

The rendering of xsd:Attributes is not supported. The user can only change the name of the attribute.

Embedded simple types and Compound Elements

These are modeled in the message definition file as elements with both *minOccurs* and *maxOccurs* set to 0 and have one of the predefined *ComIbmMrm_XXX* types. During the schema generation, the type of such elements is changed to the base type of the respective *ComIbmMrm_XXX* type.

If there are global simple types that inherit from one of these *ComIbmMrm_XXX* types, these are changed to inherit from the base type of the corresponding *ComIbmMrm_XXX* type.

Generated schema files will not have any occurrence of these *ComIbmMrm_XXX* types.

For example the global element with type defined below:

```
<element name="globalElement1" type="ns1:ComIbmMrm_BaseValueBinary"/>
```

will result in the generated schema file and a global element with the corresponding xsd base type as defined below:

```
<element name="globalElement1" type="hexBinary"/>
```

Import formats

This section provides information on the supported features of formats that have been imported from an external source. Details are provided for:

- “Importing from C: supported features”
- “Importing from COBOL: supported features” on page 544
- “Importing from XML Schema: unsupported features” on page 548
- “Importing from XML DTD: unsupported features” on page 551

Importing from C: supported features

This section describes the defaults that the C importer uses when mapping C datatypes to MRM datatypes.

The table below shows how the C definitions influence the XML schema settings in the message model. Note that some xsd types have ‘-’ after the type. This indicates that it is an anonymous simple type based on this type. For strings, the purpose of the anonymous type is to add a length restriction; for numeric types the purpose is to add a minimum and/or maximum value restriction.

The following datatypes are outside the scope of the C importer:

- Packed data structure.

C datatype	Logical type	Notes
Char	xsd:string-	length=1
Char[10]	xsd:string-	length=10
Char[10][3]	xsd:string-	length=3
Char[10][3][6]	xsd:string-	length=6
Unsigned Char	xsd:unsignedByte	
Unsigned Char[2]	xsd:unsignedByte	
Signed Char	xsd:byte	

C datatype	Logical type	Notes
Signed Char[2]	xsd:byte	
Int	xsd:int	
Int[2]	xsd:int	
Int[2][3]	xsd:int	
Unsigned Int	xsd:unsignedInt	
Short	xsd:short	
Unsigned Short	xsd:unsignedShort	
Long	xsd:int	
Long Long Int	xsd:long	
Float	xsd:float	
Double	xsd:double	
Long Double ^(see note 1)	xsd:double	
<any pointer type>	xsd:hexBinary-	length= ^(see note 2)
<any enum>		^(see note 3)

The following table shows how C definitions influence the physical CWF characteristics of the elements generated in the message model.

C datatype	CWF Physical type	CWF Length characteristics	Other CWF characteristics
Char	Fixed Length	Length Count = 1 Length Units = Bytes	
Char[10]	Fixed Length	Length Count = 10 Length Units = Bytes	Left justify
Char[10][3]	Fixed Length	Length Count = 3 Repeat Count = 10 Length Units = bytes	Left justify
Char[10][3][6]	Fixed Length	Length Count = 6 Repeat Count = 30 Length Units = bytes	Left justify
Unsigned Char	Integer	Length Count = 1	Signed = no
Unsigned Char[2]	Integer	Length Count = 1 Repeat Count = 2	Signed = no
Signed Char	Integer	Length Count = 1	Signed = yes
Signed Char[2]	Integer	Length Count = 1 Repeat Count = 2	Signed = yes
Int	Integer	Length Count = 4	Signed = yes
Int[2]	Integer	Length Count = 4 Repeat Count = 2	Signed = yes

C datatype	CWF Physical type	CWF Length characteristics	Other CWF characteristics
Int[2][3]	Integer	Length Count = 4 Repeat Count = 6	Signed = yes
Unsigned Int	Integer	Length Count = 4	Signed = no
Short	Integer	Length Count = 2	Signed = yes
Unsigned Short	Integer	Length Count = 2	Signed = no
Long	Integer	Length Count = 4 ^(see note 4)	Signed = yes
Long Long Int	Integer	Length Count = 8	Signed = yes
Float	Float	Length Count = 4	
Double	Float	Length Count = 8	
Long Double ^(see note 1)	Float	Length Count = 8	
<any pointer type>		(see note 2)	
<any enum>	Integer	(see note 3)	

Notes:

- Do not set the value of C importer option size of long double to 128 bit. This will not import successfully. Please use the default 64 bit.
- The length count is affected by the Address Size C importer option as follows:
 - For 32 bit, CWF length count = 4 bytes.
 - For 64 bit, CWF length count = 8 bytes.
- The type and length of an enum is affected by the Size of enum C importer option as follows:
 - For 1: Logical type = xsd:byte, CWF physical type = Integer, CWF length count = 1 byte.
 - For 2: Logical type = xsd:short, CWF physical type = Integer, CWF length count = 2 bytes.
 - For 4: Logical type = xsd:int, CWF physical type = Integer, CWF length count = 4 bytes.
 - For Compact: The smallest representation is chosen that the enumeration fits into.
- The length of a long is affected by the Address Size C importer option as follows:
 - For 32 bit: CWF length count = 4 bytes.
 - For 64 bit: CWF length count = 8 bytes.
- Element names that clash with Java language keywords are modified by prefixing them with a single underscore character.
- The `_Packed` keyword is not supported. Only ANSI C declarations are supported.
- The C long long datatype is not supported.
- C++ object oriented extensions are not supported. Only ANSI C declarations are supported.
- Pointers will be imported as xsd:integer with CWF length count set to 4.
- Recursive C structures are not supported. If a nested structure contains a structure with a name that is the same as the parent structure, the

import succeeds but the logical definitions are not correct. To avoid this problem, ensure that the name of the nested structure is not the same as that of the outer or parent structure.

Importing from COBOL: supported features

The following table shows how COBOL definitions influence the XML schema settings in the message model. Note that some xsd types have '-' after the type. This indicates that it is an anonymous simple type based on this type. For strings, the purpose of the anonymous type is to add a length restriction; for numeric types the purpose is to add a minimum and/or maximum value restriction.

COBOL Clause	XML Schema datatype	Notes
PIC A	xsd:string -	
PIC X	xsd:string -	
PIC 9(n) n = 1-4	xsd:short -	DISPLAY, COMP, or COMP-3
PIC 9(n) n = 5-9	xsd:int -	DISPLAY, COMP, or COMP-3
PIC 9(n) n = 10-18	xsd:long -	DISPLAY, COMP, or COMP-3
PIC 9(n) n = 19-31	xsd:integer -	DISPLAY, COMP, or COMP-3
PIC 9(n)V9(m)	xsd:decimal -	DISPLAY, COMP, or COMP-3 any virtual decimal point value
COMP-1	xsd:float -	
COMP-2	xsd:double -	
Any edited string	xsd:string -	
Any edited number	xsd:string -	For example, PICZ
VALUE	All	Non-88 Level VALUE clauses can be imported as schema default values (option on import wizard).

The following table shows how COBOL definitions influence the physical CWF characteristics of the elements generated in the message model.

COBOL Keyword	CWF Physical Type	CWF Length Characteristics	Other CWF characteristics
PIC X(n) PIC A(n)	Fixed Length String	Length = n Length Units = Bytes	Justification = Left Justify Padding Character = SPACE
PIC 9(n) DISPLAY n=1-9	External Decimal	Length = n Length Units = Bytes	Justification = Right Justify Padding Character = '0' Signed = Unticked Sign Orientation = Trailing
PIC 9(n) DISPLAY n=10-31	External Decimal	Length = n Length Units = Bytes	Justification = Right Justify Padding Character = '0' Signed = Unticked Sign Orientation = Trailing

COBOL Keyword	CWF Physical Type	CWF Length Characteristics	Other CWF characteristics
PIC 9(n) COMP, COMP-4, COMP-5 or BINARY	Integer	Length = 1, 2, 4 or 8 based on n Length Units = Bytes	Signed = Unticked Sign Orientation = Blank
PIC 9(n) COMP-3 n=1-9	Packed Decimal	Length = $\text{CEILING}((n+1)/2)$ Length Units = Bytes	Signed = Unticked Sign Orientation = Blank
PIC 9(n) COMP-3 n=10-31	Packed Decimal	Length = $\text{CEILING}((n+1)/2)$ Length Units = Bytes	Signed = Unticked Sign Orientation = Blank
PIC S9(n) DISPLAY n=1-31	External Decimal	Length = n Length Units = Bytes	Signed = Unticked Sign Orientation = Blank *See Note 1
PIC S9(n) COMP or COMP-3 n=1-31	Integer or Packed Decimal	Length = See COMP and COMP-3 definitions above Length Units = Bytes	Signed = Unticked Sign Orientation = Blank
PIC 9(m)V9(n) DISPLAY n=1-31	External Decimal	Length = n+m Length Units = Bytes	Signed = Unticked Sign Orientation = Trailing Virtual Decimal Point = n
PIC 9(m)V9(n) COMP or COMP-3	Integer or Packed Decimal	Length = $\text{CEILING}((n+m+1)/2)$ for COMP-3 Length = 1, 2, 4 or 8 for COMP Length Units = Bytes	Signed = Unticked Sign Orientation = Trailing
COMP-1	Float	Length = 4 Length Units = Bytes	Signed = Ticked Sign Orientation = Blank
COMP-2	Float	Length = 8 Length Units = Bytes	Signed = Ticked Sign Orientation = Blank
SYNC	Float, Integer or Packed Decimal		Leading Skip Count as appropriate Trailing Skip Count as appropriate Byte alignment as appropriate *See note 2

COBOL Keyword	CWF Physical Type	CWF Length Characteristics	Other CWF characteristics
Notes: <ol style="list-style-type: none"> Sign Orientation can take one of the following values, based on the SEPARATE, LEADING or TRAILING keywords in the COBOL definition. <ul style="list-style-type: none"> Leading Leading Separate Trailing Trailing Separate SYNC Keyword causes the field to be aligned on a 1, 2, 4, or 8 byte boundary. This may cause 'slack bytes' to be added either before or after a field. Leading Skip Count is the number of such bytes added before; Trailing Skip Count is the number added after. Leading Skip Count and Trailing Skip Count are calculated for each of the imported elements by the importer irrespective of SYNC clause. They will have non zero values when the SYNC clause is present. Where there is a repeating element, Leading Skip Count and Trailing Skip Count are used for the first occurrence of the repeating element, for subsequent occurrences, only the Trailing Skip Count is used. Refer to COBOL reference material for details of fields requiring Byte Alignment. The COBOL importer requires all files you are importing to be syntactically correct. Results are unpredictable if this is not the case. COBOL data types including POINTER, COMP-X, INDEX and PROCEDURE-POINTER are not supported. COBOL containing the keyword NATIVE causes an error and will not import. COBOL level 66 and level 77 data items are not imported. Hexadecimal binary values cannot be attributed to non-numeric literals. They cannot reside in the LINKAGE SECTIONs that are imported by the COBOL importer. They can reside elsewhere in the COBOL file. Alternatively, you can convert the hexadecimal value to a char string for PIC X, or to a decimal number for PIC 9. Where element names clash with Java language keywords, they are modified by prefixing the element name with a single underscore character. Object-oriented extensions to COBOL 85 are not supported. For example, OBJECT-REFERENCE is not supported. COBOL OCCURS DEPENDING ON clause. The Byte Alignment, Leading Skip Count, and Trailing Skip Count CWF properties of elements within such a structure are not set up properly. You must correct these using the message editor. When the imported COBOL source file contains QUOTE or QUOTES in the value clause of a picture string, the default behavior is to fill in the data with double quotation marks unless you set the COBOL QUOTE compile option to SINGLE on the Import Options page of the COBOL importer wizard. 			

Signed external decimal numbers

The Custom Wire Format (CWF) component of the WebSphere Business Integration Message Broker provides support for the modelling of numeric data using the External Decimal (also known as Zoned Decimal) data format. In this format, a number is stored internally as decimal character data. For example, in a system using the EBCDIC code, the number 1234 stored in a 4-byte external decimal field would be stored as the character string "1234" and its actual internal hexadecimal representation would be F1F2F3F4.

With signed external decimal numbers, the sign can be incorporated into the actual data by modifying the first half of the first or last byte (depending on whether you are using a sign-leading or sign-trailing representation). Typically, '0xC' is used to represent a positive number, '0xD' is used to represent a negative number and '0xF' is used to represent an unsigned number.

Note: In general, any of '0xA', '0xC', '0xE' or '0xF' may be used to indicate a positive value and '0xB' or '0xD' used to indicate a negative value. The actual preferred representation is dependent upon the actual hardware architecture.

On ASCII machines there are a number of mechanisms for the internal representation of external decimal data. One representation ('Sign ASCII') employed by IBM's pSeries machines uses the normal ASCII codes ("0" [hex 30] to "9" [hex 39]) for the first/last digit of both unsigned and positive numbers, and the characters "p" [hex 70] to "y" [hex 79] for negative numbers.

An alternative method (Sign EBCDIC Custom) is used on some other ASCII based machines. This uses the same characters as an EBCDIC based machine, even though the actual internal hexadecimal representation for them are different. Using this technique the character string for both EBCDIC and ASCII platforms is identical. You could potentially receive a message from an EBCDIC platform (created from a COBOL copy book that contains such entries as PIC XXX and PIC S999) and convert the whole message to ASCII or the other way around. The character string representing the external decimal field in the message (after the ASCII/EBCDIC conversion) maps to the code point which represents the correct sign for the decimal. You should note that there is a limitation with this method. Curly brace characters are variant (i.e. they have different code points in different EBCDIC codepages). This mechanism works only for those EBCDIC codepages where the curly brace characters '{' and '}' (which are used to represent signed 0) have exactly the code points x'C0' and x'D0'. For example, it will work for code page 500 and not for code page 871 where the curly braces have code points X'8E' X'9C.

In an ASCII environment (determined by the CCSID property at runtime), the default for both input and output is the 'Sign ASCII' representation. It is possible to specify the applicable representation in the CWF physical layer for local attributes and local elements of types decimal, float, and integer.

Note: This is only appropriate for those elements/attributes which have an external decimal physical representation and which have an embedded ('Leading' or 'Trailing') sign (determined by the 'Sign Orientation' property).

The table below show the internal representation (both character and actual hexadecimal value) of the first or last digit for external decimal numbers with an included (embedded) leading or trailing sign respectively. (Note: the table does not specify the representation for unsigned values which are 0x30-0x39 for ASCII and 0xF0-0xF9 for EBCDIC)

	Positively signed values				Negatively signed values		
	ASCII environment		EBCDIC environment		ASCII environment		EBCDIC environment
Digit	Sign ASCII	Sign EBCDIC Custom			Sign ASCII	Sign EBCDIC Custom	
0	0(30)	{(7B)	{(C0)		p(70)	}(7D)	}(D0)
1	1(31)	A(41)	A(C1)		q(71)	J(4A)	J(D1)
2	2(32)	B(42)	B(C2)		r(72)	K(4B)	K(D2)
3	3(33)	C(43)	C(C3)		s(73)	L(4C)	L(D3)
4	4(34)	D(44)	D(C4)		t(74)	M(4D)	M(D4)
5	5(35)	E(45)	E(C5)		u(75)	N(4E)	N(D5)
6	6(36)	F(46)	F(C6)		v(76)	O(4F)	O(D6)
7	7(37)	G(47)	G(C7)		w(77)	P(50)	P(D7)
8	8(38)	H(48)	H(C8)		x(78)	Q(51)	Q(D8)
9	9(39)	I(49)	I(C9)		y(79)	R(52)	R(D9)

This next table gives some examples for a range of simple numbers that would be representative of what could be transmitted or received using these approaches.

	Sign leading			Sign trailing		
	ASCII Environment		EBCDIC Environment	ASCII Environment		EBCDIC Environment
Decimal value	Sign ASCII	Sign EBCDIC Custom		Sign ASCII	Sign EBCDIC Custom	
1234	31 32 33 34 "1234"	31 32 33 34 "1234"	F1 F2 F3 F4 "1234"	31 32 33 34 "1234"	31 32 33 34 "1234"	F1 F2 F3 F4 "1234"
+1234	31 32 33 34 "1234"	41 32 33 34 "A234"	C1 F2 F3 F4 "A234"	31 32 33 34 "1234"	31 32 33 44 "123D"	F1 F2 F3 C4 "123D"
-1234	71 32 33 34 "q234"	4A 32 33 34 "J234"	D1 F2 F3 F4 "J234"	31 32 33 74 "123t"	31 32 33 4D "123M"	F1 F2 F3 D4 "123M"
7890	37 38 39 30 "7890"	37 38 39 30 "7890"	F7 F8 F9 F0 "7890"	37 38 39 30 "7890"	37 38 39 30 "7890"	F7 F8 F9 F0 "7890"
+7890	37 38 39 30 "7890"	47 38 39 30 "G890"	C7 F8 F9 F0 "G890"	37 38 39 30 "7890"	37 38 39 7B "789{"	F7 F8 F9 C0 "789{"
-7890	77 38 39 30 "w890"	50 38 39 30 "P890"	D7 F8 F9 F0 "P890"	37 38 39 70 "789p"	37 38 39 7D "789}"	F7 F8 F9 D0 "789}"

Importing from XML Schema: unsupported features

Message sets with namespace support

- Constructs not accepted when importing from an XML schema.

When importing an XML Schema into a message set that supports namespaces, the constructs below are not accepted. When the user tries to import a schema containing one or more of these constructs, an error is reported that indicates why and where the import fails.

The following is a list of these schema declarations:

- Redefine
- List

– Union

The following is an example of each of these constructs:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.ibm.com" xmlns:ibm="http://www.ibm.com">

    <!-- Unsupported feature: redefine -->
    <redefine schemaLocation="test.xsd"/>

    <!-- Unsupported feature: list -->
    <simpleType name="type1">
        <list itemType="string" />
    </simpleType>

    <!-- Unsupported feature: union -->
    <simpleType name="type2">
        <union memberTypes="string" />
    </simpleType>

</schema>
```

- Constructs accepted and ignored when importing from an XML schema.

When importing an XML Schema into a message set that supports namespaces, the constructs below are accepted, but will be ignored and will not be deployed to the broker. When the user tries to import a schema containing one or more of these constructs, a warning is issued stating that they will be ignored. You will be able to delete these constructs, but there are no properties that can be configured for them.

If you extract the logical model for the message (for example using the Schema Generation facility) these constructs are ignored and are not included in the output schema.

The following is a list of these schema declarations:

- Unique
- Key
- Keyref

The following is an example of each of these constructs:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.ibm.com" xmlns:ibm="http://www.ibm.com">

    <!-- Accepted feature: unique -->
    <element name="element1" type="string">
        <unique name="unique1">
            <selector xpath="path1"></selector>
            <field xpath="path1"></field>
        </unique>
    </element>

    <!-- Accepted feature: key -->
    <element name="element2" type="string">
        <key name="key1">
            <selector xpath="path1"></selector>
            <field xpath="path1"></field>
        </key>
    </element>

    <!-- Accepted feature: keyref -->
    <element name="element3" type="string">
        <keyref name="keyref1" refer="ibm:key1">
            <selector xpath="path1"></selector>
            <field xpath="path1"></field>
        </keyref>
    </element>
```

```

    </keyref>
  </element>

</schema>

```

- Where the Schema Location generated for some include statements could be incorrect.

The problem occurs when an XML schema that has a target namespace, includes another XML schema that has no target namespace. The Schema Location created in the message model for the include just specifies the filename. This does not take into account that the including and included message definition files are in different directories within the message model.

You can fix problem from the workbench by selecting the **Properties** tab of the including message definition file in the Message Definition Editor. Delete the include entry with the incorrect Schema Location then add another include entry, selecting the correct file from the file menu.

Message sets without namespace support

- Constructs not accepted as they stand when importing an XML schema.

When importing an XML Schema into a message set that does not support namespaces, the following constructs are not accepted as they stand:

- Redefine
- Import
- List
- Union
- Abstract complex type
- Abstract element

The user has the choice to reject or modify these constructs based on the preferences set by the user in the XML Schema Importer preference page (or the **mqscreatemsgdefs** import options file). If the user specifies "reject", when the construct is encountered, the import stops and an error is reported. If the user specifies "modify", when the construct is encountered, the importer modifies the resulting message definition file as follows:

- Redefine statements are ignored and not imported.
- Import statements are replaced with include statements.
- Any list/union types are modified to be a restriction with a base type `xsd:string`.
- Abstract complex types are changed into non-abstract complex types.
- Abstract elements are changed into non abstract elements.

Note that these modifications are the equivalent modifications that were performed by the WMQI 2.1 XML Schema importer command, for compatibility.

The following is an example of each of these constructs:

```

<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.ibm.com" xmlns:ibm="http://www.ibm.com"/>

  <!-- Unsupported feature: redefine -->
  <redefine schemaLocation="test.xsd"/>

  <!-- Unsupported feature: import -->
  <import schemaLocation="test1.xsd" namespace="http://www.ibm1.com"/>

  <!-- Unsupported feature: list -->
  <simpleType name="type1">
    <list itemType="string" />
  </simpleType>

```

```

</simpleType>

<!-- Unsupported feature: union -->
<simpleType name="type2">
  <union memberTypes="string" />
</simpleType>

<!-- Unsupported feature: abstract complex type -->
<complexType name="type3" abstract="true"></complexType>

<!-- Unsupported feature: abstract element -->
<element name="element" type="string" abstract="true"></element>

</schema>

```

Unique, Key and Keyref constructs are imported and appear in the generated message definition file but, since these constructs are not supported, error task entries are created to inform the user that the generated .mxsd file has problems.

- Target namespaces not qualified with a prefix.

When importing an XML schema into a message set that does not support namespaces, you cannot import a schema document that has a target namespace that is not qualified with a prefix. For example:

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.ibm.com" xmlns="http://www.ibm.com">
</xsd:schema>

```

Importing from XML DTD: unsupported features

The same limitations that apply to importing an XML Schema also apply to importing an XML DTD. The DTD importer converts XML DTD to an XML schema and then the schema is imported. If the generated XML Schema happens to have any of the XML Schema limitations then the behavior is as described in “Importing from XML Schema: unsupported features” on page 548.

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this information in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this information. The furnishing of this information does not give you any license to these patents. You can send license inquiries, in writing, to:

- IBM Director of Licensing
- IBM Corporation
- North Castle Drive
- Armonk, NY 10504-1785
- U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

- IBM World Trade Asia Corporation
- Licensing
- 2-31 Roppongi 3-chome, Minato-ku
- Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

- IBM United Kingdom Laboratories,
- Mail Point 151,
- Hursley Park,
- Winchester,
- Hampshire,
- England
- SO21 2JN

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information includes examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX	CICS	DB2
DB2 Universal Database	developerWorks	
Everyplace	FFST	First Failure Support Technology
IBM	IMS	IMS/ESA
iSeries	Language Environment	MQSeries
MVS	NetView	OS/400
OS/390	pSeries	RACF
RETAIN	RS/6000	SupportPac
Tivoli	VisualAge	WebSphere
xSeries	z/OS	zSeries

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Pentium is a registered trademark of Intel.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.



Printed in USA