

# **Gentran:Server<sup>®</sup> for Windows<sup>®</sup>**

## **Forms Integration User's Guide**

Version 5.2

**Sterling Commerce**  
An IBM Company

## Copyright Notice

### **Gentran:Server for Windows**

© Copyright 1995–2008  
Sterling Commerce, Inc.  
ALL RIGHTS RESERVED

### **Sterling Commerce Software** Trade Secret Notice

**THE GENTRAN:SERVER FOR WINDOWS SOFTWARE ("STERLING COMMERCE SOFTWARE") IS THE CONFIDENTIAL AND TRADE SECRET PROPERTY OF STERLING COMMERCE, INC., ITS AFFILIATED COMPANIES OR ITS OR THEIR LICENSORS, AND IS PROVIDED UNDER THE TERMS OF A LICENSE AGREEMENT. NO DUPLICATION OR DISCLOSURE WITHOUT PRIOR WRITTEN PERMISSION. RESTRICTED RIGHTS.**

This documentation, the Sterling Commerce Software it describes, and the information and know-how they contain constitute the proprietary, confidential and valuable trade secret information of Sterling Commerce, Inc., its affiliated companies or its or their licensors, and may not be used for any unauthorized purpose, or disclosed to others without the prior written permission of the applicable Sterling Commerce entity. This documentation and the Sterling Commerce Software that it describes have been provided pursuant to a license agreement that contains prohibitions against and/or restrictions on their copying, modification and use. Duplication, in whole or in part, if and when permitted, shall bear this notice and the Sterling Commerce, Inc. copyright notice.

As and when provided to any governmental entity, government contractor or subcontractor subject to the FARs, this documentation is provided with RESTRICTED RIGHTS under Title 48 CFR 52.227-19. Further, as and when provided to any governmental entity, government contractor or subcontractor subject to DFARs, this documentation and the Sterling Commerce Software it describes are provided pursuant to the customary Sterling Commerce license, as described in Title 48 CFR 227-7202 with respect to commercial software and commercial software documentation.

These terms of use shall be governed by the laws of the State of Ohio, USA, without regard to its conflict of laws provisions. If you are accessing the Sterling Commerce Software under an executed agreement, then nothing in these terms and conditions supersedes or modifies the executed agreement.

Product names mentioned herein may be trademarks and/or registered trademarks of their respective companies. Gentran and Gentran:Server are registered trademarks of Sterling Commerce, Inc.

### **Third Party Software:**

Portions of the Sterling Commerce Software may include products, or may be distributed on the same storage media with products, ("Third Party Software") offered by third parties ("Third Party Licensors").

### **Warranty Disclaimer**

This documentation and the Sterling Commerce Software which it describes are licensed either "AS IS" or with a limited warranty, as set forth in the Sterling Commerce license agreement. Other than any limited warranties provided, NO OTHER WARRANTY IS EXPRESSED AND NONE SHALL BE IMPLIED, INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR USE OR FOR A PARTICULAR PURPOSE. The applicable Sterling Commerce entity reserves the right to revise this publication from time to time and to make changes in the content hereof without the obligation to notify any person or entity of such revisions or changes.

The Third Party Software is provided 'AS IS' WITHOUT ANY WARRANTY AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. FURTHER, IF YOU ARE LOCATED OR ACCESSING THIS SOFTWARE IN THE UNITED STATES, ANY EXPRESS OR IMPLIED WARRANTY REGARDING TITLE OR NON-INFRINGEMENT ARE DISCLAIMED.

---

# Table of Contents

Preface	About This Guide	
	• Introduction .....	x
	• What's in this Manual .....	xi
	• Description of Contents .....	xii
	• Online Help .....	xiii
	• Getting Support .....	xiv
<b>Chapter 1</b>	<b>Form Integration Basics</b>	
	• Introduction .....	1-2
	• Introducing Forms Integration .....	1-3
	• Forms Integration Subsystem .....	1-4
	• Building a Form with Forms Integration .....	1-10
<b>Chapter 2</b>	<b>Designing Your Form</b>	
	• Introduction .....	2-2
	<b>Getting Started</b> .....	<b>2-3</b>
	• Preparation and Analysis .....	2-3
	<b>Setting Global Defaults</b> .....	<b>2-4</b>
	• Overview .....	2-4
	• Global Default Preference Settings .....	2-6
	• Customizing Display, Colours and Fonts .....	2-16
	• Customizing Display of EDI File Format and Layout Windows .....	2-19
	• Working With the Default Date Format .....	2-20
	<b>Creating a New Form</b> .....	<b>2-23</b>
	• Overview .....	2-23
	• Creating a Form and Defining Form Details .....	2-24
	• Activating Form Components .....	2-29
<b>Chapter 3</b>	<b>Modifying Form Components</b>	
	• Introduction .....	3-2
	<b>Working with the EDI File Format</b> .....	<b>3-3</b>
	• Modifying an EDI File Format .....	3-3
	• Promoting Groups and Repeating Segments .....	3-4
	• Splitting Groups and Repeating Segments .....	3-5
	• Using Cut, Copy, and Paste .....	3-6

- Verifying EDI Delimiters ..... 3-7
    - Saving a File Definition ..... 3-8
  - Working with Groups ..... 3-9**
    - Creating a Group ..... 3-9
    - Modifying Group Properties ..... 3-11
  - Working with Segments ..... 3-14**
    - Modifying Segment Properties ..... 3-14
    - Using the LS and LE Segments ..... 3-18
  - Working with Composites ..... 3-23**
    - Modifying Composite Properties ..... 3-23
  - Working with Elements ..... 3-25**
    - Overview ..... 3-25
    - Formatting Numbers ..... 3-26
    - Formatting Dates and Times ..... 3-29
    - Formatting Strings ..... 3-33
    - Creating a Syntax Token (Western European languages) ..... 3-34
    - Creating a Syntax Token (Far Eastern languages) ..... 3-38
    - Editing a Syntax Token ..... 3-40
    - Deleting a Syntax Token ..... 3-42
    - Deleting a character range from a syntax token ..... 3-43
    - Using Syntax Tokens ..... 3-44
    - Modifying Element Properties ..... 3-46
    - Storing EDI Code Value Descriptions ..... 3-50
    - Defining Relational Conditions ..... 3-52
- Chapter 4 Completing Your Form**
  - Overview ..... 4-2
  - Compiling a Translation Object ..... 4-3
  - Printing a Report ..... 4-6
  - Registering a Translation Object ..... 4-7
  - Creating a Trading Partnership ..... 4-8
  - Validating a Translation Object ..... 4-9
- Chapter 5 Using Standard Rules**
  - Introduction ..... 5-3
  - Using the Select Standard Rule ..... 5-4**
    - Overview ..... 5-4
    - Using Select in a Form ..... 5-6
    - Using Information from the Partner Definition ..... 5-11
    - Using Information from Location Tables ..... 5-14
    - Using Information from Lookup Tables ..... 5-16
    - Using Information from Cross-Reference Tables ..... 5-18

<b>Using the Update Standard Rule</b> .....	<b>5-20</b>
▶ Overview .....	5-20
▶ Using Update in a Form .....	5-22
▶ Update Example: Document Name .....	5-24
<b>Using the Use System Variable Standard Rule</b> .....	<b>5-27</b>
▶ Overview .....	5-27
▶ Using a System Variable in a Form .....	5-28
▶ Using the System Date and Time .....	5-30
<b>Using the Use Constant Standard Rule</b> .....	<b>5-32</b>
▶ Overview .....	5-32
▶ Using Constants in a Form .....	5-34
▶ Use Constant Example .....	5-36
<b>Using Literal Constants and Qualifying Relationships</b> .....	<b>5-40</b>
▶ Overview .....	5-40
▶ Defining Literal Constants .....	5-41
▶ Editing Literal Constants .....	5-43
▶ Deleting Literal Constants .....	5-44
▶ Using Literal Constants .....	5-45
▶ Generating Qualifiers .....	5-47
<b>Using the Use Accumulator Standard Rule</b> .....	<b>5-49</b>
▶ Overview .....	5-49
▶ Using an Accumulator in a Form .....	5-51
▶ Use Accumulator Example: Counting Line Items .....	5-54
▶ Use Accumulator Example: Calculating Hash Totals .....	5-61
▶ Use Accumulator Example: Resetting and Calculating a Value Total .....	5-64
<b>Using the Loop Count Standard Rule</b> .....	<b>5-73</b>
▶ Overview .....	5-73
▶ Using Loop Count in a Form .....	5-75
<b>Using the Use Code Standard Rule</b> .....	<b>5-77</b>
▶ Overview .....	5-77
▶ Using Use Code in a Form .....	5-80
▶ Use Code Example: Selecting a Code List Item .....	5-83
<b>Using Code List Tables</b> .....	<b>5-86</b>
▶ Overview .....	5-86
▶ Defining a Code List .....	5-88
▶ Modifying a Code List .....	5-90
▶ Deleting a Code List or Code List Entry .....	5-92
▶ Importing a Code List .....	5-95
▶ Exporting a Code List .....	5-97
▶ Loading a Code List Table from the Standard .....	5-99
▶ Copying and Pasting Code Lists .....	5-102

	Validating Data Against Code List Tables .....	5-104
	Loading Code Item Descriptions .....	5-106
<b>Chapter 6</b>	<b>Using Extended Rules</b>	
	<b>Using Extended Rules .....</b>	<b>6-4</b>
	Overview .....	6-4
	Declarations Section .....	6-5
	Statements Section .....	6-6
	When Rules are Processed .....	6-7
	<b>Defining Extended Rules .....</b>	<b>6-9</b>
	Overview .....	6-9
	Defining a Session Rule .....	6-10
	Defining a Form Component Extended Rule .....	6-11
	<b>Extended Rule Syntax .....</b>	<b>6-13</b>
	Overview .....	6-13
	Keywords and Commands .....	6-14
	Operators .....	6-16
	Symbols .....	6-17
	<b>Common Statements and Examples .....</b>	<b>6-20</b>
	Overview .....	6-20
	Assignment .....	6-21
	Datetime Expressions .....	6-22
	Conditional Logic .....	6-24
	String Conditions and Functions .....	6-25
	Numerical Functions .....	6-27
	Raise Compliance Error Function (CERROR) .....	6-28
	Remove Field Value Function (EMPTY) .....	6-29
	Existence of Data Function (EXIST) .....	6-30
	Count Function (COUNT) .....	6-31
	Delete Function (DELETE) .....	6-32
	File Pointer Functions (FSEEK, FTELL) .....	6-33
	Block of Data Functions (READBLOCK, WRITEBLOCK) .....	6-34
	Select Function (SELECT) .....	6-35
	Update Function (UPDATE) .....	6-36
	Insert Function (INSERT) .....	6-37
	User Exit Function (EXEC) .....	6-38
	ActiveX and User Exit Functions .....	6-39
	<b>Alphabetic Language Reference .....</b>	<b>6-41</b>
	Overview .....	6-41
	atoi .....	6-42
	atol .....	6-43
	auditlog .....	6-44

▶ begin	6-46
▶ break	6-47
▶ cerror	6-48
▶ concat	6-55
▶ continue	6-56
▶ count	6-57
▶ createobject	6-58
▶ date	6-59
▶ delete	6-61
▶ deleteobject	6-62
▶ empty	6-63
▶ end	6-64
▶ exec	6-65
▶ exist	6-66
▶ fseek	6-67
▶ ftell	6-68
▶ get	6-69
▶ getiid	6-70
▶ if then...else	6-71
▶ index	6-72
▶ insert	6-73
▶ left	6-74
▶ len	6-75
▶ messagebox	6-76
▶ mid	6-78
▶ ntoa	6-79
▶ param	6-80
▶ queryobject	6-81
▶ readblock	6-82
▶ readbytes	6-83
▶ right	6-84
▶ select	6-85
▶ set	6-86
▶ strdate	6-87
▶ strstr	6-89
▶ unreadblock	6-90
▶ update	6-91
▶ while...do	6-92
▶ winexec	6-93
▶ writeblock	6-95
▶ writebytes	6-96
▶ Select and Update Available Options	6-97

<b>Chapter 7</b>	<b>Customizing Screen Entry and Print Forms</b>	
	Introduction	7-1
	<b>Screen Entry and Print Forms</b>	<b>7-2</b>
	Customizing Field Labels	7-2
	Hiding Fields	7-3
	<b>Screen Entry Forms</b>	<b>7-4</b>
	Setting an Initial Value for a Field	7-4
	Using a Constant Value for a Field	7-7
	Creating Definitions for a List Box	7-9
	Creating Help Text for Fields and List Boxes	7-11
	Creating Help Text for Frames	7-13
	Formatting Entry Fields	7-14
	Formatting Display-Only (Non-Editable) Fields	7-15
<b>Chapter 8</b>	<b>Formatting Screen Entry and Print Forms</b>	
	Introduction	8-2
	<b>Selecting and Grouping Fields</b>	<b>8-3</b>
	Selecting One or More Fields or Field Labels	8-3
	Grouping Fields and Field Labels	8-5
	<b>Resizing Fields</b>	<b>8-6</b>
	Overview	8-6
	Using Size to Length	8-7
	Resizing a Field Manually	8-8
	Resizing Groups of Fields Manually	8-9
	<b>Arranging Fields</b>	<b>8-10</b>
	Overview	8-10
	Changing Grid Settings	8-11
	Using Alignment Lines	8-13
	Moving Fields and Field Labels	8-14
	Aligning Fields and Field Labels	8-15
	Spacing Fields and Field Labels	8-17
	Resizing and Positioning a Frame	8-18
	Resizing Drop-Down Combo Boxes	8-19
	Setting Tab Sequences	8-21
	Adding and Positioning Static Text	8-22
	Modifying Frame Titles	8-23
	Modifying List Box Names	8-24
	Adding Column Headings to List Boxes	8-25



## Appendix A Error Messages

- ▶ Overview ..... A-2
- ▶ Compile Error Messages ..... A-3
- ▶ Gentran:Server Error Messages ..... A-16

## Appendix B User Exits

- ▶ Overview ..... B-2
- ▶ Definition of ActiveX Terminology ..... B-3
- ▶ Gentran:Server User Exit Overview ..... B-4
- ▶ Examples of User Exits ..... B-7
- ▶ Examples of Automation Servers ..... B-12
- ▶ How to Create a User Exit ..... B-15

## Appendix C Using the Translator Command Line Interface

- ▶ Introduction ..... C-2
- ▶ Command Line Syntax ..... C-3

## Glossary



# About This Guide

---

## Contents

▶ Introduction . . . . .	x
▶ What's in this Manual . . . . .	.xi
▶ Description of Contents. . . . .	xii
▶ Online Help . . . . .	xiii
▶ Getting Support . . . . .	.xiv

---

# Introduction

---

**Welcome**

Welcome to Gentran:Server® Windows® Forms Integration subsystem, Sterling Commerce's Electronic Data Interchange (EDI) form-generation software for the Windows operating system. Gentran:Server provides you with the tools that you need to design forms to facilitate the keying and printing of EDI documents.

---

## What's in this Manual

---

**Introduction** This manual provides step-by-step instructions to assist you in performing the various tasks associated with the Forms Integration subsystem. This approach is intended to answer any questions you may have about Forms Integration.

---

**Intended audience** This manual is intended for the following users:

- Technical users— users with a detailed knowledge of the Windows operating system, networking, databases, and communications.
- Novice-to-average experienced users—users who are new to Gentran:Server for Windows, EDI, or both.
- EDI Coordinators—users responsible for creating and maintaining the transfer of EDI data between Trading Partners.

---

**Prerequisite knowledge** The audience using this software should be familiar with:

- PC and Microsoft® Windows functions
- EDI concepts
- EDI standard structure
- the internal application format
- data mapping concepts
- Gentran:Server product

---

## Description of Contents

---

**Introduction** The “Forms Integration User’s Guide” contains the tasks associated with using the Forms Integration subsystem.

---

**Organization of chapters** The “Forms Integration User’s Guide” is organized into the following chapters.

- About This Guide explains the guide’s content and organization. This chapter also introduces the online Help system and how to get technical support.
- Forms Integration Basics, chapter 1, explains basic Form Integration terms and concepts.
- Designing Your Forms, chapter 2, explains how to perform basic Form Integration functions, including: setting global display options, creating a new form; modifying form components; working with groups, segments, composites, and elements; and completing the form.
- Modifying Form Components, chapter 3, describes how to modify form components, including the EDI File Format, groups, segments, composites, and elements.
- Completing Your Form, chapter 4, describes how to complete your form, which includes compiling the translation object and printing a report.
- Using Standard Rules, chapter 5, explains how to use the Gentran:Server standard rules. This chapter also provides example scenarios of when to use certain standard rules.
- Using Extended Rules, chapter 6, defines what extended rules are, when they are used, and the proprietary programming language syntax.
- Customizing Screen Entry and Print Forms, chapter 7, describes how to customize Screen Entry and Print Translation Object attributes. Topics include: Customizing Field Labels, Hiding Field Labels, and Storing EDI Code Value Descriptions.
- Formatting Screen Entry and Print Forms, chapter 7, explains how to format, organize, and size translation object fields.
- Error Messages, Appendix A, describes Compile and Gentran:Server error messages and corrective actions.
- User Exits, Appendix B, describes how to use extended rules to take advantage of Gentran:Server user exit functionality.
- Using the Translator Command Line Interface, Appendix C, describes how to use the command line to invoke the services of the TXDE.EXE translator to perform translation outside Gentran:Server.
- The Glossary describes Gentran:Server and Forms Integration terminology and concepts.

---

# Online Help

---

**Introduction**

The majority of the Gentran:Server Forms Integration documentation found in this manual is also contained in the online Help system. This includes all dialog box element definitions, detailed processing information, and how-to information.

---

**Field-level Help**

To view field-level descriptions for the Forms Integration subsystem, navigate to the component for which you want field-level descriptions. Press **F1** to display a parts and functions table.

---

# Getting Support

**Introduction** Sterling Commerce's Gentran:Server software is supported by trained product support personnel who are available to help you with product questions or concerns.

**Note**

Gentran:Server Customer Support does not support non-Sterling Commerce products (e.g., SQL Server, Oracle, etc.), but can assist you in configuring non-Sterling Commerce products to work with Gentran:Server.

**Phone number** For assistance, please refer to your *Getting Started Guide* to determine which support phone number you should use.

**Before calling support**

To help us provide prompt service, we ask that you do the following:

- Attempt to recreate any problem that you encounter and record the exact sequence of events.
- When you call product support, you should be prepared to provide us with the information below.

Information	Description
Identification	Your company name, your name, telephone number and extension, and the case number (if the question refers to a previously reported issue).
System Configuration	The Gentran:Server version (and any service packs installed) and information about the primary Gentran system controller and all machines experiencing problems, including: the Windows operating system version, amount of memory, available disk space, database version, Microsoft Data Access (MDAC) version, and Internet Explorer version.  Also, please describe any recent changes in your hardware, software, or the configuration of your system.
System Data Store	Which machines contain folders in the system data store?
Error Messages	Record the exact wording of any error messages you receive and the point in the software where the error occurred, as well as any log files.
Attempted Solutions	Record any steps that you took attempting to resolve the problem and note all the outcomes, and provide an estimate on how many times the problem occurred and whether it can be reproduced.



---

**Accessing the Sterling Commerce Support Web Site**

The Sterling Commerce Customer Support Web Site contains valuable information about getting support for Gentran:Server for Windows, including the:

- scope of support services
- customer support policies
- call prioritizing
- customer support phone directory
- how to create new Support on Demand cases
- how to check the status of Support on Demand cases
- how to add information to Support on Demand cases

The Customer Support Web Site is constantly updated and all Sterling Commerce customers have access to it. This web site also contains the most recent product updates and is a valuable source of product information.

**Reference**

Refer to the *Getting Started Guide* for information about how to access the Customer Support World Wide Web Site.

---

**Documentation**

The Customer Support Web Site contains a documentation library, which has the entire Gentran:Server for Windows documentation set. You can download the product manuals in PDF format from this library at any time.

---



---

# Form Integration Basics

---

**Contents**

- Introduction . . . . . 1 - 2
  - Introducing Forms Integration . . . . . 1 - 3
  - Forms Integration Subsystem . . . . . 1 - 4
  - Building a Form with Forms Integration. . . . . 1 - 10
-

# Introduction

---

**In this chapter** This chapter describes basic concepts and functionality associated with Forms Integration.

---

# Introducing Forms Integration

**Introduction** The Gentran:Server Forms Integration subsystem enables you to design forms to facilitate the keying (screen entry) and printing (print) of EDI documents that you receive inbound or send outbound.

**What is a form?** A form is a set of instructions that you define in the Forms Integration subsystem to indicate how the system should format data. When you compile the form, Gentran:Server generates a translation object. After you register the translation object with Gentran:Server and associate the translation object with a partner relationship, the system uses the translation object to format EDI documents.

**Types of Forms** This table describes the two types of forms that you can create using Forms Integration.

Form	Description
Print translation object	Organizes and formats the printout of EDI documents that are received from or sent to the trading partners for which you have established a trading relationship (inbound or outbound) that utilizes that print translation object. The print translation object enables you to view the EDI document in an easily readable format.
Screen entry translation object	Provides a standardized format for keying an EDI document into the Document Editor, for translation and transmission to your trading partners, for which you have established an outbound trading relationship that utilizes that screen entry translation object. The screen entry translation object ensures that your users key all the data necessary to create the required EDI document.

## Forms Integration Subsystem

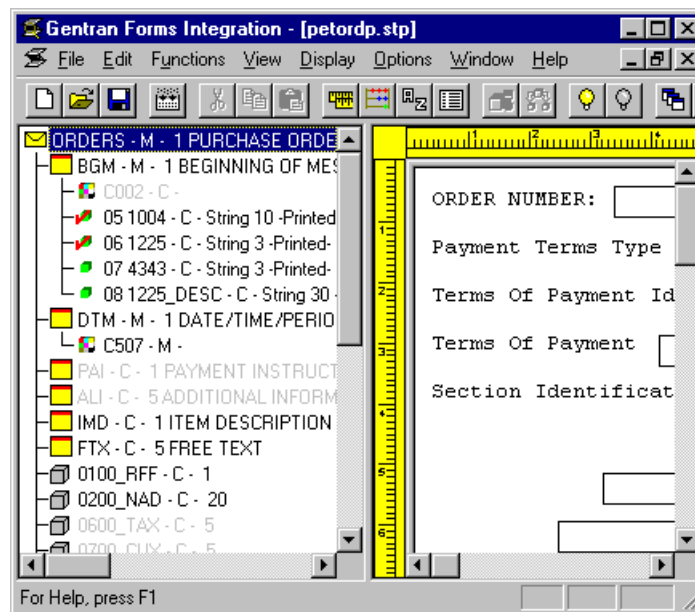
### Introduction

The Forms Integration subsystem contains two windows: The EDI File Format Window (left side of the screen) and the Layout Window (right side of the screen).

When you click a segment or group in the EDI File Format Window, the corresponding frame is displayed in the Layout Window. When you click an element in the EDI File Format Window, the corresponding layout component is highlighted in the Layout Window.

### Main Window

This illustration shows the Forms Integration Subsystem Main Window.



### Parts and Function

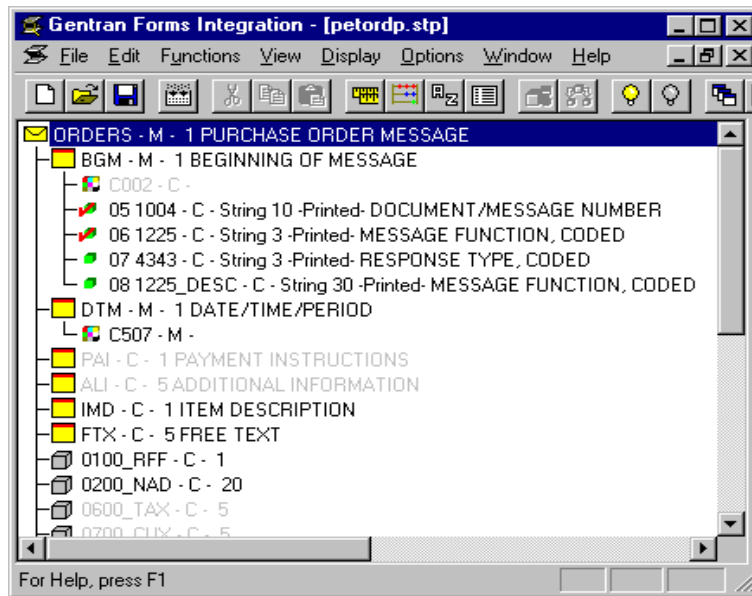
This table describes the Forms Integration Main Window parts and their functions. .

Window	Description
Title Bar	Displays the title of the open form.
Main Menu Bar	Contains drop-down menus of Form Integration commands.
Main Toolbar	Provides short-cut options to common Form Integration components.
The EDI File Format Window (left side of the Forms Integration subsystem main window)	Contains the EDI file format. (Continued on next page)

Window	Description
Title Bar	Displays the title of the open form.
Main Menu Bar	Contains drop-down menus of Form Integration commands.
Main Toolbar	Provides short-cut options to common Form Integration components.
The Layout Window (right side of the Forms Integration subsystem main window)	Contains the actual format of the translation object.




**EDI File Format Window**

This illustration shows the EDI File Format Window.



## EDI File Format Window Parts and Functions

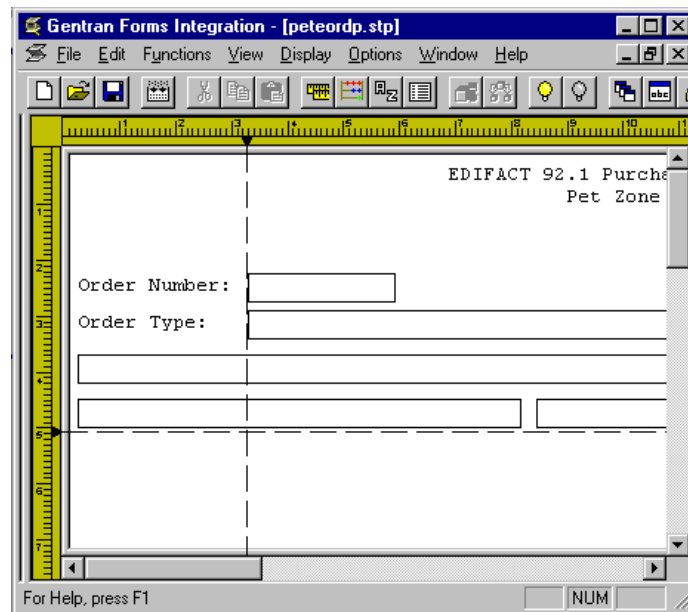
This table describes the parts of the EDI File Format window and their functions.

Icon	Description
	<p>A <b>group</b> is a looping structure that contains related segments and/or groups that repeat in sequence until either the group data ends or the maximum number of times that the loop is allowed to repeat is exhausted.</p> <p>Groups are defined by the EDI standards. A group that is subordinate to another group is a subgroup (this corresponds to a nested looping structure – a loop within a loop).</p>
	<p>A <b>segment</b> contains a group of related elements or composite data elements that combine to communicate useful data. Segments are defined by the EDI standards. A segment can occur once or can repeat multiple times.</p>
	<p>A <b>composite</b> is a data element that contains two or more component data elements or subelements. Composites are defined by the EDI standards that use them (EDIFACT, TRADACOMS, and certain ANSI X12 standards).</p> <p>A <b>repeating composite</b> is a related group of EDI subelements that have the ability to loop as a whole (occur more than once) within a particular EDI segment. To allow a composite to repeat multiple times within a segment, each occurrence of the composite must be separated by a special delimiter, known as the repeating element delimiter.</p> <p><b>Example</b> Your delimiters are set as follows:</p> <p>Segment        ~ Element        * Tag             * Sub Element    : Repeating Ele ^</p> <p>Then, if a BGM segment contains a repeating composite that occurs 3 times within the data followed by 1 regular element, and the composite contains 2 subelements, the segment is in the following form:</p> <pre>BGM*SubA-1:SubB-1^SubA-2:SubB-2^SubA-3:SubB-3*Field2~</pre> <p style="text-align: right;">(Continued on next page)</p>



<b>(Contd) Icon</b>	<b>Description</b>
■	<p>An <b>element</b> is the smallest piece of information defined by the EDI standards. An individual element can have different meanings depending on the context. Therefore, elements are normally not considered to have useful meaning until they are combined into segments.</p> <p>An element is the EDI map component that is mapped (linked) to a corresponding application field to move data to and from the EDI file.</p> <p>A <b>repeating element</b> is an EDI elements with the ability to loop (occur more than once) within a particular EDI segment. To allow a single element to repeat multiple times within a segment, each element must be separated by a special delimiter, known as the repeating element delimiter. The use of this delimiter prevents the system from mistaking the repeating elements for normal elements.</p> <p><b>Example</b> Your delimiters are set as follows:</p> <p>Segment        ~ Element        * Tag             * Sub Element    : Repeating Ele ^</p> <p>Then, if a BGM segment contains 3 elements and the second element is a repeating element that occurs 4 times, the segment is in the following form:</p> <p>BGM*Field1*Field2.1^Field2.2^Field2.3^Field2.4*Field3~</p>

**Layout Window** This illustration shows the Layout Window.



**Layout Window  
Parts and  
Functions**

This table describes the parts of the Layout window and their functions.

Part	Function
Horizontal and Vertical Ruler	Enables you to manipulate the length, width, and alignment of your form.
Field Label <b>Note</b> In the Layout Window illustration, Order Number: is an example of a field label.	Defines the field in which a user enters information.
Field	Text box associated with a field label in which a user can enter information. For print translation objects, groups, repeating segments, and elements are always formatted as fields.
Frame	Used to format the EDI file for the translation object. Each frame contains the groups, repeating segments, and elements at that level (single segments are not represented on the frame).
Alignment Lines	Vertical or horizontal grid lines that enable you align the components of the Layout Window.

---

**List formats**

For screen entry translation objects, groups (with a maximum usage greater than one) and repeating segments are formatted as lists. Elements are formatted as edit boxes (fields) or lists on the frame. Elements are typically formatted as fields, but are formatted as a list if you apply a standard rule that allows a selection of multiple items. Each group and repeating segment also has a corresponding frame that contains all of the groups (with a maximum usage greater than one), repeating segments, and elements at that level.

---

# Building a Form with Forms Integration

## Introduction

This section defines how to proceed with building a form using the Gentran:Server Forms Integration subsystem.

## Stages of building a form

This table describes the process to build a form using Forms Integration.

Stage	Description
1	<p>Prepare and Analyze how you want your form to look and what components your form will need.</p> <p><b>Reference</b> See <i>Preparation and Analysis</i> on page 2 - 3 for more information.</p>
2	<p>Set Global Defaults (first time only).</p> <p><b>Reference</b> See <i>Global Default Preference Settings</i> on page 2 - 6 for more information.</p>
3	<p>Create, Save, and Name a New Form.</p> <p><b>Reference</b> See the <i>Creating a Form and Defining Form Details</i> on page 2 - 24 for more information.</p>
4	<p>Activate the Appropriate EDI Groups, Segments, and Elements.</p> <p><b>Reference</b> Please see <i>Activating Form Components</i> on page 2 - 29 for more information.</p>
5	<p>Define the Layout of the Form.</p> <p><b>Reference</b> See <i>Modifying an EDI File Format</i> on page 3 - 3 for more information on customizing your EDI file, including setting the EDI delimiters.</p> <p>See Using Standard Rules, chapter 5, of this guide for more information.</p> <p>See Using Extended Rules, chapter 6, of this guide for more information.</p>
6	<p>Compile the Translation Object.</p> <p><b>Reference</b> See <i>Compiling a Translation Object</i> on page 4 - 3 for more information on compiling the translation object and translation object naming conventions.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Stage</b>	<b>Description</b>
7	<p>Print the Report.</p> <p><b>Reference</b> See <i>Printing a Report</i> on page 4 - 6 for more information.</p>
8	<p>Register the Translation Object with Gentran:Server.</p> <p><b>Reference</b> See the <i>Gentran:Server for Windows User's Guide</i> for more information about registering translation objects.</p>
9	<p>Create the Appropriate Trading Relationship. Establish the appropriate trading relationship in Gentran:Server for your trading partners.</p> <p><b>Reference</b> See the <i>Gentran:Server for Windows User's Guide</i> for more information about creating trading relationships.</p>
10	<p>Validate the Translation Object. For a screen entry translation object, create a new document in Document Editor, using that translation object. For a print translation object, print the document. Verify that the output is correct.</p> <p><b>Reference</b> See the <i>Gentran:Server for Windows User's Guide</i> for more information about creating a new document in the Document Editor using that translation object.</p>



---

# Designing Your Form

---

<b>Contents</b>	<ul style="list-style-type: none"> <li>▶ Introduction ..... 2 - 2</li> </ul>
	<b>Getting Started ..... 2 - 3</b>
	<ul style="list-style-type: none"> <li>▶ Preparation and Analysis ..... 2 - 3</li> </ul>
	<b>Setting Global Defaults ..... 2 - 4</b>
	<ul style="list-style-type: none"> <li>▶ Overview ..... 2 - 4</li> <li>▶ Global Default Preference Settings ..... 2 - 6</li> <li>▶ Customizing Display, Colours and Fonts ..... 2 - 16</li> <li>▶ Customizing Display of EDI File Format and Layout Windows ..... 2 - 19</li> <li>▶ Working With the Default Date Format ..... 2 - 20</li> </ul>
	<b>Creating a New Form ..... 2 - 23</b>
	<ul style="list-style-type: none"> <li>▶ Overview ..... 2 - 23</li> <li>▶ Creating a Form and Defining Form Details ..... 2 - 24</li> <li>▶ Activating Form Components ..... 2 - 29</li> </ul>

---

## Introduction

---

**In this chapter** This chapter describes how to design your form.

---



# Getting Started

## Preparation and Analysis

---

**Introduction**

Before you begin to create a form, we recommend that you work with your trading partner to:

- determine necessary form fields and labels
  - obtain a layout of how you want the completed form to look
  - determine how your layout corresponds with the EDI standard you use
  - determine the operations that you need perform to generate the required layout
-

# Setting Global Defaults

## Overview

**Introduction** This section describes Forms Integration Global default Preference and Date/Time settings.

**Preference Defaults** The Preferences dialog box is a property sheet that enables you to set global Form Integration defaults. This table describes the customizable global preference options that you can define on the Preferences dialog box to control the look of your form display

**Note**

Form display options can be set or changed at any time.

Customizable Items	Description
Tree settings (Global option)	<p>Enables you to display group, record, and field descriptions.</p> <p>You can also access:</p> <ul style="list-style-type: none"> <li>▶ Colours (Global option) - Enables you to select foreground and background colours to visually define form components. The use of colour is optional.</li> <li>▶ Fonts (Global option)- Enables you to globally change how fonts display. This includes the typeface, style, and size. The default font that Gentran:Server uses is a Sans Serif 9 point.</li> </ul>
Standard Formats	<p>Displays the default date to use when elements are read from the Standards database.</p> <p>Valid options are:</p> <ul style="list-style-type: none"> <li>▶ Six-character dates</li> <li>▶ Eight-character dates.</li> </ul>
Positional Defaults	Defines default field formats.
Files	Defines how Forms Integration works with files.
Layout	Defines the look of the layout window.
Confirmations	Controls when you receive a confirmation prompt.
Version	Controls how and when the system increments a form's version number.

---

**Date/Time Format**

The Default Date formats dialog box, available from the Options menu, enables you to Control the default date and time formats that are used when elements are read from the standards database.

---

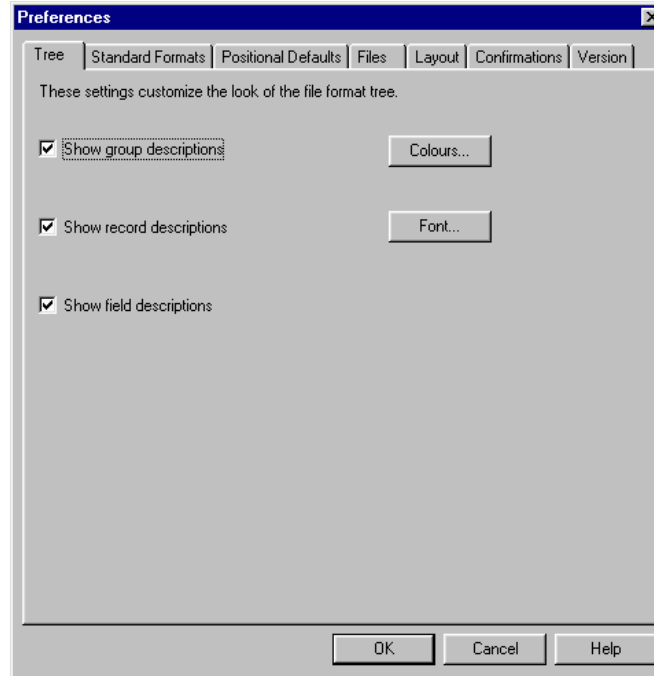
# Global Default Preference Settings

## Introduction

Global defaults are defined on the Preferences dialog box. This section describes each tab of that dialog box.

## Preferences dialog box (Tree tab)

This illustration describes the Tree tab of the Preferences dialog box.



## Parts and Functions

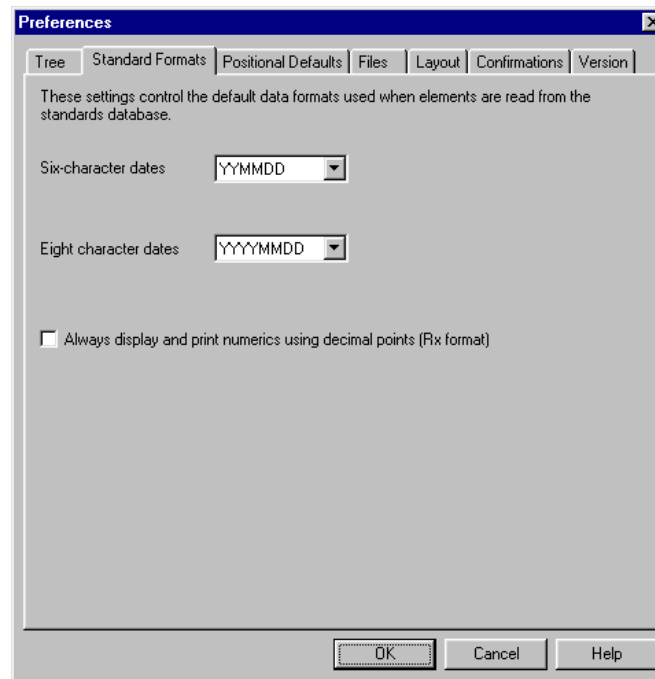
This table describes the parts and functions of the Tree tab on the Preferences dialog box.

Part	Function
Descriptions check boxes	<p>Defines what form details that you want to display in the file format tree.</p> <p>Options are:</p> <ul style="list-style-type: none"> <li>▶ Show group descriptions</li> <li>▶ Show record descriptions</li> <li>▶ Show field descriptions</li> </ul>
Colours	<p>Accesses the Colours dialog box, which enables you to select foreground and background colours to visually define form items and/or attributes. The use of colour is optional.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>Part</b>	<b>Function</b>
<b>Fonts</b>	Accesses the Fonts dialog box, which enables you to globally change how fonts display. Customizable font options include the typeface, style, and size. The default font that Gentran:Server uses is a Sans Serif 9 point.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Help</b>	Launches the Forms Integration online Help system.

**Preferences dialog box  
(Standard Formats tab)**

This This illustration describes the Standard Formats tab of the Preferences dialog box.



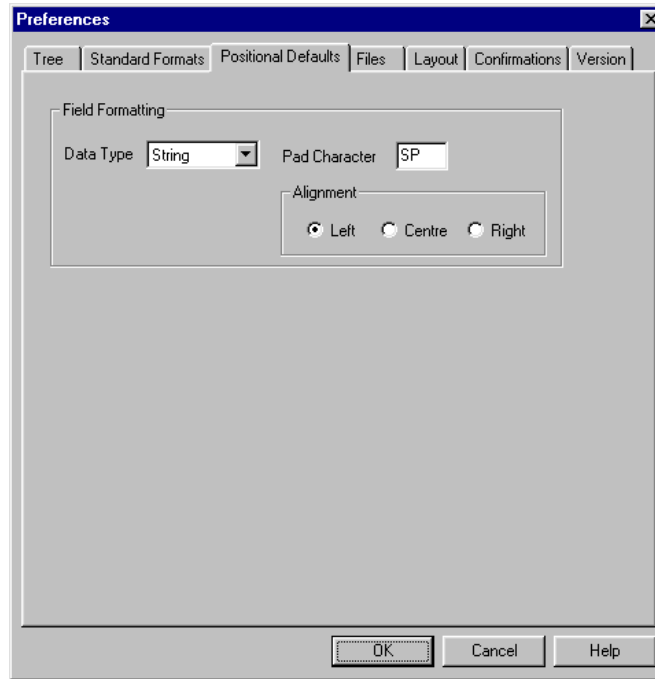
**Parts and Functions**

This table describes the parts and functions of the Standard Format tab on the Preferences dialog box.

Part	Function
Date lists	Controls the default date formats that are used when the element is read from the Standards database.  Customizable items include: <ul style="list-style-type: none"> <li>▶ Six-character dates</li> <li>▶ Eight-character dates</li> </ul>
Always display and print numerics using decimal points check box	Specifies that Gentran:Server should display and print all numeric-format elements using decimal points (formatted as real). This default data format is used when elements are read from the standards database.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Help</b>	Launches the Forms Integration online Help system.

**Preferences dialog box (Positional Default tab)**

This illustration describes the Positional Defaults tab of the Preferences dialog box.



**Parts and Functions**

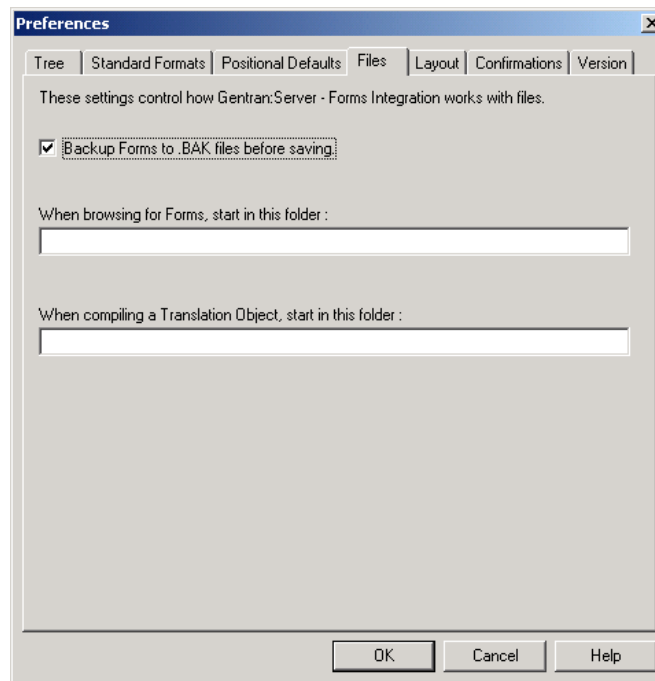
This table describes the parts and functions of the Positional Defaults tab on the Preferences dialog box.

Part	Function
Field Formatting Data Type	Describes the default data type. Valid options are: <ul style="list-style-type: none"> <li>▶ String</li> <li>▶ Number</li> <li>▶ Dt/Tm</li> </ul>
Pad character	Specifies whether you want characters padded by default.
Alignment	Specifies field formatting alignment. Valid options are: <ul style="list-style-type: none"> <li>▶ Left</li> <li>▶ Centre</li> <li>▶ Right</li> </ul>
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Help</b>	Launches the Forms Integration online Help system.



## Preferences dialog box (Files tab)

This illustration describes the Files tab of the Preferences dialog box.



## Parts and Functions

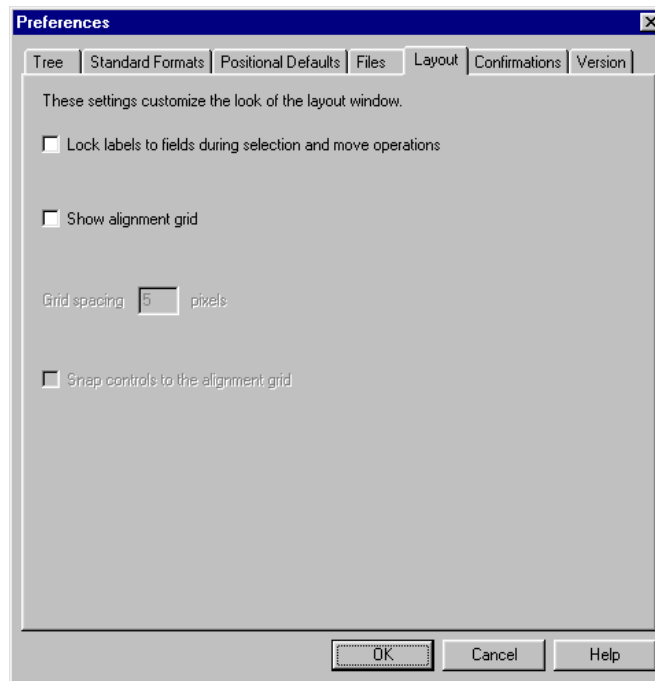
This table describes the parts and functions of the Files tab on the Preferences dialog box.

Part	Function
Backup forms to .BAK files before saving	A selected check box indicates that you want to create backup Forms Integration files.
When browsing to Forms, start at this folder	Specifies the path and folder where you want the system to begin browsing for the Forms Integration subsystem. The default location is C:\GENSRVNT\Forms.
When compiling a translation object, start at this folder	Specifies the path and folder where you want the system to start the translation object compilation process.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancel changes; exits the dialog box.
<b>Help</b>	Launches the Forms Integration online Help system.



**Preferences dialog box (Layout tab)**

This illustration describes the Layout tab of the Preferences dialog box.



**Parts and Functions**

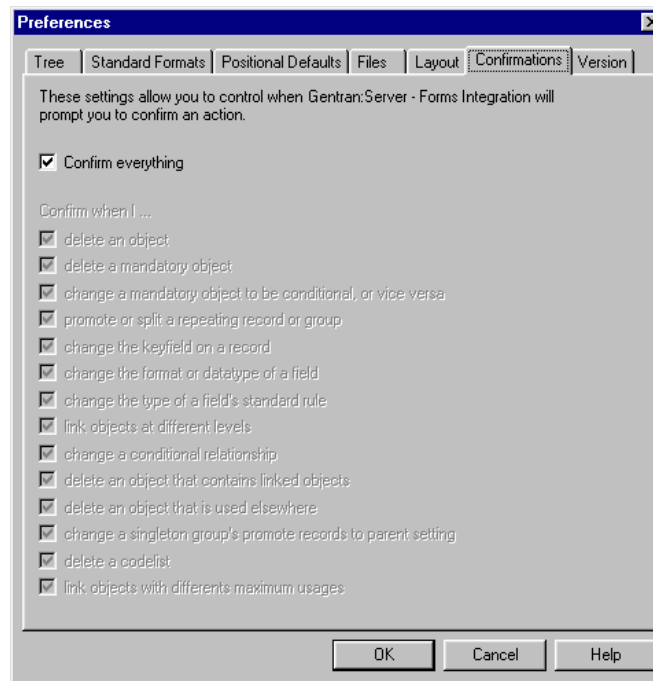
This table describes the parts and functions of the Layout tab on the Preferences dialog box.

Part	Function
Lock labels to field during selection and move operations check box	Groups the field label with the field in the Layout Window.
Show alignment grid	Displays Layout Window grid lines for alignment purposes.
Grid spacing in Pixels	Defines default grid line spacing in Pixels.
Snap controls to the alignment grid.	Aligns a field label and field on the nearest alignment grid setting.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Help</b>	Launches the Forms Integration online Help system.



**Preferences dialog  
box  
(Confirmations  
tab)**

This illustration describes the Confirmations tab of the Preferences dialog box.



**Parts and  
Functions**

This table describes the parts and functions of the Confirmations tab on the Preferences dialog box.

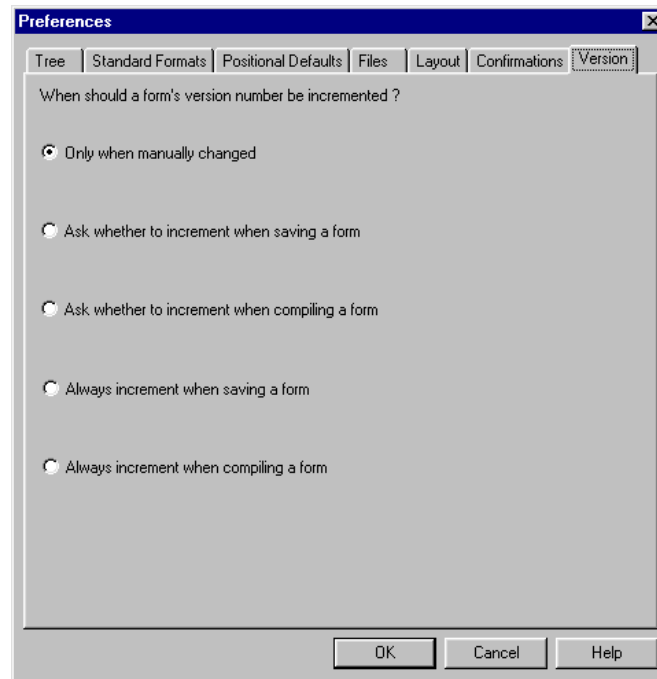
Part	Function
Confirm everything	Specifies that you want to receive a prompt to confirm all actions.
Confirm when I...	Specifies specific instances in which you want to receive a prompt to confirm an action. Valid options are: <ul style="list-style-type: none"> <li>▶ Delete an object</li> <li>▶ Delete a mandatory object</li> <li>▶ Change a mandatory object to be conditional or vice versa</li> <li>▶ Promote or split a repeating record or group</li> <li>▶ Change the keyfield on a record</li> <li>▶ Change the format or datatype of a field</li> <li>▶ Change the type of a field's standard rule</li> <li>▶ Link objects at different levels</li> <li>▶ Change a conditional relationship</li> <li>▶ Delete an object that contains linked objects</li> </ul> <p style="text-align: right;">(Continued on next page)</p>

<b>Part</b>	<b>Function</b>
Confirm when I... (contd)	<ul style="list-style-type: none"><li>▶ Delete an object that is used elsewhere</li><li>▶ Change a single group's promote record to parent setting</li><li>▶ Delete a codelist</li><li>▶ Link objects with different maximum usages.</li></ul>
<b>OK</b>	Saves changes; exits the dialog.
<b>Cancel</b>	Cancels changes; exits the dialog.
<b>Help</b>	Launches the Forms Integration online Help system.

---

## Preferences dialog box (Version tab)

This illustration describes the Version tab of the Preferences dialog box.



## Parts and Functions

This table describes the parts and functions of the Version tab on the Preferences dialog box.

Part	Function
When should a form's version number be incremented?	<p>Options are:</p> <ul style="list-style-type: none"> <li>▶ Only when manually changed - specifies that you want the system to increment the form's version number when you manually make a change.</li> <li>▶ Ask whether to increment when saving a form - specifies that, upon saving a form, you want the system to display a prompt asking whether you want the form's version number to increment.</li> <li>▶ Ask whether to increment when compiling a form - specifies that, upon compiling a form, you want the system to display a prompt asking whether you want the form's version number to increment.</li> <li>▶ Always increment when saving a form - specifies that, upon saving a form, you want the system to automatically increment the version number.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>Part</b>	<b>Function</b>
When should a form's version number be incremented? (contd)	▶ Always increment when compiling a form - specifies that, upon compiling a form, you want the system to automatically increment the version number.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Help</b>	Launches the Forms Integration online Help system.

---

## Customizing Display, Colours and Fonts

### Introduction

This section describes how to customize global (for all forms) display options, colours, and fonts.

### How to customize global display options

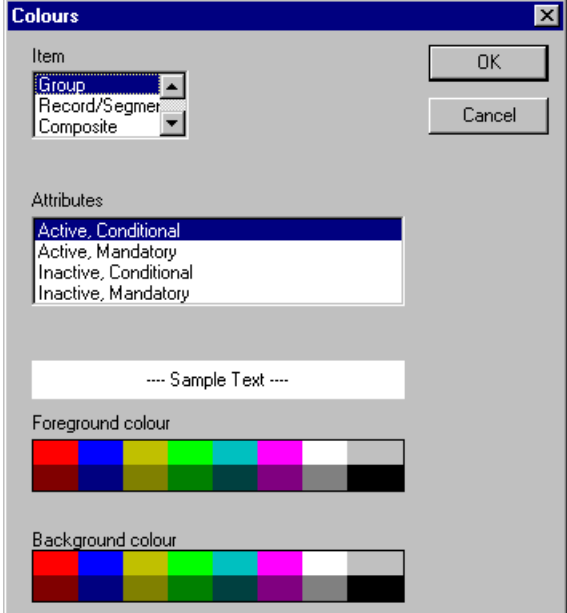
Use this procedure to customize global display options.

Step	Action
1	Select <b>Preferences</b> from the Options menu.  <b>System Response</b> The system displays the Tree tab of the Preferences dialog box.
2	By default, the options to show group record and field descriptions are selected. Do you want to clear any of these options? <ul style="list-style-type: none"> <li>▶ If yes, click the check box associated with the option that you do not want to display. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul> <b>Note</b> Typically, you want to have all the descriptions displayed for reference. However, depending on the size of your monitor, it may be easier to see the entire form if the descriptions are not displayed.  You may want to experiment with shrinking the size of the font for the form before you turn off the display of descriptions.  <b>Reference</b> See <i>How to customize global fonts</i> on page 2 - 18 for more information.
3	Click <b>OK</b> to save changes and exit the Preferences dialog box.

### How to customize global colours

Use this procedure to customize global colours.

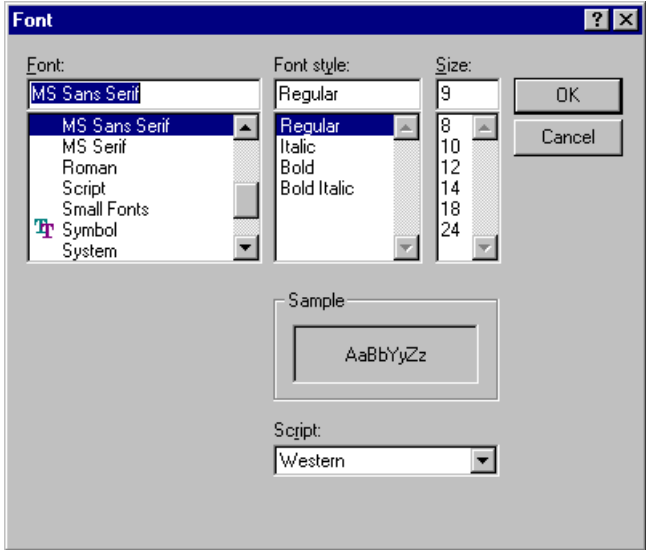
Step	Action
1	Select <b>Preferences</b> from the Options menu.  <b>System Response</b> The system displays the Tree tab of the Preferences dialog box.  (Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
2	<p>Click <b>Colours</b> to display the Colours dialog box.</p> 
3	<p>Select the form component type from the Item list (Group, Record/Segment, Composite, Field/Element).</p>
4	<p>Do you want to further define the form component type?</p> <ul style="list-style-type: none"> <li>▶ If yes, choose whether the form component is active or inactive and conditional or mandatory from the Attributes list. Valid selections are: <ul style="list-style-type: none"> <li>— Active, Conditional</li> <li>— Active, Mandatory</li> <li>— Inactive, Conditional</li> <li>— Inactive, Mandatory</li> </ul> </li> <li>▶ If no, proceed to the next step.</li> </ul>
5	<p>Repeat <b>Step 3</b> and <b>Step 4</b> for each combination of Items and Attributes for which you want to define colours.</p>
6	<p>Click <b>OK</b> to globally define the selected colours for all forms.</p>
7	<p>Click <b>OK</b> to save changes and to exit the Preferences dialog box.</p>



### How to customize global fonts

Use this procedure to customize global fonts.

Step	Action
1	Select <b>Preferences</b> from the Options menu.  <b>System Response</b> The system displays the Tree tab of the Preferences dialog box.
2	Click <b>Font</b> to display the Font dialog box.  
3	Select the type of font from the Font list. The default is MS Sans Serif.
4	Select the style from the Font Style list. The default is Regular.
5	Select the point size from the Size list. The default is 9.
6	You can preview the changes to the font in the Sample box. Click <b>OK</b> to make the global font change to all forms.
7	Click <b>OK</b> to save changes and exit the Preferences dialog box.



# Customizing Display of EDI File Format and Layout Windows

---

## Introduction

Gentran:Server enables you to choose how you want the EDI File Format and Layout Windows to display. These options, which are accessible from the View menu of the Main Menu bar, include:

- Maximize Tree - maximizes the EDI File Format Window.
- Maximize Layout - maximizes the Layout Window.
- Rearrange toggles the display of the EDI File Format Window and the Layout Window from left/right to above/below.

Additionally, Gentran:Server enables you to customize the measurement system that the vertical and horizontal rulers in the Layout Window use. The default measurement system is inches, but you can toggle the display to centimeters by double-clicking the yellow square where the vertical and horizontal rulers meet.

---

## Working With the Default Date Format

### Introduction

This section describes how to set, rearrange, add, or delete a date format.

### Setting a default date format

You typically establish the date format for all date elements *one time only*. This default date format is used when EDI documents are initially loaded from the standard. The format can be overridden on the validation tab of the Element Properties dialog box.

#### Note

This global option changes the default date format for all new forms. The format of the existing date elements does not change.

### Adding, rearranging, or deleting date formats

The Date/Time Format dialog box lists valid date formats. Using that dialog box, you can add, delete date formats, or rearrange the order in which date formats appear in the six- and eight-character date lists in the Standards Format tab of the Preferences dialog box.

### How to set the default Date/Time format

Use this procedure to establish the default Date/Time format.

Step	Action
1	Select <b>Preferences</b> from the Options menu.  <b>System Response</b> The system displays the Tree tab of the Preferences dialog box.
2	Click the <b>Standard Formats</b> tab.
3	Do you want to accept the default date formats for Six- and Eight-character dates? <ul style="list-style-type: none"> <li>▶ If yes, click <b>OK</b> to save your changes and to exit the Preferences dialog box.</li> <li>▶ If no, select the appropriate date formats from the Six-character dates and Eight-character dates lists. Click <b>OK</b> to save your changes and to exit the Preferences dialog box.</li> </ul>

### How to rearrange order of Date/Time formats

Use this procedure to rearrange the order of Date/Time formats that appear in the Six- and Eight-character date lists on the Standard Format tab of the Preferences dialog box.

Step	Action	
1	Select <b>Date Formats</b> from the Options menu.  <b>System Response</b> The system displays the Date/Time Formats dialog box.	
2	Select the Date/Time format that you want to move up or move down in the list order.	
3	Use this table to determine your next step.	
4	<b>IF you want to move the date format....</b>	<b>THEN...</b>
	Up	Press <b>Up</b> until the date format is in the desired list position. Proceed to the next step.
	Down	Press <b>Down</b> until the date format is in the desired list position. Proceed to the next step.
5	Click <b>OK</b> to save your changes and to exit the dialog box.	

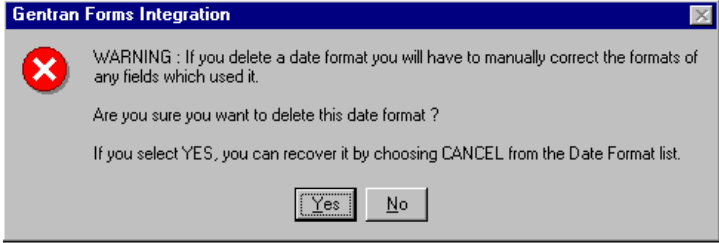
### How to add a new Date/Time format

Use this procedure to add a new Date/Time format to the Six- or Eight-character date lists on the Standard Format tab of the Preferences dialog box.

Step	Action
1	Select <b>Date Formats</b> from the <u>O</u> ptions menu.  <b>System Response</b> The system displays the Date/Time Formats dialog box.
2	Click <b>New</b> .
3	In the Name box, enter the name of the new date format that you want to create.
4	In the Picture box, enter how you want the date format to display.
5	Click <b>OK</b> to save your changes and to exit the Date/Time Formats dialog box.

### How to delete a date/time format

Use this procedure to delete a new Date/Time format from the Six- or Eight-character date lists on the Standard Format tab of the Preferences dialog box.

Step	Action
1	Select <b>Date Formats</b> from the Options menu.  <b>System Response</b> The system displays the Date/Time Formats dialog box.
2	Select the Date/Time format that you want to delete.
3	Press <b>Delete</b> .  <b>System Response</b> The system displays a warning dialog that specifies that you must manually correct formats for fields that use that date format that you want to delete.  
4	Verify that you want to delete the date/time format, and click <b>Yes</b> .  <b>System Response</b> You return to the Date/Time Format dialog box.  <b>Note</b> To recover the Date/Time format that you just deleted, click the Cancel button on the Date/Time Format dialog box.
5	Click <b>OK</b> to save your changes and to exit the Date/Time Formats dialog box.

# Creating a New Form

## Overview

---

### Introduction

You can create two different types of forms:

- Screen Entry - for keying documents in Document Editor
- Print - for printing documents

---

### Defining Form Components

After you create a new form, it is important to define and save the form details. Form components include the standard, version, transaction set (document) selected, groups, segments, composites, and elements that your company requires. The specific form components that you use depend on the type of form that you create.

After you select your form components, Gentran:Server generates an EDI file format for you.

### Recommendation

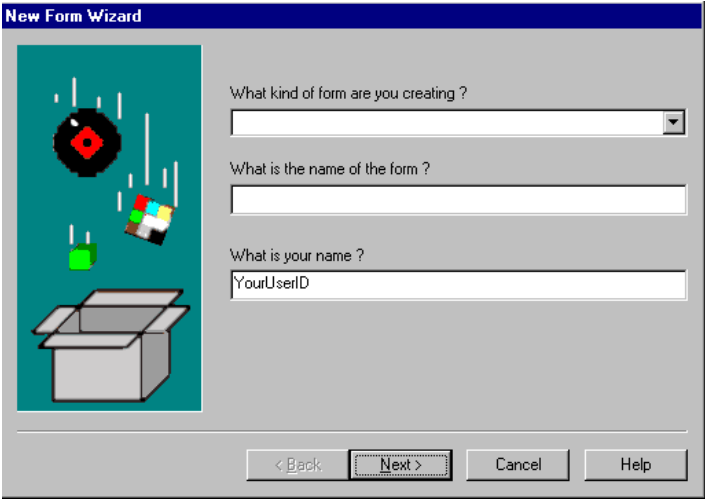
We recommend that you determine which form components you are using before generating or defining an EDI file format.

---

## Creating a Form and Defining Form Details

**Introduction** This section describes how to create a new form and define form details.

**How to create a new form** Use this procedure to create a new form.

Step	Action
1	<p>Select <b>New</b> from the File menu.</p> <p><b>System Response</b> The system displays the New Form Wizard dialog.</p> 
2	<p>From the What kind of Form are you creating list, select the type of form. Valid values are “Screen entry” or “Print.”</p>
3	<p>Enter the name of the form in the What is the name of the form box.</p> <p><b>Recommendation</b> We recommend that you use the following identifying characteristics: the partner that this form is used for, the standard, the version, the type of transaction this form uses, and the type of form.</p> <p><b>Example</b> For example, “MWT X 3030 850 Print” is the description of a print translation object used to print documents for partner MWT, for an ANSI X.12 version 003030 Purchase Order (850).</p>
4	<p>Verify that the system displays your User ID in the What is your name box.</p>
5	<p>Click <b>Next</b>.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>	
6	In the How would you like to define the data format section, use the following table to determine your next step.	
	<b>IF...</b>	<b>THEN...</b>
	If you want to create a new data format using Delimited, Positional, or CII syntax,	Proceed to <b>Step 7</b> .
	If you want to load the data format from a previously saved file definition,	Proceed to <b>Step 22</b> .
7	Click the <b>Create a new data format using this syntax</b> radio button.	
8	From the list, select a syntax option.	
9	Use this table to determine your next step.	
	<b>IF the syntax you selected is...</b>	<b>THEN...</b>
	Delimited EDI	Proceed to the next step.
	CII	Proceed to the next step.
	Positional	Click <b>Next</b> . Click <b>Finish</b> to begin editing your form.  You have completed this procedure.
10	Click <b>Customize</b> .  <b>System Response</b> The system displays the Wizard for the syntax that you selected.	
11	Click <b>Next</b> .	
12	Select the ODBC data source that you want to use from the list. The default is Gentran Standards.	
13	Click <b>Next</b> .	
14	Select the standards agency that you want to use from the list.	
15	Select the version of the standard from the Version list. The versions that are available depend on which standard you selected.  <span style="color: green;">(Continued on next page)</span>	



<b>(Contd) Step</b>	<b>Action</b>
16	<p>Select the transaction set (document or message) from the Transaction list. The transactions that are available depend on which standard and version you selected.</p> <p><b>Note</b> For TRADACOMS only, select the release number of the version that you are using from the Release list. The releases that are available depend on which standard, version, and transaction you selected.</p>
17	Click <b>Next</b> .
18	<p>Are you using CII syntax?</p> <ul style="list-style-type: none"> <li>▶ If yes, select a Multi-Detail Header option.</li> <li>▶ If no, proceed to the next step.</li> </ul>
19	<p>Click <b>Finish</b> to load the transaction set.</p> <p><b>System Response</b> You return to the New Form Wizard.</p>
20	Click <b>Next</b> .
21	<p>Click <b>Finish</b> to begin editing your form.</p> <p>You have completed this procedure.</p>
22	Click the Load the data format from a saved definition radio button.
23	<p>Type the path and file name that you want to load or click <b>Browse</b>.</p> <p><b>System Response</b> The system displays the Open File Definition dialog.</p>
24	The Look in list should reflect the drive\folder where Gentran:Server Forms Integration is installed. The default is GentranNT\Bin. If the current folder is not the correct one, select the appropriate folder.
25	Select the type of file that you want to display from the Files of type list. The default file extension is .IFD or .DDF.
26	<p>Select the file definition that you want to load into the system from the list. Double-click the filename or click Open to load the selected file format definition.</p> <p><b>System Response</b> You return to the New Form Wizard.</p>
27	Click <b>Next</b> .

(Continued on next page)



<b>(Contd) Step</b>	<b>Action</b>
28	<p>Click <b>Finish</b>.</p> <p><b>System response</b> The system finishes your form and displays the EDI file format in the Gentran:Server - Forms Integration Window.</p>

**How to define form details**

Use this procedure to define form details.

<b>Step</b>	<b>Action</b>
1	Open the form.
2	<p>Click <b>Details</b> from the Edit menu.</p> <p><b>System Response</b> The system displays the Translation Object Details dialog box.</p>
3	<p>Do you want to designate this form as a system translation object (one that is used internally by the system)?</p> <ul style="list-style-type: none"> <li>▶ If yes, under the Flags section, click the <b>System</b> check box. Proceed to the next step.</li> </ul> <p><b>WARNING</b> Be certain that you want to define this translation object as a system translation object. Once a translation object is registered with Gentran:Server, it cannot be deleted.</p> <ul style="list-style-type: none"> <li>▶ If no, proceed to the next step.</li> </ul>
4	<p>Do you want to designate a version number for this translation object?</p> <ul style="list-style-type: none"> <li>▶ If yes, under Version Control, enter the version number in the Major box and the release number in the Minor box. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>
5	<p>The Input and Output boxes (Agency, Version, Transaction, Release, and F Group) contain the EDI format. Do you want to change the information in these boxes?</p> <ul style="list-style-type: none"> <li>▶ If yes, make the desired changes, and proceed to the next step.</li> </ul> <p><b>Note</b> You can change the information in these boxes, but it will not alter the content of the form. You may wish to alter these boxes if, for example, you want the form to reflect a standards version that is not loaded on your system. You can change the version on this dialog box, and then alter the form to be compliant with that version.</p> <ul style="list-style-type: none"> <li>▶ If no, proceed to the next step.</li> </ul>
6	<p>Click <b>OK</b>.</p> <p><b>System Response</b> The system displays the Save As dialog box.</p> <p><b>Recommendation</b> We recommend that you store forms in the translation object folder where Gentran:Server is installed.</p>
7	<p>Type the name of the form (one to eight characters) in the File name box. The default file extension is .STP.</p>
8	<p>Click <b>Save</b> to save the form.</p>

# Activating Form Components

## Introduction

When Gentran:Server generates the form, the system includes all the groups, segments, composites, and elements that are defined by the standard agency for the version of the document you selected. The system activates all the groups, segments, composites, and elements that are defined as “mandatory” (must be present) by the standard. The system does not enable you to deactivate the mandatory groups, segments, composites, and elements.

## Activating components for translation

When translating data, the system does not process groups, segments, composites, and elements that are not activated. Therefore, *you* must activate the groups, segments, composites, and elements that are not defined as mandatory by the standard, but that you have determined that you need to use in the form.

## Procedure

Use this procedure to activate groups, segments, composites, and elements.

Step	Action
1	Click the <b>Activate</b> icon on the Main Toolbar.
2	Double-click to open each group that contains groups or segments that you want to activate.  <b>Note</b> As an alternative to opening each form component (Steps 2, 5, and 7), you can select Expand All from the View menu to open every form component. You can also select a form component and select Expand Branch from the View menu to open that form component.
3	Click each inactive group that you want to activate.  <b>Note</b> If you activate a group, segment, composite, or element by mistake, click the item with the right mouse button to access the shortcut menu. Select Deactivate from the shortcut menu.
4	Click each inactive segment that you want to activate.
5	Double-click to open each segment that contains composites or elements that you want to activate.
6	Click each inactive composite that you want to activate.
7	Double-click to open each composite that contains elements that you want to activate.  <i>(Continued on next page)</i>

<b>(Contd) Step</b>	<b>Action</b>
8	Click each inactive element that you want to activate.  <b>Note</b> When you activate an element for a print translation object, the word “Printed” displays before that element’s name. When you activate an element for a screen entry translation object, the words “Edit Box” or “Drop Down” display before that element’s name. “Edit Box” indicates that the element is formatted as a field. “Drop Down” indicates that the element is formatted as a list because you applied a standard rule that allows a selection of multiple items.
9	Verify that all groups, segments, and composites and elements that you need are activated.
10	Click the <b>Activate</b> icon on the Main Toolbar to turn activation mode off.

---

---

# Modifying Form Components

<b>Contents</b>	<ul style="list-style-type: none"> <li>▶ Introduction . . . . . 3 - 2</li> </ul>
	<b>Working with the EDI File Format . . . . . 3 - 3</b>
	<ul style="list-style-type: none"> <li>▶ Modifying an EDI File Format . . . . . 3 - 3</li> <li>▶ Promoting Groups and Repeating Segments . . . . . 3 - 4</li> <li>▶ Splitting Groups and Repeating Segments . . . . . 3 - 5</li> <li>▶ Using Cut, Copy, and Paste . . . . . 3 - 6</li> <li>▶ Verifying EDI Delimiters . . . . . 3 - 7</li> <li>▶ Saving a File Definition . . . . . 3 - 8</li> </ul>
	<b>Working with Groups . . . . . 3 - 9</b>
	<ul style="list-style-type: none"> <li>▶ Creating a Group . . . . . 3 - 9</li> <li>▶ Modifying Group Properties . . . . . 3 - 11</li> </ul>
	<b>Working with Segments . . . . . 3 - 14</b>
	<ul style="list-style-type: none"> <li>▶ Modifying Segment Properties . . . . . 3 - 14</li> <li>▶ Using the LS and LE Segments . . . . . 3 - 18</li> </ul>
	<b>Working with Composites . . . . . 3 - 23</b>
	<ul style="list-style-type: none"> <li>▶ Modifying Composite Properties . . . . . 3 - 23</li> </ul>
	<b>Working with Elements . . . . . 3 - 25</b>
	<ul style="list-style-type: none"> <li>▶ Overview . . . . . 3 - 25</li> <li>▶ Formatting Numbers . . . . . 3 - 26</li> <li>▶ Formatting Dates and Times . . . . . 3 - 29</li> <li>▶ Formatting Strings . . . . . 3 - 33</li> <li>▶ Creating a Syntax Token (Western European languages) . . . . . 3 - 34</li> <li>▶ Creating a Syntax Token (Far Eastern languages) . . . . . 3 - 38</li> <li>▶ Editing a Syntax Token . . . . . 3 - 40</li> <li>▶ Deleting a Syntax Token . . . . . 3 - 42</li> <li>▶ Deleting a character range from a syntax token . . . . . 3 - 43</li> <li>▶ Using Syntax Tokens . . . . . 3 - 44</li> <li>▶ Modifying Element Properties . . . . . 3 - 46</li> <li>▶ Storing EDI Code Value Descriptions . . . . . 3 - 50</li> <li>▶ Defining Relational Conditions . . . . . 3 - 52</li> </ul>

---

## Introduction

---

**In this chapter** This chapter describes how to modify form components.

---

# Working with the EDI File Format

## Modifying an EDI File Format

---

**Introduction**

You can modify the system-generated EDI file format by modifying the modifying form component properties and by using the promote, split, copy, cut, and paste functions.

**Reference**

See *Creating a Form and Defining Form Details* on page 2 - 24 for more information.

---

**Defining an EDI  
file format**

If you want to use a specialized version of an EDI standard that is not available in the Gentran:Server standards database, it may be appropriate for you to define the EDI file format yourself.

---

## Promoting Groups and Repeating Segments

---

**Introduction** The Promote function extracts one iteration (instance) of a group or repeating segment. It also sets the promote flag and places the active elements in the parent frame.

**Note**

The Promote function is only available if a group or repeating segment is selected.

---

**Promote vs. Copy and Paste** You can use the Copy and Paste functions (and change the number in the maximum usage box) to accomplish the same task. However, Promote is a specialized function that guarantees the integrity of your EDI structure.

---

**Related topics** See *Splitting Groups and Repeating Segments* on page 3 - 5 for more information and *Using Cut, Copy, and Paste* on page 3 - 6 for more information.

---

**Procedure** Use this procedure to promote a group or repeating segment.

Step	Action
1	Highlight the group or repeating segment from which you want to extract one iteration.
2	Click the <b>Promote</b> icon the Main Toolbar. This extracts one iteration (instance) of the looping structure.

---



# Splitting Groups and Repeating Segments

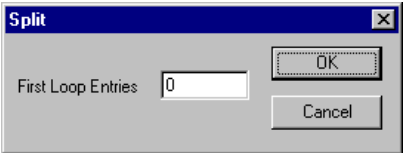
**Introduction** The Split function enables you to break a group or repeating segment into two loops.

**When to use** You typically use this function when you need more than one instance of the same form component that still occurs multiple times. You can also use the Split function to split one instance of a group without setting the promote flag (the Split function never sets the promote flag).

**Note**  
The Split function is only available if a group or repeating segment is selected.

**Split vs. Copy and Paste** You can use the Copy and Paste functions (and change the number in the maximum usage box) to accomplish the same task. However, Split is a specialized function that guarantees the integrity of your EDI structure.

**Procedure** Use this procedure to split a group or repeating segment.

Step	Action
1	Highlight the group or repeating segment from which you want to extract one iteration.
2	Click the <b>Split</b> icon on the Main Toolbar. <b>System Response</b> The system displays the Split dialog box. 
3	In the First Loop Entries box, type the number that indicates the sequential number of iterations where you want the group or repeating segment split. <b>Note</b> The number you enter in the First Loop Entries box must be greater than zero and less than the maximum number of iterations of the loop. <b>Example</b> For example, if the X loop repeats a maximum number of 5 times, and you type “2” in the First Loop Entries box, the resulting split generates one X loop that repeats a maximum of 2 times and a second X loop that repeats a maximum of 3 times.
4	Click <b>OK</b> to exit the Split dialog box.



## Using Cut, Copy, and Paste

**Introduction** The Copy, Cut, and Paste functions are typically used to move EDI information in the form. You can cut or copy a single form component (loop, segment, composite, element, record, or field) and paste it in another location in the form. Copied form components retain all the information of the original form component. If the copied form component contains subordinate form components (like a segment contains subordinate elements), the subordinate form components are also copied. You can also cut, copy, and paste a form component from one form to another.

**When to use** You may need to use Cut, Copy, or Paste if you are using an EDI standard differently, or if you need to create nested looping structures.

**Procedure** Use this procedure to cut, copy, and paste a form component.

Step	Action
1	Select the form component or text that you want to cut or copy.
2	Click the <b>Cut</b> or <b>Copy</b> icon on the Main Toolbar.
3	Are you are pasting the form component or text into another form? <ul style="list-style-type: none"> <li>▶ If yes, open that form, if it is not already open.</li> <li>▶ If no, proceed to the next step.</li> </ul>
4	Position the cursor where you want the cut or copied selection pasted <i>after</i> .
5	Click <b>Paste</b> on the Main Toolbar to paste the contents of the Clipboard.
6	Are you pasting a form component? <ul style="list-style-type: none"> <li>▶ If yes, proceed to the next step.</li> <li>▶ If no, you have completed this procedure.</li> </ul>
7	Is the form component part of a group? <ul style="list-style-type: none"> <li>▶ If yes, Gentran:Server prompts you to specify whether you want the contents of the Clipboard pasted as a child (subordinate) of the group or pasted at the same level as the group. Select the appropriate option, and click <b>OK</b>.</li> <li>▶ If no, you have completed this procedure.</li> </ul>

## Verifying EDI Delimiters

**Introduction** If you are using an EDI standard that contains composite elements or sub-elements, you must verify that Gentran:Server is specifying the correct delimiters.

**What is a Delimiter?** Delimiters are flags that you define to the system as separating specific EDI components. Delimiters are necessary for all variable field-length standards, because the data is compressed (and the leading zeroes and trailing blanks are removed). Since the fields vary in length, the system needs a flag to determine where one element ends and another begins. For example, an element delimiter marks the beginning of a new element.

### Recommendation

Although verifying EDI delimiters in Gentran:Server is mandatory *only* if you are using a standard with composite elements or sub-elements, we recommend that perform this task regardless of which standard you use.


**Procedure** Use this procedure to verify EDI delimiters.

Step	Action
1	Click the <b>EDI file</b> icon with the right mouse button.  <b>System Response</b> The system displays the shortcut menu.
2	Select <b>Properties</b> from the shortcut menu.  <b>System Response</b> The system displays the File Format Properties dialog box.
3	Click the <b>Delimiters</b> tab.  <b>System Response</b> The system displays the Delimiters tab of File Format Properties dialog box.
4	Do the required delimiters for the EDI standard you are using differ from the default Delimiter tab settings?  <ul style="list-style-type: none"> <li>▶ If yes, type either the character or the hexadecimal value in the correct box. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul>
5	Click <b>OK</b> to exit the File Format Properties dialog box.

## Saving a File Definition

**Introduction** Gentran:Server enables you to save an individual file format definition so that you can use it as a guide in future forms. This provides you with a quick way to build a form.

**Procedure** Use this procedure to save an individual file format definition.

Step	Action
1	Open the form.
2	Right-click the <b>EDI file</b> icon.  <b>System Response</b> The system displays the shortcut menu.
3	Select <b>Save File Definition As</b> from the shortcut menu.  <b>System Response</b> The system displays the Save File Definition dialog box.   <p>The Save in list should reflect the drive\folder where Gentran:Server is installed. The default is GENSRVNT.</p>
4	If the current folder is not the correct one, select the appropriate folder.
5	Select the file definition from the list or type the name of the file definition in the File name box. The default file extension is .IFD or .DDF.
6	Click <b>Save</b> to save the file definition.

# Working with Groups

## Creating a Group

**Introduction** A group contains related segments and/or groups that repeat in sequence until either the data ends or the maximum number of times that the loop is allowed to repeat is exhausted. If you create a group that is subordinate to another group, this corresponds to a nested looping structure (a loop within a loop).

**Procedure** Use this procedure to create a group.

Step	Action	
1	Use this table to determine your next step.	
	<b>IF you want to...</b>	<b>THEN...</b>
	Create a group that is at the same level as a selected form component,	<ul style="list-style-type: none"> <li>▶ highlight the form component and select <b>Insert</b> from the Edit menu. Select <b>Group</b> from the cascading menu.</li> </ul> <p><b>System Response</b> The system displays the Group Properties dialog box with the Name tab showing.</p> <ul style="list-style-type: none"> <li>▶ Proceed to the next step.</li> </ul>
Create a group that is subordinate to a selected form component,	<ul style="list-style-type: none"> <li>▶ highlight the form component and select <b>Create Sub</b> from the Edit menu. Select <b>Group</b> from the cascading menu.</li> </ul> <p><b>System Response</b> The system displays the Group Properties dialog box with the Name tab showing.</p> <ul style="list-style-type: none"> <li>▶ Proceed to the next step.</li> </ul>	
2	In the first box on the Name tab, type the group name.  <b>Note</b>  Do not use spaces or dashes (-) in the Name box. You can use the underscore (_) to separate words.	
3	In the second box on the Name tab, type the loop description. This box is used to provide a brief explanation of the loop.  (Continued on next page)	



<b>(Contd) Step</b>	<b>Action</b>	
4	Use this table to determine your next step.	
	<b>IF this form is a...</b>	<b>THEN...</b>
	Screen entry form,	click the <b>Display</b> tab to access display options. Proceed to the next step.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about display options.
Print form,	click the <b>Print</b> tab to access print options. Proceed to the next step.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about print options.	
5	Click the <b>Looping</b> tab to access looping options.	
6	In the Min Usage box, type the minimum number of times the loop must be repeated.  <b>Note</b> Mop, the minimum usage should be “1” or greater.	
7	In the Max Usage box, type the maximum number of times the loop can be repeated.	
8	Click the <b>Loop Extended Rules</b> tab only if you want to specify an extended rule for this group.  <b>Reference</b> See the Using Extended Rules, chapter 6 in this guide, for more information.	
9	Click <b>OK</b> to create the group.	

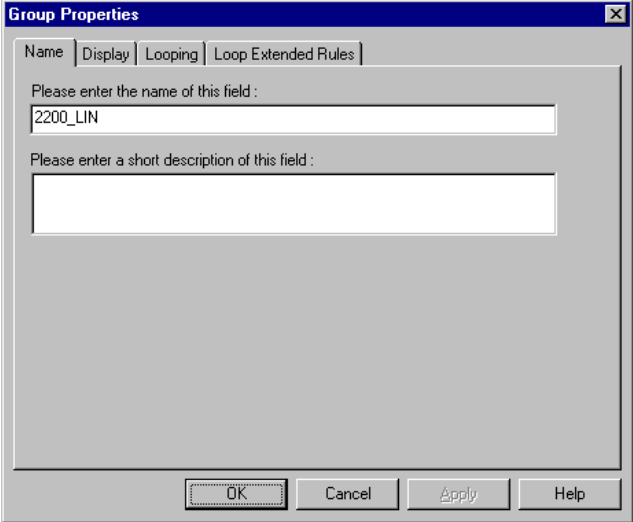
**Related Topic**

See the *Modifying Group Properties* on page 3 - 11 section in the this chapter for information on modifying group properties.

# Modifying Group Properties

**Introduction** A group contains related segments and/or groups that repeat in sequence until either the group data ends or the maximum number of times that the loop is allowed to repeat is exhausted. If you create a group that is subordinate to another group, this corresponds to a nested looping structure (a loop within a loop).

**Procedure** Use this procedure to modify the properties of a group.

Step	Action
1	<p>Right-click the group you want to modify, and select <b>Properties</b> from the shortcut menu.</p> <p><b>System Response</b> The system displays the Group Properties dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>		
2	Use this table to determine your next step.		
	<b>IF you want to modify the...</b>	<b>THEN...</b>	
	group name,	type the group name in the first box on the Name tab. Proceed to the next step.  <b>Note</b> Do not use spaces or dashes (-) in the Name box. You can use the underscore "0" to separate words.  <b>Note</b> If a group occurs more than once in a form it is identified by its name <ID>. The second and subsequent occurrences are identified by <ID>:n, where "n" is the number of the occurrence in the form.	
	group description,	type the new description in the second box on the Name tab. This box is used to provide a brief explanation of the loop. Proceed to the next step.	
3	Use this table to determine your next step.		
	<b>IF this is a...</b>	<b>AND you want to...</b>	<b>THEN...</b>
	Screen entry form,	change display options,	click the <b>Display</b> tab.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about display options.
Print form,	print options,	click the <b>Print</b> tab.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about print options.	
4	Click the <b>Looping</b> tab to access looping options.  (Continued on next page)		



<b>(Contd) Step</b>	<b>Action</b>
5	<p>In the Min Usage box, type the minimum number of times the loop must be repeated.</p> <p><b>Note</b> For a conditional loop, the minimum usage should always be “0” (zero). For a mandatory loop, the minimum usage should be “1” or greater.</p>
6	<p>In the Max Usage box, type the maximum number of times the loop can be repeated.</p>
7	<p>Do you want to specify an extended rule for this group?</p> <ul style="list-style-type: none"><li>▶ If yes, click the <b>Loop Extended Rules</b> tab. Define the minimum and maximum usage. Proceed to the next step.</li><li>▶ If no, proceed to the next step.</li></ul> <p><b>Reference</b> See Using Extended Rules, chapter 6 in this guide, for more information.</p>
8	<p>Click <b>OK</b> to save the changes to the group.</p>

---

## Working with Segments

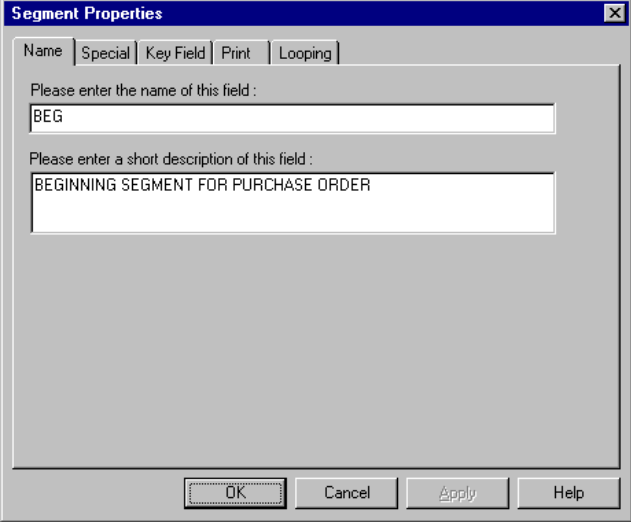
### Modifying Segment Properties

**Introduction** You can modify the properties of any segment, including the minimum and maximum number of times the segment can repeat, whether the segment is mandatory or conditional, and whether it is a loop start or loop end segment.

**Reference**

See *Using the LS and LE Segments* on page 3 - 18 for more information.

**Procedure** Use this procedure to modify the properties of a segment.

Step	Action
1	<p>Right-click the segment you want to modify, and select <b>Properties</b> from the shortcut menu.</p> <p><b>System Response</b> The system displays the Segment Properties dialog box.</p>  <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
2	<p>Do you want to modify the name of the segment?</p> <ul style="list-style-type: none"> <li>▶ If yes, type the segment name in the first box on the Name tab. Proceed to the next step.</li> </ul> <p><b>Note</b> If a segment occurs more than once in a form it is identified by its name &lt;ID&gt;. The second and subsequent occurrences are identified by &lt;ID&gt;:n, where “n” is the number of the occurrence in the form.</p> <ul style="list-style-type: none"> <li>▶ If no, proceed to the next step.</li> </ul>
3	<p>Do you want to modify the description of the segment?</p> <ul style="list-style-type: none"> <li>▶ If yes, type a new description in the second box on the Name tab. Proceed to the next step.</li> </ul> <p><b>Note</b> This box is used to provide a brief explanation of the segment that allows you to differentiate it from similar segments.</p> <ul style="list-style-type: none"> <li>▶ If no, proceed to the next step.</li> </ul>
4	<p>Do you want to flag this segment as containing the binary data?</p> <ul style="list-style-type: none"> <li>▶ If yes, click the <b>Special</b> tab. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul> <p><b>Note</b> If you select “Binary,” you must define an element of data-type “Bin Length” and another element of data-type “Bin Data.” The “Bin Length” element must precede the “Bin Data” element.</p> <p><b>Reference</b> See <i>Modifying Element Properties</i> on page 3 - 46 for more information about defining an element data-type.</p>
5	<p>Do you want to use a key field?</p> <p><b>Note</b> The key field function enables you to specify a second qualification in selecting a segment (the segment name is the first qualification). Data must be provided in the order designated through the use of key fields.</p> <ul style="list-style-type: none"> <li>▶ If yes, click the <b>Key Field</b> tab. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>		
6	Use this table to determine your next step.		
	<b>IF this is a...</b>	<b>AND you want to...</b>	<b>THEN...</b>
	screen entry form,	you want to change display options,	click the <b>Display</b> tab.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about display options.
print form,	change print options,	click the <b>Print</b> tab.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about print options.	
7	<p>Is this segment is a looping structure?</p> <ul style="list-style-type: none"> <li>▶ If yes, click the <b>Looping</b> tab. Proceed to the next step.</li> <li>▶ If no, accept the defaults. Proceed to the next step.</li> </ul>		
8	<p>In the Min Usage box, type the minimum amount of times the segment must repeat.</p> <p><b>Note</b> If the Min Usage box contains a “0” (zero), the segment is “conditional.” If the Min Usage box contains a “1” or greater, the segment is “mandatory.”</p>		
9	<p>Do you want the segment to repeat (loop)?</p> <ul style="list-style-type: none"> <li>▶ If yes, type the maximum amount of times it can repeat in the Max Usage box. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>		

<b>(Contd) Step</b>	<b>Action</b>
10	<p>Do you want to indicate that the segment is the beginning of a loop (Loop Start) or the end of a loop (Loop End)?</p> <ul style="list-style-type: none"> <li>▶ If yes, select the appropriate options.</li> <li>▶ If no, accept the default of Normal for the Normal/Loop Start/Loop End options. These options are defined as follows:                             <ul style="list-style-type: none"> <li>— Normal - This segment is in the loop but is not the beginning or ending segment.</li> <li>— Loop Start - This segment marks the beginning of the loop.</li> <li>— Loop End - This segment marks the end of the loop.</li> </ul> </li> </ul>
11	<p>Do you want to specify an extended rule for this group?</p> <ul style="list-style-type: none"> <li>▶ If yes, click the <b>Loop Extended Rules</b> tab.</li> <li>▶ If no, proceed to the next step.</li> </ul> <p><b>Reference</b> See <i>Defining a Form Component Extended Rule</i> on page 6 - 11 for more information.</p>
12	Click <b>OK</b> to save the changes to the segment.



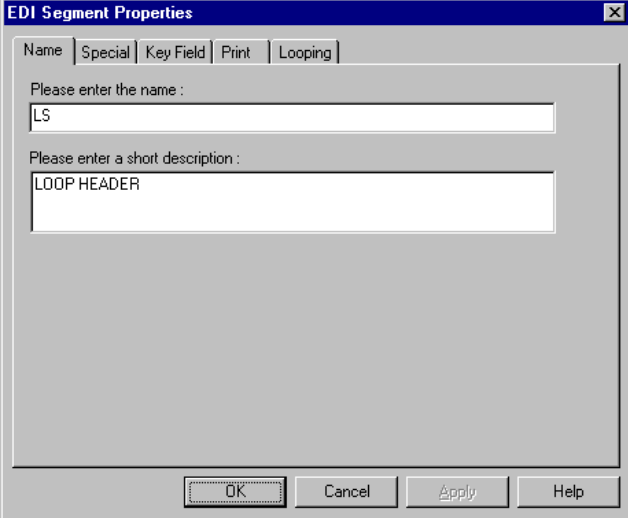
## Using the LS and LE Segments

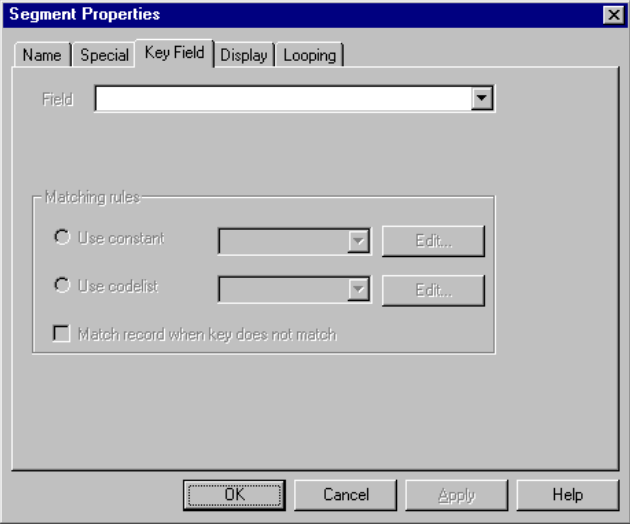
### Introduction

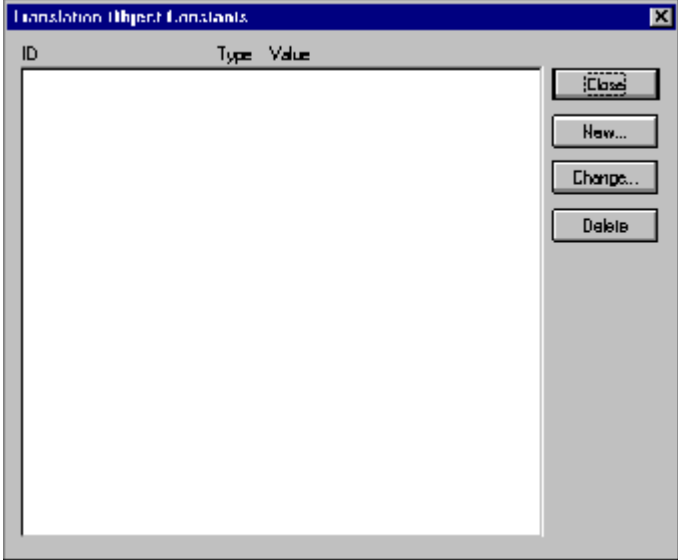
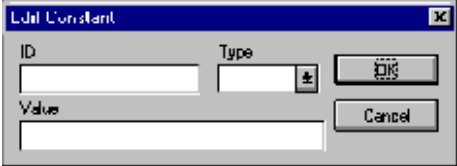
Certain EDI standards use Loop Start (LS) and Loop End (LE) segments. LS and LE segments specify the explicit start and end of loops.

### How to define a Loop Start

Use this procedure to define an LS segment for a form.

Step	Action
1	<p>Right-click the <b>LS segment</b> to access the shortcut menu. Select <b>Properties</b> from the shortcut menu.</p> <p><b>System Response</b> The system displays the Segment Properties dialog box (Name tab).</p> 
2	Click the <b>Looping tab</b> to access the loop options.
3	<p>Verify that the <b>Loop Start</b> option on the Looping tab is selected.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

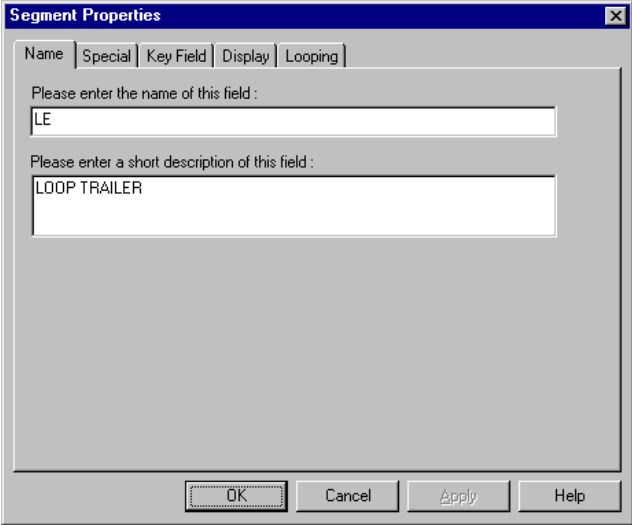
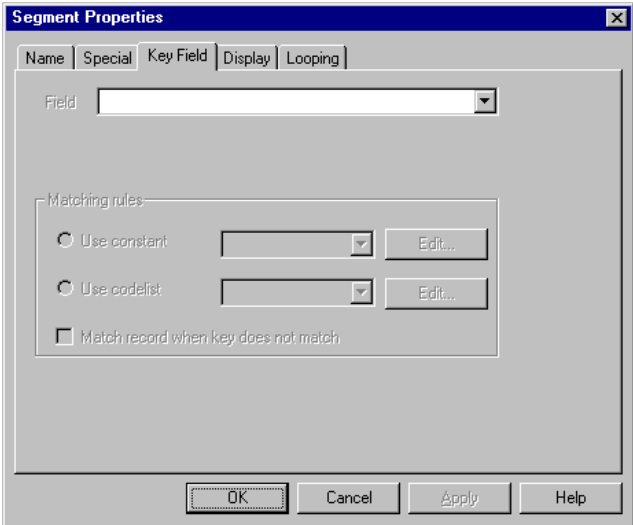
<b>(Contd) Step</b>	<b>Action</b>
4	<p>Click the <b>Key Field</b> tab to access the key field options.</p> 
5	<p>From the Field list, select the <b>Loop Identifier Code</b>.</p> <p><b>Note</b> This list contains all the elements that are contained in the segment. This list gives you the ability to define the Loop Start segment definition, by specifying that the loop identifier code must have the value you specify in the Matching rules section.</p>
6	<p>The Matching Rules section enables you to access all the literal constants and code lists currently defined for this form. Click the <b>Use constant</b> option.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

(Contd) Step	Action
7	<p>Click <b>Edit</b> (at the right of the Use constant list) to access the Map Constants dialog box.</p> 
8	<p>Click <b>New</b> to access the Edit Constant dialog box.</p> 
9	<p>In the ID box, type the literal constant identifier.</p>
10	<p>From the Type list, select <b>String</b>. This is the category of this literal constant.</p>
11	<p>In the Value box, type the value of the literal constant.</p> <p>For a screen entry form, this is the loop identifier code that you expect to receive from your trading partner.</p>
12	<p>Click <b>OK</b> to add the constant to the system.</p>
13	<p>Select the constant you created from the list. The system matches the selected constant against the loop identifier code.</p>
14	<p>Click <b>OK</b> to exit the Segment Properties dialog box.</p>



**How to define a Loop End**

Use this procedure to define an LE segment for a form.

Step	Action
1	<p>Right-click the <b>LE segment</b> to access the shortcut menu. Select <b>Properties</b> from the shortcut menu.</p> <p><b>System Response</b> The system displays the Segment Properties dialog box (Name tab).</p> 
2	Click the <b>Looping</b> tab to access the loop options.
3	Verify that the <b>Loop End</b> option on the Looping tab is selected.
4	<p>Click the <b>Key Field</b> tab to access the key field options.</p> 

(Continued on next page)



<b>(Contd) Step</b>	<b>Action</b>
5	<p>From the Field list, select the <b>Loop Identifier Code</b>. This list contains all the elements that are contained in the segment.</p> <p>This list gives you the ability to define the Loop Start segment definition, by specifying that the loop identifier code must have the value you specify in the Matching rules section.</p>
6	<p>The Matching Rules section enables you to access all the literal constants and code lists currently defined for this form. Select the constant you created for the LS segment from the Use Constant list. The system matches the selected constant against the loop identifier code.</p>
7	<p>Click <b>OK</b> to exit the Segment Properties dialog box.</p>

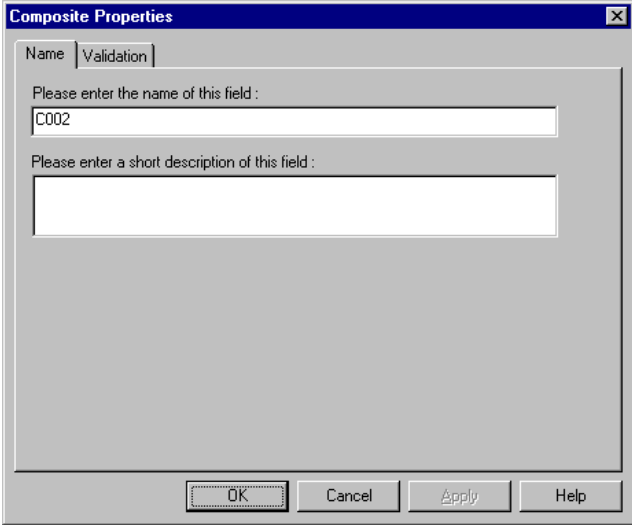
---

# Working with Composites

## Modifying Composite Properties

**Introduction** A composite is a data element that contains two or more component data elements or subelements. You can modify the composite name and description, and whether or not it is mandatory.

**Procedure** Use this procedure to modify composite properties.

Step	Action
1	<p>Right-click the composite you want to modify, and select <b>Properties</b> from the shortcut menu.</p> <p><b>System Response</b> The system displays the Composite Properties dialog box</p> 
2	<p>Do you want to change the composite name?</p> <ul style="list-style-type: none"> <li>▶ If yes, type the data element identification number in the first box on the Name tab. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul> <p><b>Note</b> If a composite occurs more than once in a form it is identified by its name &lt;ID&gt;. The second and subsequent occurrences are identified by &lt;ID&gt;:n, where “n” is the number of the occurrence in the form.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>
3	Do you want to change the composite description? <ul style="list-style-type: none"><li>• If yes, type the new description in the second box on the Name tab. Proceed to the next step.</li><li>• If no, proceed to the next step.</li></ul>
4	Do you want to specify the composite as mandatory? <ul style="list-style-type: none"><li>• If yes, click the Validation tab. Select the check box on the Validation tab. Proceed to the next step.</li><li>• If no, proceed to the next step.</li></ul>
5	Click <b>OK</b> to save the changes to the composite.

---

# Working with Elements

## Overview

---

**Introduction**

When you define or modify an element, you must specify the type and format. Available element format options depend on which element type you select (string, number, or Date/Time).

---

# Formatting Numbers

## Introduction

A number-type element contains either an integer or a real number. If a field is a number type, you must specify the format (either “R” or “N”) and the number of decimal places.

## “R” type numbers

Number formats beginning with “R” represent numbers with an explicit decimal point. Trailing zeros are truncated. For example, if the format you select is R2, then numbers are formatted with two places to the right of the decimal point (e.g., “2.01”).

## “N” type numbers

Number formats beginning with “N” represent numbers with an implicit decimal point. The decimal point is not displayed, but the system still processes the number correctly. For example, if the format you select is N2, then numbers are formatted with two places to the right of an implicit decimal point.

## Choosing a number-type format

The number type format that you choose depends on your requirements. Regardless of whether you use the “N” or “R” format, you must also indicate the number of decimal places in the field.

You can format numbers in various numeric formats.

## Valid number formats

This table describes valid number formats.

Number Format	Description
R0	Number formatted with an explicit decimal point and no decimal places
R1	Number formatted with an explicit decimal point up to 1 decimal place
R2	Number formatted with an explicit decimal point up to 2 decimal places
R3	Number formatted with an explicit decimal point up to 3 decimal places
R4	Number formatted with an explicit decimal point up to 4 decimal places
R5	Number formatted with an explicit decimal point up to 5 decimal places
R6	Number formatted with an explicit decimal point up to 6 decimal places

(Continued on next page)

<b>Number Format</b>	<b>Description</b>
R7	Number formatted with an explicit decimal point up to 7 decimal places
R8	Number formatted with an explicit decimal point up to 8 decimal places
R9	Number formatted with an explicit decimal point up to 9 decimal places
N0	Number formatted with an implicit decimal point and no decimal places
N1	Number formatted with an implicit decimal point up to 1 decimal place
N2	Number formatted with an implicit decimal point up to 2 decimal places
N3	Number formatted with an implicit decimal point up to 3 decimal places
N4	Number formatted with an implicit decimal point up to 4 decimal places
N5	Number formatted with an implicit decimal point up to 5 decimal places
N6	Number formatted with an implicit decimal point up to 6 decimal places
N7	Number formatted with an implicit decimal point up to 7 decimal places
N8	Number formatted with an implicit decimal point up to 8 decimal places
N9	Number formatted with an implicit decimal point up to 9 decimal places

**Procedure** Use this procedure to format numbers in a screen entry or print form.

Step	Action
1	Double-click an existing element.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).
2	Click the <b>Validation</b> tab to access validation options.
3	From the data-type list, select <b>Number</b> . This indicates that the element is a numeric or real number that can be mathematically manipulated.
4	From the form format list, select the format for the number.  <b>Note</b> Usually, the number formats in the form format list and in the file format list are the same. However, if there is an “N” format selected in the form format list, you might want to select an “R” format in the file format list so that the decimal point in the numbers is displayed.
5	Verify that the number format selected in the file format list is the correct format that is written to the EDI file.  <b>Note</b> A number format must be designated for number-type elements in the file format list, even for hidden fields.
6	Click <b>OK</b> to exit the Element Properties dialog box.



## Formatting Dates and Times

### Introduction

A date/time-type element contains either a date or a time. If an element is a date/time type, you must designate how the date or time is formatted in the screen entry form. You can format dates and times that are written to an EDI file in various formats.

### Valid Date/Time Formats

This table lists Valid Date/Time formats.

Format	Description
YYMMDD	Two-digit year, two-digit month, two-digit day
MMDDYY	Two-digit month, two-digit day, last two digits of year (example: 121599)
YYYYMMDD	Four-digit year, two-digit month, two-digit day (example: 19991215)
DDMMYYYY	Two-digit day, two-digit month, four-digit year (example: 15121999)
MMDDYYYY	Two-digit month, two-digit day, four-digit year (example: 12151999)
DDMMYY	Two-digit day, two-digit month, last two digits of year (example: 151299)
YYMMMDD	Last two digits of year, three-letter abbreviation of the month, two-digit day (example: 99JAN02)
DDMMYY	Two-digit day, three-letter abbreviation of the month, last two digits of year (example: 02JAN99)
MMMDDYY	Three-letter abbreviation of the month, two-digit day, last two digits of year (example: JAN0299)
YYYYMMMDD	Four-digit year, three-letter abbreviation of the month, two-digit day (example: 2003JUL04)
DDMMYYYY	Two-digit day, three-letter abbreviation of the month, four-digit year (example: 04JUL2003)
MMMDDYYYY	Three-letter abbreviation of the month, two-digit day, four-digit year (example: JUL042003)
YYDDD	Last two digits of year, three-digit Julian day (example: 99349 for the 349th day of 1999)
DDDY	Three-digit Julian day, last two digits of year (example: 34999)

(Continued on next page)

<b>(Contd) Format</b>	<b>Description</b>
YYYYDDD	Four-digit year, three-digit Julian day (example: 1999349)
DDDDYYYY	Three-digit Julian day, four-digit year (example: 3491999)
YY/MM/DD	Last two digits of year, separator, two-digit month, separator, two-digit day (example: 99/12/05)
DD/MM/YY	Two-digit day, separator, two-digit month, separator, last two digits of year (example: 05/12/99)
MM/DD/YY	Two-digit month, separator, two-digit day, separator, last two digits of year (example: 12/15/99)
YYYY/MM/DD	Four-digit year, separator, two-digit month, separator, two-digit day (example: 1999/12/15)
DD/MM/YYYY	Two-digit day, separator, two-digit month, separator, four-digit year (example: 15/12/1999)
MM/DD/YYYY	Two-digit month, separator, two-digit day, separator, four-digit year (example: 12/15/1999)
YY/MMM/DD	Two-digit year, separator, three-letter abbreviation of the month, separator, two-digit day (example: 99/JUL/20)
DD/MMM/YY	Two-digit day, separator, three-letter abbreviation of the month, separator, two-digit year (example: 20/JUL/99)
MMM/DD/YY	Three-letter abbreviation of the month, separator, two-digit day, separator, two-digit year (example: JUL/20/99)
YYYY/MMM/DD	Four-digit year, separator, three-letter abbreviation of the month, separator, two-digit day (example: 2003/JUL/25)
DD/MMM/YYYY	Two-digit day, separator, three-letter abbreviation of the month, separator, four-digit year (example: 25/JUL/2003)
MMM/DD/YYYY	Three-letter abbreviation of the month, separator, two-digit day, separator, four-digit year (example: JUL/25/2003)
YY/DDD	Last two digits of year, separator, three-digit Julian day (example: 99/349)
DDD/YY	Three-digit Julian day, separator, last two digits of year (example: 349/99)
YYYY/DDD	Four-digit year, separator, three-digit Julian day (example: 1999/349)

(Continued on next page)

<b>(Contd) Format</b>	<b>Description</b>
DDD/YYYY	Three-digit Julian day, separator, four-digit year (example: 349/1999)
MONTH	Month (example: December)
DAY	Day of the week (example: Friday)
HHMM	Two-digit hour, two-digit minutes (example: 0330 for 30 minutes past 3 o'clock)
HHMMSS	Two-digit hour, two-digit minutes, two-digit seconds (example: 033045 for 30 minutes and 45 seconds past 3 o'clock)
HH:MM	Two-digit hour, separator, two-digit minutes (example: 03:30)
HH:MM:SS	Two-digit hour, separator, two-digit minutes, separator, two-digit seconds (example: 03:30:45)
ISO-8601	YYYYMMDDTHHMMSS.mmmZ format: Four-digit year, two-digit month, two-digit day, T (time) indicator, two-digit hour, two-digit minutes, two-digit seconds in Universal Time (also called Zulu Time or Greenwich Mean Time), Z (Zulu time) indicator (example: 20031209T123000.000Z)
YYYYMMDDZ	ISO-8601 date format: Four-digit year, two-digit month, two-digit day, Z (Zulu time) indicator (example: 20031209Z)
MM/DD/YY HH:MM:SS	Two-digit month, separator, two-digit day, separator, last two digits of year, two-digit hour, separator, two-digit minutes, separator, two-digit seconds (example: 12/15/99 03:30:45)
YYMMDD HHMMSS	Last two digits of year, two-digit month, two-digit day, two-digit hour, two-digit minutes, two-digit seconds (example: 991025 033045)
YYYY-MM-DDTHH:MM:SS	Four-digit year, separator, two-digit month, separator, two-digit day, T represents a blank separator, two-digit hour, separator, two-digit minutes, separator, two-digit seconds (example: 2002-02-02 03:30:45)
YYYY-MM-DD	Four-digit year, separator, two-digit month, separator, two-digit day (example: 2002-02-02)
YYYY-MM	Four-digit year, separator, two-digit month (example: 2002-02)
YYYY	Four-digit year (example: 2002)
--MM-DD	Two dashes, two-digit month, separator, two-digit day (example: -12-02)
---DD	Three dashes, two-digit day (example: ---02)



**Procedure** Use this procedure to format dates and times in a screen entry or print form.

Step	Action
1	Double-click an existing element.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).
2	Click the <b>Validation</b> tab to access validation options.
3	If the minimum allowed field length is longer than the length in the Maximum box, type a new maximum field length in the Maximum box. This value should be the maximum number of digits, including separators, that you expect for the date or time.  <b>Note</b> The value in the Maximum box should be the same as the value in the maximum characters to be displayed box on the screen entry form Display tab or Print form Print tab.
4	Verify that <b>Date/Time</b> is selected in the data-type list, indicating that the field value must be a date or a time.
5	From the form format list, select the format for how you want the date or time to be formatted in the screen entry translation object.
6	Verify that the date or time format selected in the file format list is the correct format that is written to the EDI file.  <b>Note</b> A Date/Time format must be designated for Date/Time-type elements in the file format list, even for hidden fields.
7	Click <b>OK</b> to exit the Element Properties dialog box.

# Formatting Strings

**Introduction** A string-type field or element contains one or more printable characters. If you specify that an element is a string type, you must designate the format by specifying a syntax token.

**Procedure** Use this procedure to format strings in a screen entry or print form.

Step	Action
1	Double-click an existing element.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).
2	Click the <b>Validation</b> tab to access validation options.
3	From the data-type list, select <b>String</b> . This indicates that the element contains characters.
4	From the form format list, select a predefined syntax token to denote that this element must be formatted as the specified syntax token dictates.  <b>Note</b> When you installed Gentran:Server, you assigned a default format to the string fields. This format serves as the basis for character validation. Most U.S. users use a default format that corresponds to ASCII characters (for example, the X syntax token). Most users of Asian or European languages and encoded character sets should use the Free Format (0x01-0xFF).
5	From the file format list, verify that the string format selected is the correct format that is written to the EDI file.  <b>Note</b> A string format must be designated for string-type elements in the file format list, even for those that you hide.
6	Click <b>OK</b> to exit the Element Properties dialog box.

**Related topic** You can create syntax tokens to define ranges of characters and/or numbers that are allowed to be used for a string-type field or element.

## Creating a Syntax Token (Western European languages)

### Introduction

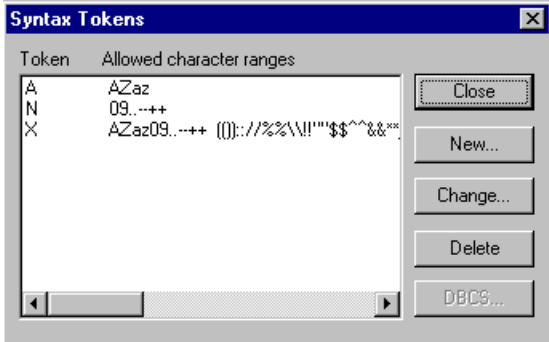
Syntax tokens enable you to designate a “token” that defines ranges of characters and/or numbers that are allowed to be used for a string-type element. You can then use the syntax tokens in the Format lists located on the Validation tab of the Element Properties dialog box. This enables you to define what type of characters should be used while compliance checking each element (i.e., alphanumeric within a certain range, numeric within a certain range, etc.).

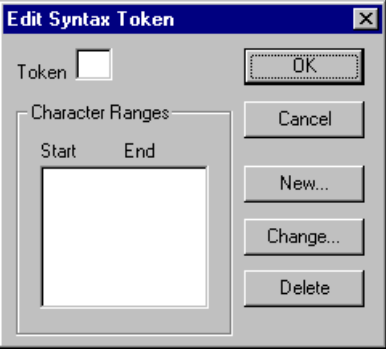
### Note

When you set up a token, it applies only to that form. You may need to set one up for each form that you create.

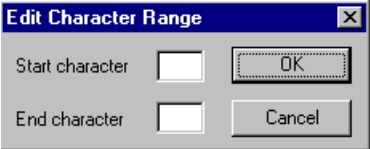
### Procedure

Use this procedure to create a syntax token.

Step	Action
1	<p>Select <b>Syntax Tokens</b> from the Edit menu.</p> <p><b>System Response</b> The system displays the Syntax Tokens dialog box.</p>  <p>The Token column contains the value designated as the syntax token for each existing syntax token. The Token contains a range of characters that, when applied to an element, dictate the way that element must be formatted. If the element is not formatted as specified, the system generates a compliance error.</p> <p>The Allowed character ranges column contains the ranges of characters that are permitted for each existing syntax token. Each range consists of a pair of characters that define the start and end characters.</p> <p><b>Note</b> The <b>DBCS</b> button is only available if you are executing a double-byte version of Windows.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
<p>2</p>	<p>Click <b>New</b>.</p> <p><b>System Response</b> The system displays the Edit Syntax Token dialog box.</p> 
<p>3</p>	<p>In the Token box, type the unique one-character alphanumeric value that the system recognizes as containing the allowed range of characters you designate.</p> <p><b>Note</b> The Token can only be one unique character, upper- or lower-case alphabetic or numeric “1” - “9”.</p> <p>The Character Ranges list contains the character range or ranges that you define for this token. You can define more than one character range for each token. For example, you can define the token “A” as allowing the character range “A” through “Z” <i>and</i> the character range “a” through “z”. This indicates that token “A” only allows upper-case <i>and</i> lower-case alphabets.</p> <p style="text-align: right;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>														
4	<p>Click the <b>New</b> button on the Edit Syntax Tokens dialog box.</p> <p><b>System Response</b> The system displays the Edit Character Range dialog box. This dialog box enables you to create a character range.</p>  <p><b>Note</b> Gentran:Server uses the ANSI Character Set when determining the start-end range.</p> <p>For Western European languages, consult the ANSI character chart (1252 Windows Latin 1). This chart also displays ranges so that you can enter appropriate ranges for the characters in your language.</p> <p>If no chart is available, use the following guidelines:</p> <p>To include all the accented characters in the major languages of Western Europe, add the following ranges:</p> <table border="1" data-bbox="618 1079 1159 1314"> <thead> <tr> <th>Start</th> <th>End</th> </tr> </thead> <tbody> <tr> <td>0xC0</td> <td>0xD6</td> </tr> <tr> <td>0xD8</td> <td>0xF6</td> </tr> <tr> <td>0xF8</td> <td>0xFC</td> </tr> </tbody> </table> <p>Scandinavian users should also add the following in order to include Æ and œ.</p> <table border="1" data-bbox="618 1404 1159 1572"> <thead> <tr> <th>Start</th> <th>End</th> </tr> </thead> <tbody> <tr> <td>0x8C</td> <td>0x8C</td> </tr> <tr> <td>0x9C</td> <td>0x9C</td> </tr> </tbody> </table>	Start	End	0xC0	0xD6	0xD8	0xF6	0xF8	0xFC	Start	End	0x8C	0x8C	0x9C	0x9C
Start	End														
0xC0	0xD6														
0xD8	0xF6														
0xF8	0xFC														
Start	End														
0x8C	0x8C														
0x9C	0x9C														
	(Continued on next page)														



<b>(Contd) Step</b>	<b>Action</b>
5	<p>In the Start character box, type the character that begins the allowed token range.</p> <p><b>Note</b> The Start character and End character can only be one (1) character, upper- or lower-case alphabetic or numeric “1” - “9.”</p> <p><b>Example</b> For example, if the character range you want to define is “B” through “D,” type “B” in the Start character box. If you type a character, such as é, that is not accepted, you need to enter it in hex code. To enter hex characters, “0(zero)x” or “0X,” followed by the hex code. For example, the hex equivalent of é is 0xE9.</p>
6	<p>In the End Character box, type the character that terminates the allowed token range.</p> <p><b>Note</b> The End character can only be one (1) character, upper- or lower-case alphabetic or numeric “1” - “9.”</p>
7	<p>Click <b>OK</b>.</p> <p><b>System Response</b> You return to the Edit Syntax Tokens dialog box.</p>
8	<p>Do you want to add additional character ranges to the syntax token?</p> <ul style="list-style-type: none"> <li>▶ If yes, repeat Step 4 through Step 7 as many times as necessary. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul>
9	<p>Click <b>OK</b>.</p> <p><b>System Response</b> You save the syntax token and return to the Syntax Tokens dialog box.</p>
10	<p>Click <b>Close</b>.</p> <p><b>System Response</b> You exit the Syntax Tokens dialog box.</p>

## Creating a Syntax Token (Far Eastern languages)

### Introduction

The DBCS syntax token function enables you to create a form that accepts double-byte characters. If you are running Gentran:Server on a Chinese, Japanese, or Korean version of Windows, the DBSC button on the Syntax Tokens dialog box is available.

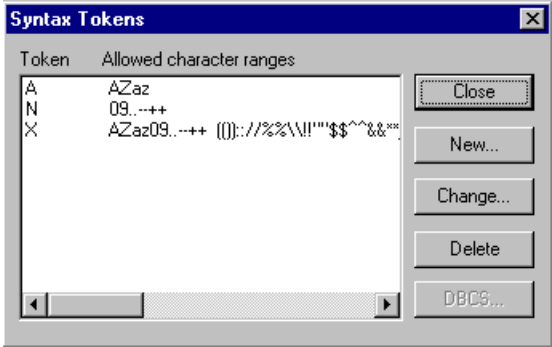

DBCS tokens are displayed only in the DBCS Syntax Tokens dialog box, not in the list in the Syntax Tokens dialog box.

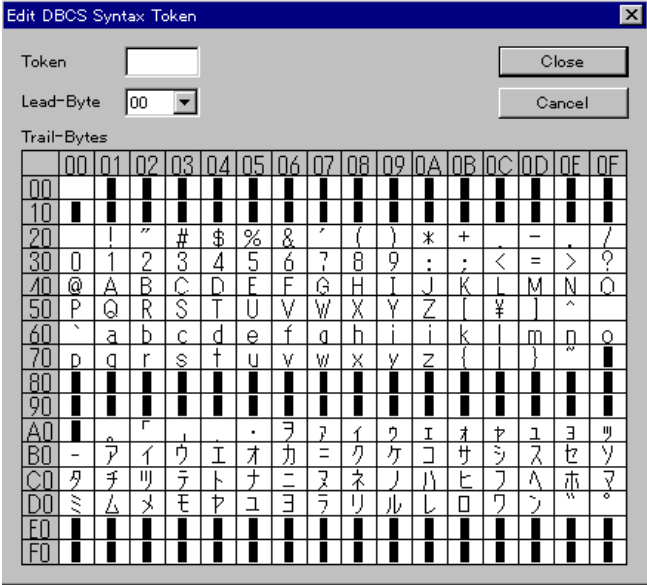
### Note

When you set up a DBCS token, it applies only to that form. You may need to set one up for each form that you create.

### Procedure

Use this procedure to create a DBCS syntax token.

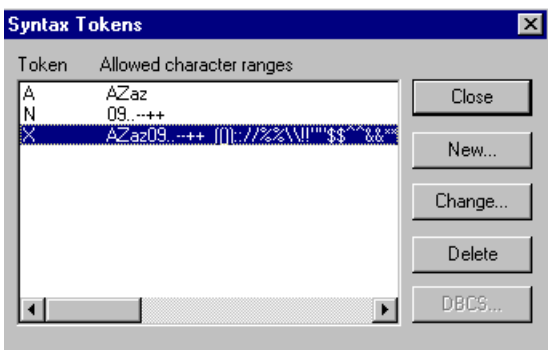
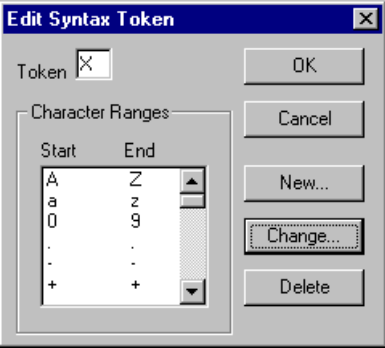
Step	Action
1	<p>Select <b>Syntax Tokens</b> from the Edit menu.</p> <p><b>System Response</b> The system displays the Syntax Tokens dialog box.</p> 
2	<p>Click <b>DCBS</b>.</p> <p><b>System Response</b> The system displays the DCBS Syntax Token dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

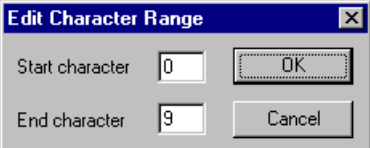
(Contd) Step	Action
3	<p>Click <b>New</b>.</p> <p><b>System Response</b> The system displays the Edit DBCS Syntax Token dialog box.</p>  <p><b>Note</b> Double-byte characters are composed of a lead byte and a trail byte. In the above example, the character A is code point 0041.</p>
4	<p>In the Token box, type the unique one-character alphanumeric value that the system recognizes as containing the allowed range of characters you designate.</p>
5	<p>Do you want to include all characters in the Trail-Bytes section of the dialog box?</p> <ul style="list-style-type: none"> <li>▶ If <i>yes</i>, continue with step.</li> <li>▶ If <i>no</i>, continue with step.</li> </ul>
6	<p>Use the Lead-Byte list to view other characters in the code page on your system. To exclude individual characters or groups of characters from the token, select them (grey them out).</p>
7	<p>Click <b>Close</b> to save the syntax token and return to the Syntax Tokens dialog box.</p>
8	<p>Click <b>Close</b> to exit the Syntax Tokens dialog box.</p>



## Editing a Syntax Token

**Procedure** Use this procedure to edit a syntax token.

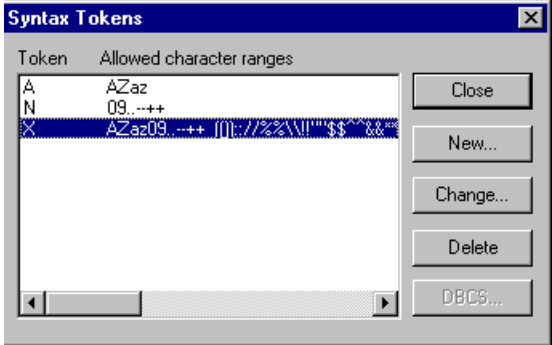
Step	Action
1	<p>Select <b>Syntax Tokens</b> from the Edit menu.</p> <p><b>System Response</b> The system displays the Syntax Tokens dialog box.</p>  <p>The Token column contains the value designated as the syntax token, for each existing syntax token. The Token contains a range of characters that, when applied to an element, dictate the way that element must be formatted. If the element is not formatted as specified, the system generates a compliance error.</p> <p>The Allowed character ranges column contains the ranges of characters that are permitted for each existing syntax token. Each range consists of a pair of characters that define the start and end characters.</p>
2	<p>Select a syntax token and click <b>Change</b> to access the Edit Syntax Token dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>Do you want to change the token value?</p> <ul style="list-style-type: none"> <li>• If yes, type the unique one-character alphanumeric value that the system recognizes as containing the allowed range of characters you designate in the Token box. Proceed to the next step.</li> <li>• If no, proceed to the next step.</li> </ul>
4	<p>Select the character range you want to modify and click <b>Change</b> on the Edit Syntax Tokens dialog box.</p> <p><b>System Response</b> The system displays the Edit Character Range dialog box.</p> 
5	<p>In the Start character box, type the character that begins the allowed token range.</p> <p><b>Note</b> The Start character and End character can only be one (1) character, upper- or lower-case alphabetic or numeric “1” - “9.”</p>
6	<p>In the End Character box, type the character that terminates the allowed token range.</p>
7	<p>Click <b>OK</b> to return to the Edit Syntax Tokens dialog box.</p>
8	<p>To edit additional character ranges for this syntax token, repeat steps 4 - 7 as many times as necessary.</p>
9	<p>Click <b>OK</b> to save the syntax token and return to the Syntax Tokens dialog box.</p>
10	<p>Click <b>Close</b> to exit the Syntax Tokens dialog box.</p>

## Deleting a Syntax Token

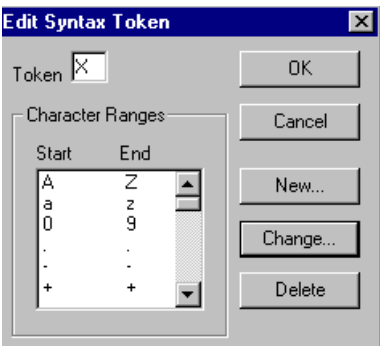
**Introduction** You can delete a syntax token from the system. You can also delete a specific character range from a syntax token.

**Procedure** Use this procedure to delete a syntax token.

Step	Action
1	<p>Select <b>Syntax Tokens</b> from the Edit menu.</p> <p><b>System Response</b> The system displays the Syntax Tokens dialog box.</p> 
2	<p>Highlight the token that you want to delete and click <b>Delete</b>.</p> <p><b>WARNING</b> <b>THE SELECTED ENTRY IS DELETED WITHOUT FURTHER WARNING.</b></p>

## Deleting a character range from a syntax token

**Procedure** Use this procedure to delete a character range from a syntax token.

Step	Action
1	Select <b>Syntax Tokens</b> from the Edit menu.  <b>System Response</b> The system displays the Syntax Tokens dialog box.
2	Select a syntax token and click <b>Change</b> .  <b>System Response</b> The system displays the Edit Syntax Token dialog box.  
3	Highlight the character range that you want to delete, and click <b>Delete</b> .



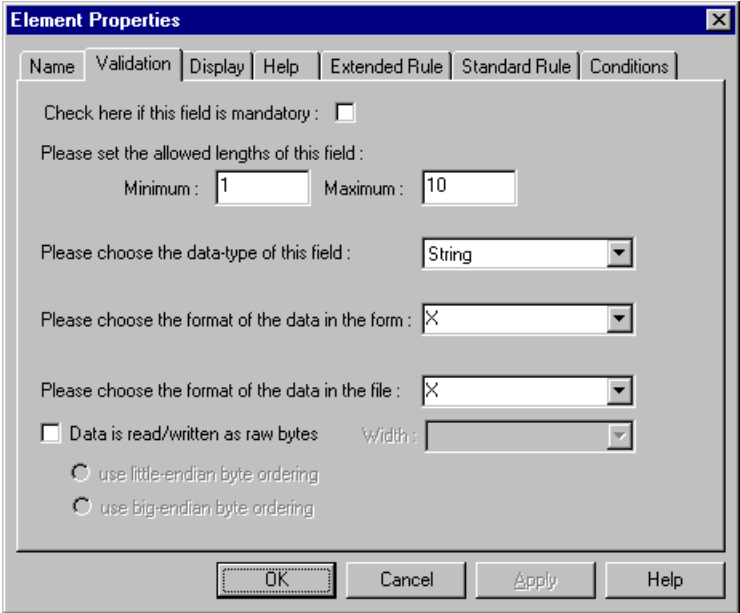
## Using Syntax Tokens

### Introduction

You can use syntax tokens in the Format lists located on the Validation tab of the Element Properties dialog box. This enables you to designate a “token” that defines ranges of characters and/or numbers that are allowed to be used for a string-type element. The use of a syntax token enables you to define what characters should be used while compliance checking this element (i.e., alphanumeric within a certain range, numeric within a certain range, etc.).

### Procedure

Use this procedure to use syntax tokens.

Step	Action
1	Double-click an existing element.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).
2	Click the <b>Validation</b> tab to access validation options.  
3	From the data-type list, select <b>String</b> . This indicates that the element contains characters.  <div style="text-align: right; color: green;">(Continued on next page)</div>



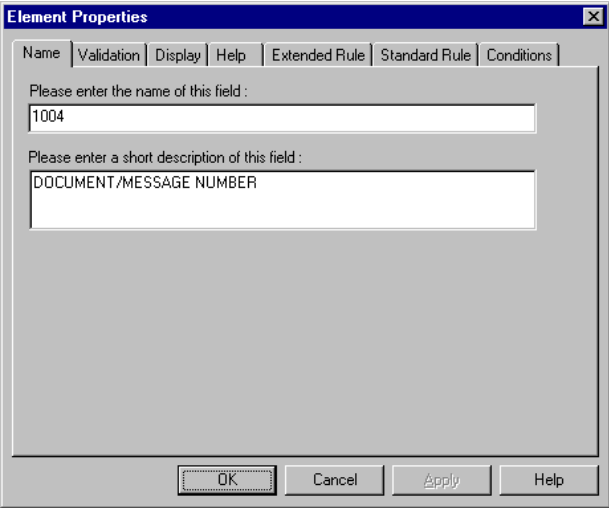
<b>(Contd) Step</b>	<b>Action</b>
4	<p>From the form format list, select a predefined syntax token to denote that this element must be formatted as the specified syntax token dictates.</p> <p><b>Note</b> When you installed Gentrans:Server, you assigned a default format to the string fields. This format serves as the basis for character validation. Most U.S. users use a default format that corresponds to ASCII characters (for example, the X syntax token). Most users of Asian or European languages and encoded character sets should use the Free Format (0x01-0xFF).</p> <p><b>Reference</b> See <i>Creating a Syntax Token (Western European languages)</i> on page 3 - 34 or <i>Creating a Syntax Token (Far Eastern languages)</i> on page 3 - 38 for more information.</p>
5	Click <b>OK</b> to exit the Element Properties dialog box.

---

# Modifying Element Properties

**Introduction** Each segment or composite contains a group of logically-related elements. These elements define the structure of the EDI data that your system needs to process the document.

**Procedure** Use this procedure to modify the properties of an element.

Step	Action
1	<p>Double-click the element you want to modify.</p> <p><b>System Response</b> The system displays the Element Properties dialog box.</p> 
2	<p>Do you want to change the name of the element?</p> <ul style="list-style-type: none"> <li>• If yes, type the new name in the first box on the Name tab. The name is typically the element sequence number. Proceed to the next step.</li> </ul> <p><b>Note</b></p> <p>Do not use spaces or dashes (-) in the Name box. You can use the underscore ( ) to separate words.</p> <ul style="list-style-type: none"> <li>• If no, proceed to the next step.</li> </ul>
3	<p>Do you want to change the description of the element?</p> <ul style="list-style-type: none"> <li>• If yes, type the new description in the second box on the Name tab. The description is used to provide a brief explanation of the element that allows you to differentiate it from similar elements. Proceed to the next step.</li> <li>• If no, proceed to the next step.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
4	Click the <b>Validation</b> tab to access validation options.
5	Do you want to designate the element as “mandatory” (must be present)? <ul style="list-style-type: none"> <li>▶ If yes, select the first check box on the Validation tab.</li> <li>▶ If no, proceed to the next step.</li> </ul>
6	Do you want to change the minimum length of the element? <ul style="list-style-type: none"> <li>▶ If yes, type the minimum length in the Minimum box. Proceed to the next step.</li> </ul> <p><b>Note</b> If the data is less than the minimum length, a compliance error is generated when translation occurs.</p> <ul style="list-style-type: none"> <li>▶ If no, proceed to the next step.</li> </ul>
7	Do you want to change the maximum length of the element? <ul style="list-style-type: none"> <li>▶ If yes, type the maximum length in the Maximum box. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul>
8	From the data-type list, select the type of the element. Valid values are: <ul style="list-style-type: none"> <li>▶ String = alpha element</li> <li>▶ Number = numeric or real element</li> <li>▶ Date/Time = date or time element</li> <li>▶ Bin Data = binary data (only available if you select “Binary” on the Special tab of the EDI Segment Properties dialog box)</li> <li>▶ Bin Length = length of binary data (only available if you select “Binary” on the Special tab of the EDI Segment Properties dialog box)</li> </ul> <p><b>Reference</b> See <i>Modifying Segment Properties</i> on page 3 - 14 for more information about selecting “Binary” on the Special tab.</p>

<b>(Contd) Step</b>	<b>Action</b>
9	<p>From the form format list, select how the data element is formatted on the form.</p> <p><b>Note</b> The choices for this list depend on the type of the element you selected from the data-type list. If you choose Number or Date/Time in the data-type list, you can select the data format from the data format list. If you selected String from the data-type list, you can select a syntax token to denote that this element must be formatted as the specified syntax token dictates (the default syntax token is “X”).</p> <p>When you installed Gentrans:Server, you assigned a default format to the string fields. This format serves as the basis for character validation. Most U.S. users use a default format that corresponds to ASCII characters (for example, the X syntax token). Most users of Asian or European languages and encoded character sets should use the Free Format (0x01-0xFF).</p> <p><b>Reference</b> See <i>Creating a Syntax Token (Western European languages)</i> on page 3 - 34 or <i>Creating a Syntax Token (Far Eastern languages)</i> on page 3 - 38 for more information.</p>
10	<p>From the file format list, select how the data element is formatted in the EDI file.</p> <p><b>Note</b> The choices for this list depend on the type of the element you selected from the data-type list.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

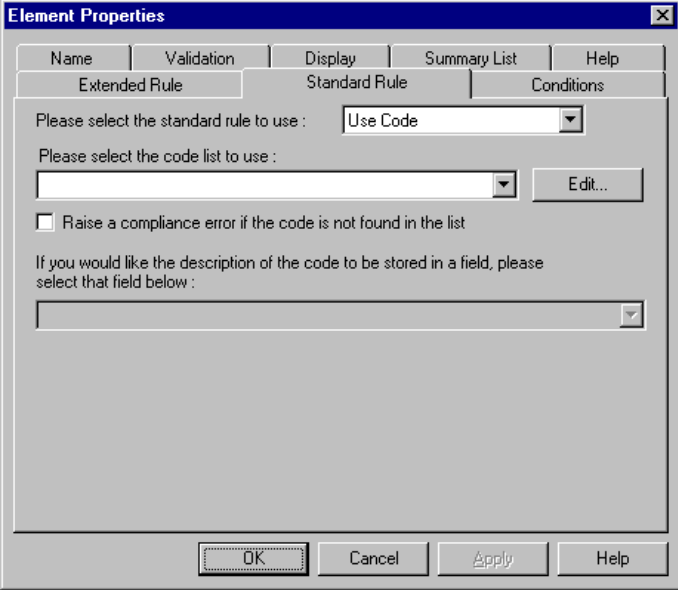
<b>(Contd) Step</b>	<b>Action</b>		
11	Use this table to determine your next step.		
	<b>IF this is a...</b>	<b>AND you want to...</b>	<b>THEN...</b>
	screen entry form,	you want to change display options,	click the <b>Display</b> tab. Proceed to the next step.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about display options.
print form,	you want to change print options,	click the <b>Print</b> tab. Proceed to the next step.  <b>Reference</b> See Customizing Screen Entry and Print Forms, chapter 7 in this guide, for more information about print options.	
12	<p>Is the element in a repeating group or segment?</p> <ul style="list-style-type: none"> <li>• If yes, do you want to change the summary list options?                             <ul style="list-style-type: none"> <li>— If yes, click the <b>Summary List</b> tab. Proceed to the next step.</li> <li>— If no, proceed to the next step.</li> </ul> </li> </ul> <p><b>Note</b> See <i>Creating Definitions for a List Box</i> on page 7 - 9 for more information about lists.</p> <ul style="list-style-type: none"> <li>• If no, proceed to the next step.</li> </ul>		
13	Click <b>OK</b> to save the changes to the element.		

## Storing EDI Code Value Descriptions

**Introduction** Gentran:Server enables you to store code value descriptions in code storage fields. This feature is useful for elements that contain an EDI code, and for which you want to print the description of the code instead of or in addition to the code itself. You set up a standard rule to direct the system to look up an EDI code value, find the description for that value, and to display the description in another field.

**When to use** You might use this feature if, for example, you have a code list that contains first names and you want the corresponding last names loaded in the descriptive element.

**Procedure** Use this procedure to store an EDI code value description in a code storage field.

Step	Action
1	Double-click the element for which you need to load a code value description.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, select <b>Use Code</b> .  <b>System Response</b> Your dialog box should look like this.  

(Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
4	<p>From the code list, select the code list table that the data in this element will be validated against. If the Tables list is empty, you need to load a code table.</p> <p><b>Reference</b> Please see <i>Loading a Code List Table from the Standard</i> on page 5 - 99 for more information.</p>
5	<p>If you need to specify that, for compliance reasons, the element <i>must</i> contain one of the codes from the specified table (nothing else is valid for that field), select the compliance error check box.</p>
6	<p>From the store description list, the element to which you want the description of the code item (that is used) to be mapped to when the selection is made.</p> <p>For example, if the code you used is “BP,” it would be useful to view the description of the code (“Buyer’s Part Number”). If you selected element “0350” from the Store Field list, the description for the BP code is mapped to element “0350.”</p>
7	<p>Click <b>OK</b> to add this standard rule to the element.</p>

**Related topic**

Now that you have set up the EDI code value description, and directed the system to display the description of the EDI code value in an element, you can format and customize that element.

**Reference**

See *Hiding Fields* on page 7 - 3 for information about how to hide the EDI code value field.

See *Loading a Code List Table from the Standard* on page 5 - 99 for information about using code list tables and using code item descriptions.



## Defining Relational Conditions

### Introduction

You can use relational conditions to connect elements together for syntax or compliance reasons. You can also view the conditional relationships between elements, as provided by the standard.

### Example

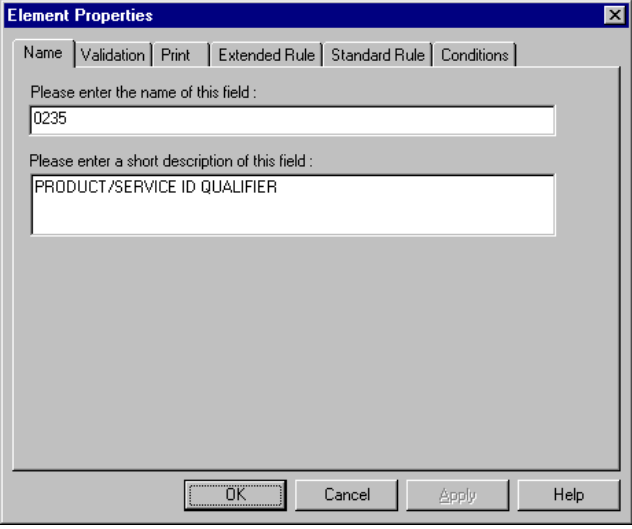
Element A is invalid unless Element B is present. Therefore, if you set up a condition that pairs Elements A and B, the system generates a compliance error if one of those elements is not present.

### WARNING

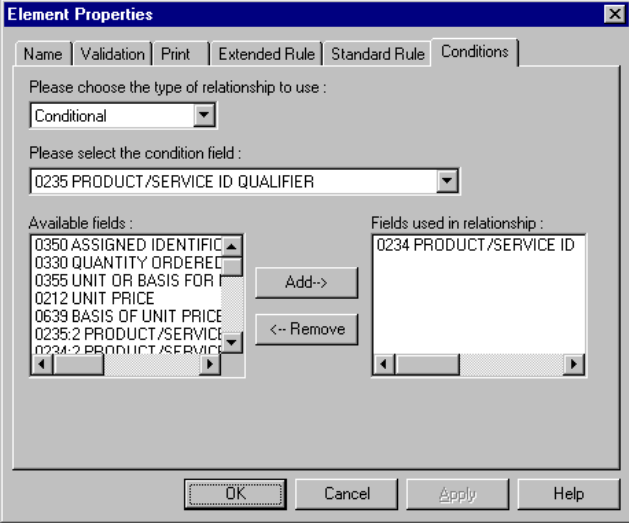
**THE SYSTEM ENABLES YOU TO EDIT EDI CONDITIONAL RELATIONSHIPS, BUT IF YOU DO, YOU MAY ALTER THE COMPLIANCE CHECKING CRITERIA. WE RECOMMEND THAT YOU VIEW EDI CONDITIONS ONLY.**

### Procedure

Use this procedure to define, modify, and view element relational conditions.

Step	Action
1	<p>Double-click the element you want to modify.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>



(Contd) Step	Action
2	<p>Click the <b>Conditions</b> tab.</p> <p><b>System Response</b> The system displays the relational condition property sheet.</p>  <p>This tab enables you to view, modify, or define field relational conditions. Continue with the following steps to define or modify conditions.</p>
3	<p>Select the field connection condition from the type of relationship list. Valid values are:</p> <ul style="list-style-type: none"> <li>▶ <b>Paired/Multiple</b> – if any of the specified elements are present then all elements must be present.</li> <li>▶ <b>Required</b> – at least one of the specified elements must be present.</li> <li>▶ <b>Exclusion</b> – no more than one of the specified elements may be present.</li> <li>▶ <b>Conditional</b> – if the first Condition element is present, the rest of the elements must also be present.</li> <li>▶ <b>List Conditional</b> – if the first Condition element is present, at least one of the specified elements must also be present.</li> </ul>
4	<p>Select the first element from the condition field list. This is the element on which the conditional relationship hinges if you chose “Conditional” or “List Conditional” from the type of relationship list.</p> <p><b>System Response</b> The condition field list is only active if you chose either “Conditional” or “List Conditional” from the type of relationship list.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>
5	<p>The Available fields list contains all the elements in the translation object that are valid to be used in a condition at this point. Highlight an element or elements in the Available fields list and click the <b>ADD--&gt;</b> button to move the elements to the Fields used in relationship list, to include the elements as a part of the conditional relationship.</p> <p>To highlight a block of adjacent items, click the first item and then press the <b>SHIFT</b> key and click the last item. To highlight several items that are not adjacent to each other, press the <b>CTRL</b> key and click the items.</p>
6	<p>The Fields used in relationship list contains the elements that you selected to be a part of the conditional relationship. To remove the elements from the conditional relationship, highlight an element or elements in the Fields used in relationship list and click the <b>Remove</b> button to move the elements back to the Available fields list.</p>
7	<p>Click <b>OK</b> to add the conditional relationship to the element.</p>

---

---

# Completing Your Form

---

## Contents

- ▶ Overview . . . . . 4 - 2
  - ▶ Compiling a Translation Object . . . . . 4 - 3
  - ▶ Printing a Report . . . . . 4 - 6
  - ▶ Registering a Translation Object. . . . . 4 - 7
  - ▶ Creating a Trading Partnership . . . . . 4 - 8
  - ▶ Validating a Translation Object . . . . . 4 - 9
-

## Overview

**In this chapter**

---

This chapter describes how to complete your form.

---

# Compiling a Translation Object

**Introduction**

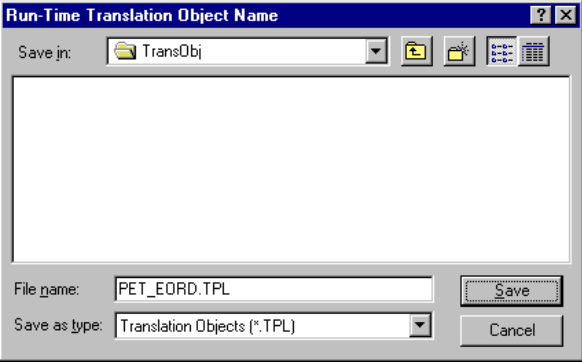
You use the Compile function after the form is completed and saved. The Compile function compiles the form and creates a translation object. The form that you create using Gentran:Server is a **source form**. When that source form is compiled, the result is a compiled **translation object**.

**Note**

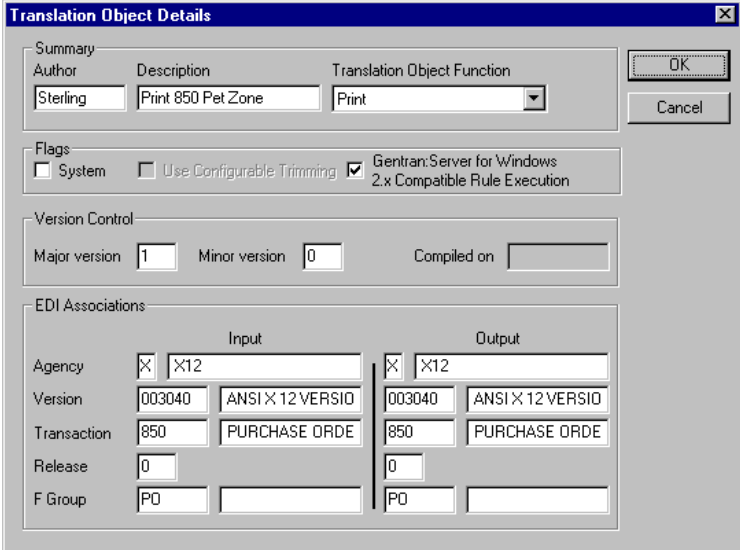
The translation object must be registered with the Gentran:Server system before you can use it.

**Procedure**

Use this procedure to compile a form and to generate a translation object.

Step	Action
1	<p>Select <b>Compile</b> from the File menu.</p> <p><b>System Response</b> The system displays the Run-Time Translation Object Name dialog.</p>  <p>The system defaults the name of the translation object (.TPL file) with the same filename (one to eight characters long) as you named the form (.STP file). Preserving the same filename (with different file extensions) means that the relationship between the source form and the compiled translation object will remain evident. For example, if the source form name is MWT_850.STP, the compiled translation object name is MWT_850.TPL.</p> <p><b>WARNING</b> <b>DO NOT TO OVERLAY THE SOURCE FORM WITH THE COMPILED TRANSLATION OBJECT. USE THE .TPL FILE EXTENSION TO DISTINGUISH THE TRANSLATION OBJECT.</b></p> <p style="text-align: right;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>
2	<p>From the Save in list, change the drive\folder where the compiled translation object is stored, if necessary.</p> <p><b>WARNING</b>  <b>DO NOT STORE THE COMPILED TRANSLATION OBJECT IN THE GENSRVNT\REGTRANSOBJ FOLDER. THIS FOLDER IS RESERVED FOR STORING A COPY OF EACH TRANSLATION OBJECT YOU REGISTER WITH GENTRAN:SERVER.</b></p>
3	<p>Click <b>Save</b>.</p> <p><b>System Response</b>                      The system compiles the form and creates a translation object. The Compile Errors dialog displays. it indicate errors or warnings that occurred during the compile process.</p>
4	<p>Does the Compile Errors dialog note any errors or warnings that occurred during processing?</p> <ul style="list-style-type: none"> <li>▶ If yes, click <b>OK</b> to exit the Compile Errors dialog. Correct the problems and repeat <b>Step 1</b> through <b>Step 3</b> to recompile the translation object. Proceed to the next step.</li> <li>▶ If no, click <b>OK</b> to exit the Compile Errors dialog. Proceed to the next step.</li> </ul> <p><b>Note</b>                      The date that the translation object was compiled on is automatically loaded into the Compiled on box on the Translation Object Details dialog.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
5	Select <b>Save</b> from the File menu.  <b>System Response</b> You save the source form with the Compiled on date.

---

**Related topic**

This translation object must be registered with the Gentran:Server system before you can use it.

**Reference**

See the *Gentran:Server for Windows User's Guide* for information on registering a translation object.

---

## Printing a Report

---

### Introduction

The Print function available from the File menu enables you to print a customized report in either hard copy or file format. Using the Print option, you can select the following options:

- ▶ Branching diagram
  - ▶ Record Details
  - ▶ Extended Rules
  - ▶ Code Lists
-



# Registering a Translation Object

---

**Introduction**

After you have compiled your translation object, it must be registered with the Gentran:Server system before you can use it.

**Reference**

See the *Gentran:Server for Windows User's Guide* for information on registering a translation object.

---

## Creating a Trading Partnership

---

### **Introduction**

After you have registered your translation object with the Gentran:Server system, you must associate it with a trading partner.

### **Reference**

See the Using Partners chapter of the *Gentran:Server for Windows User's Guide* for information.

---

## Validating a Translation Object

---

### **Introduction**

For a screen entry translation object, create a new document in Document Editor, using that translation object. For a print translation object, print a document. Verify that the output is correct.

### **Reference**

See the *Gentran:Server for Windows User's Guide* for information on using the Document Editor.

---



---

# Using Standard Rules

<b>Contents</b>	<ul style="list-style-type: none"> <li>    ▶ Introduction . . . . . 5 - 3</li> </ul>
	<b>Using the Select Standard Rule . . . . . 5 - 4</b>
	<ul style="list-style-type: none"> <li>    ▶ Overview . . . . . 5 - 4</li> <li>    ▶ Using Select in a Form . . . . . 5 - 6</li> <li>    ▶ Using Information from the Partner Definition . . . . . 5 - 11</li> <li>    ▶ Using Information from Location Tables . . . . . 5 - 14</li> <li>    ▶ Using Information from Lookup Tables . . . . . 5 - 16</li> <li>    ▶ Using Information from Cross-Reference Tables . . . . . 5 - 18</li> </ul>
	<b>Using the Update Standard Rule . . . . . 5 - 20</b>
	<ul style="list-style-type: none"> <li>    ▶ Overview . . . . . 5 - 20</li> <li>    ▶ Using Update in a Form . . . . . 5 - 22</li> <li>    ▶ Update Example: Document Name . . . . . 5 - 24</li> </ul>
	<b>Using the Use System Variable Standard Rule . . . . . 5 - 27</b>
	<ul style="list-style-type: none"> <li>    ▶ Overview . . . . . 5 - 27</li> <li>    ▶ Using a System Variable in a Form . . . . . 5 - 28</li> <li>    ▶ Using the System Date and Time . . . . . 5 - 30</li> </ul>
	<b>Using the Use Constant Standard Rule . . . . . 5 - 32</b>
	<ul style="list-style-type: none"> <li>    ▶ Overview . . . . . 5 - 32</li> <li>    ▶ Using Constants in a Form . . . . . 5 - 34</li> <li>    ▶ Use Constant Example . . . . . 5 - 36</li> </ul>
	<b>Using Literal Constants and Qualifying Relationships . . . . . 5 - 40</b>
	<ul style="list-style-type: none"> <li>    ▶ Overview . . . . . 5 - 40</li> <li>    ▶ Defining Literal Constants . . . . . 5 - 41</li> <li>    ▶ Editing Literal Constants . . . . . 5 - 43</li> <li>    ▶ Deleting Literal Constants . . . . . 5 - 44</li> <li>    ▶ Using Literal Constants . . . . . 5 - 45</li> <li>    ▶ Generating Qualifiers . . . . . 5 - 47</li> </ul>
	<b>Using the Use Accumulator Standard Rule . . . . . 5 - 49</b>
	<ul style="list-style-type: none"> <li>    ▶ Overview . . . . . 5 - 49</li> </ul>

▶ Using an Accumulator in a Form . . . . .	5 - 51
▶ Use Accumulator Example: Counting Line Items . . . . .	5 - 54
▶ Use Accumulator Example: Calculating Hash Totals. . . . .	5 - 61
▶ Use Accumulator Example: Resetting and Calculating a Value Total . . . . .	5 - 64
<b>Using the Loop Count Standard Rule . . . . .</b>	<b>5 - 73</b>
▶ Overview . . . . .	5 - 73
▶ Using Loop Count in a Form. . . . .	5 - 75
<b>Using the Use Code Standard Rule . . . . .</b>	<b>5 - 77</b>
▶ Overview . . . . .	5 - 77
▶ Using Use Code in a Form . . . . .	5 - 80
▶ Use Code Example: Selecting a Code List Item . . . . .	5 - 83
<b>Using Code List Tables . . . . .</b>	<b>5 - 86</b>
▶ Overview . . . . .	5 - 86
▶ Defining a Code List . . . . .	5 - 88
▶ Modifying a Code List . . . . .	5 - 90
▶ Deleting a Code List or Code List Entry . . . . .	5 - 92
▶ Importing a Code List . . . . .	5 - 95
▶ Exporting a Code List . . . . .	5 - 97
▶ Loading a Code List Table from the Standard . . . . .	5 - 99
▶ Copying and Pasting Code Lists . . . . .	5 - 102
▶ Validating Data Against Code List Tables . . . . .	5 - 104
▶ Loading Code Item Descriptions. . . . .	5 - 106

---

## Introduction

---

### Standard Rules

Gentran:Server provides you with standard rules that you can apply to elements. Standard rules give you access to functions that are necessary for operations that are less involved than extended rules.

Each standard rule is mutually exclusive (you can use only one on a particular element). This chapter defines the standard rules and provides examples (where appropriate) of when you may use them.

#### Note

When an element contains a standard rule a black asterisk appears to the right of the element icon.

---

### Related Topic

Extended rules enable you to use a Gentran:Server proprietary programming language to perform virtually any operation you require. See Using Extended Rules, chapter 6 in this guide, for more information.

---

# Using the Select Standard Rule

## Overview

---

### Introduction

The Select function enables you to select entries from a location table, cross-reference table, partner table, or lookup table (all tables created in the Gentran:Server Partner Editor). You can then map the elements in those tables to one or more elements in the data. The Select function uses the value of the current element to perform the selection. The results of the select can be displayed on the screen or printed on a report.

---

### Using partner related information in forms

The Forms Integration subsystem allows you to use information from the Gentran:Server Partner Editor in your forms via standard rules. You can map information from your partner's profile in the Partner Editor to a selected element in the form. You can also map information from the Internal System Partner. The definition information and location tables that you define for the internal system partner can be used for all partners. The information that you can use includes:

- Partner or Internal System Partner Definition (Name, EDI Code, Application Code)
  - Partner or Internal System Partner Location Table Fields
  - Lookup Table Fields
  - Cross-Reference Table Fields
-



## Standard Rule tab (Select)

This illustration describes the Select function.

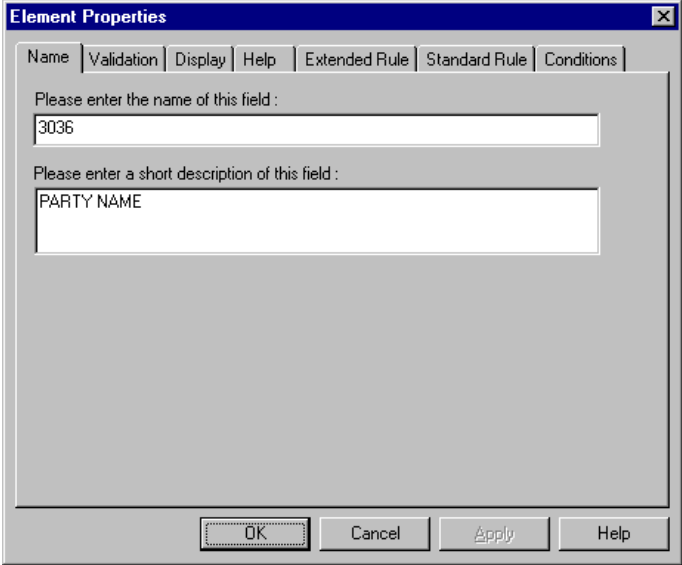
## Parts and Functions

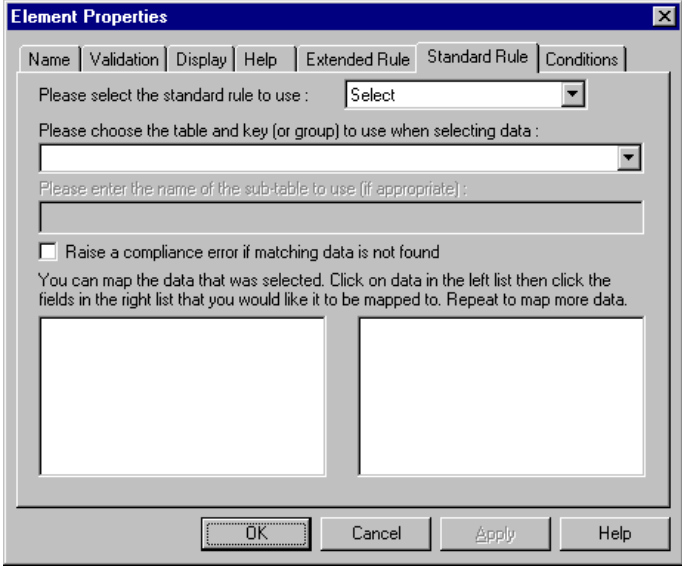
This table describes the parts and functions of the Select Standard Rule.

Part	Function
Please select a standard rule to use list	Identifies standard rule options.
Please choose a table and key to use when selecting data list	Lists available tables in the Partner Editor from which to select data.
Please enter the name of a sub-table to use box	Defines what sub-table information to use.
Compliance error check box	Determines whether a compliance error is generated if no matching data is found.
Data list (left-side list box)	Identifies data that can be mapped.
Map to list (right-side list box)	Identifies where the data can be mapped.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancel changes; exits the dialog box.
<b>Apply</b>	Unavailable.
<b>Help</b>	Launches the Forms Integration online Help system.

## Using Select in a Form

**Procedure** Use this procedure to access the Select function.

Step	Action
1	<p>Double-click an existing element, or create a new element.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

(Contd) Step	Action
3	<p>From the standard rule list, choose <b>Select</b>.</p> 
4	<p>From the table drop-down list, select a table or table and key to use when selecting data. Valid values are:</p> <ul style="list-style-type: none"> <li> <p>• <b>Partner by EDI code</b> indicates that the key is the partner's EDI Code and enables you to map from one of three fields for this partner (Name, EDI Code, Alternate Code) from the partner table. You entered these fields on the Partner Definition dialog box of the Partner Editor.</p> </li> <li> <p>• <b>Partner by alternate code</b> indicates that the key is the partner's Application Code and enables you to map from one of three fields for this partner (Name, EDI Code, Alternate Code) from the partner table. You entered these fields on the Partner Definition dialog box of the Partner Editor.</p> </li> <li> <p>• <b>Partner by partner key</b> indicates that the key is the partner's Profile ID enables you to map from one of three fields for this partner (Name, EDI Code, Alternate Code) from the partner table. You entered these fields on the Partner Definition dialog box of the Partner Editor.</p> </li> <li> <p>• <b>Partner location by name</b> indicates that the name on the Locations dialog box (for the current partner) of the Partner Editor is the key and enables you to map from any field in that table. You entered these fields on the Location Entry dialog box of the Partner Editor.</p> </li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
4 (cont.)	<ul style="list-style-type: none"> <li>▶ <b>Partner location by name</b> indicates that the name on the Locations dialog box (for the current partner) of the Partner Editor is the key and enables you to map from any field in that table. You entered these fields on the Location Entry dialog box of the Partner Editor.</li> <li>▶ <b>Partner location primary ref code</b> indicates that the Reference Code 1 field on the Locations dialog box (for the current partner) of the Partner Editor is the key and enables you to map from any field on that dialog box.</li> <li>▶ <b>Partner location by secondary ref code</b> indicates that the Reference Code 2 field on the Locations dialog box (for the current partner) of the Partner Editor is the key and enables you to map from any field on that dialog box.</li> <li>▶ <b>Partner lookup (activates Sub Table field)</b> indicates that the key is the partner lookup table name for this partner and enables you to map from any field in that table. You entered these fields on the Lookup Entry dialog box of the Partner Editor.</li> <li>▶ <b>Partner cross-reference by my item (activates Sub Table field)</b> indicates that the key is the your item (value) on the partner cross-reference table for this partner and enables you to map from any field in that table. You entered these fields on the CrossRef Entry dialog box of the Partner Editor.</li> <li>▶ <b>Partner cross-reference by partner item (activates Sub Table field)</b> indicates that the key is the your partner's item (value) on the partner cross-reference table for this partner and enables you to map from any field in that table. You entered these fields on the CrossRef Entry dialog box of the Partner Editor.</li> <li>▶ Division enables you to map from one of three fields (Name, EDI Code, Alternate Code) for the &lt;Internal System User&gt; (system partner) in that table. You entered these fields on the Partner Locations dialog box of the Partner Editor.</li> <li>▶ <b>Division location by name</b> indicates that the key is the name on the Locations dialog box of the Partner Editor for the &lt;Internal System User&gt; and enables you to map from any field in that table. You entered these fields on the Locations dialog box of the Partner Editor.</li> <li>▶ <b>Division lookup (activates Sub Table field)</b> indicates that the key is the partner lookup table name for the &lt;Internal System User&gt; partner and enables you to map from any field in that table. You entered these fields on the Lookup Entry dialog box of the Partner Editor.</li> <li>▶ <b>Division location by primary ref code</b> indicates that the key is the Reference Code 1 on the Locations dialog box of the Partner Editor for the &lt;Internal System User&gt; and enables you to map from any field on the Locations dialog box.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>

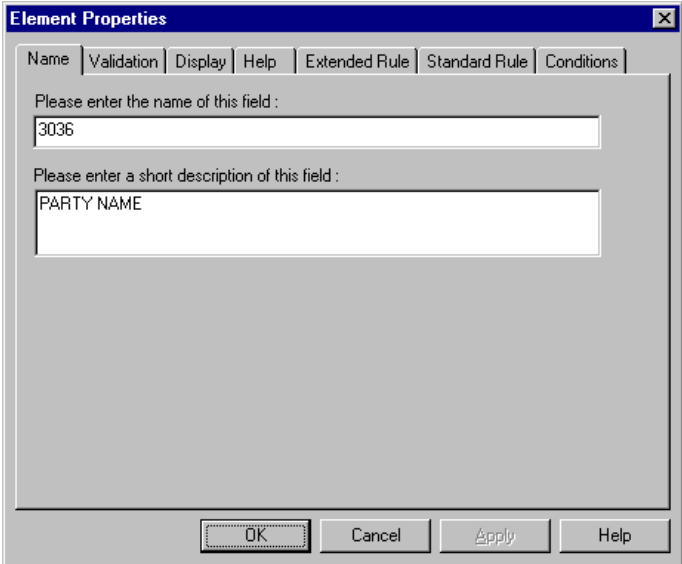
<b>(Contd) Step</b>	<b>Action</b>
4 (cont.)	<ul style="list-style-type: none"> <li>▶ <b>Division location by secondary ref code</b> indicates that the key is the Reference Code 2 on the Locations dialog box of the Partner Editor for the &lt;Internal System User&gt; and enables you to map from any field on the Locations dialog box.</li> <li>▶ <b>Division cross-reference by my item (activates Sub Table field)</b> indicates that the key is the your item (value) on the partner cross-reference table for the &lt;Internal System User&gt; partner and enables you to map from any field in that table. You entered these fields on the CrossRef Entry dialog box of the Partner Editor.</li> <li>▶ <b>Division cross-reference by partner item (activates Sub Table field)</b> indicates that the key is the your partner's item (value) on the partner cross-reference table for the &lt;Internal System User&gt; partner and enables you to map from any field in that table. You entered these fields on the CrossRef Entry dialog box of the Partner Editor.</li> <li>▶ <b>Document record</b> enables you to map from any of the fields within the current document record.</li> <li>▶ <b>Generic envelope segment</b> enables you to map from any of the fields within the current envelope information,</li> <li>▶ regardless of the EDI standard used. This function is normally not used unless you are familiar with the generic envelope table structure for a given standard.</li> <li>▶ <b>Interchange</b> enables you to map from any of the fields within the current interchange record.</li> <li>▶ <b>Group</b> enables you to map from any of the fields within the current group record.</li> </ul>
5	<p>In the Sub Table text box, enter the sub table name (if appropriate) to use.</p> <p><b>Note</b> The sub table box is only active if you select Partner Lookup, Partner cross-reference by my item, Partner xref by partner item, Division lookup, Division cross-reference by my item, or Division cross-reference by partner item from the table and key list.</p>
6	<p>Do you want the system to generate an error if the Select does not find a valid table entry?</p> <ul style="list-style-type: none"> <li>▶ If yes, select the compliance error check box. Proceed to the next step.</li> <li>▶ If no, verify that the check box is clear. Proceed to the next step.</li> </ul> <p><b>Note</b> The default is do not generate an error if the Select is not successful.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

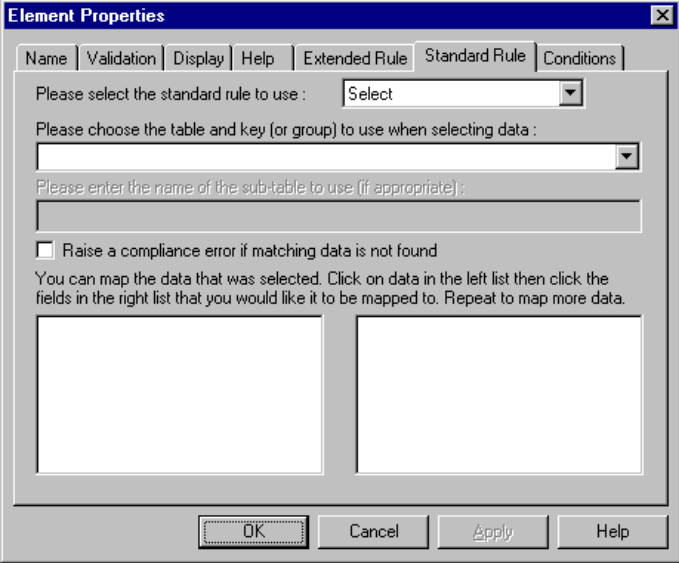
<b>(Contd) Step</b>	<b>Action</b>
7	<p>From the data list displayed on the left-hand side of the dialog box, select the element from the specified table entry from which you want to map the contents.</p> <p><b>Note</b> Entries are displayed in the map from list <i>only</i> after you select a table and key.</p>
8	<p>From the list on the right, choose the element to which you want to map the contents of the map from box. Each element is displayed in this list. A total of eight elements can be mapped using one Select rule. Proceed to the next step.</p> <p><b>Note</b> Entries are displayed in the map to list <i>only</i> after you select a table and key, and these entries are activated <i>only</i> after you highlight an entry in the map from box.</p>
9	Click <b>OK</b> to save your changes.

## Using Information from the Partner Definition

**Introduction** This function enables you to load information from the Partner or Internal System Partner Definition into a screen entry or print translation object. For a screen entry translation object, the system displays a list which allows you to select the entry from the table. For a print translation object, the system prints the information on the report.

**Procedure** Use this procedure to load information from the Partner or Internal System Partner Definition into a screen entry or print translation object.

Step	Action
1	<p>Double-click the element to which the standard rule is applied.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab.</p> <p><b>System Response</b> You access the standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, choose <b>Select</b>.</p> 
4	<p>From the table list, select the key from which the system looks up the trading partner. Valid selections are:</p> <ul style="list-style-type: none"> <li>▶ <b>Partner by EDI code</b> indicates that the key is the partner's EDI Code and enables you to map from one of three fields for this partner (Name, EDI Code, Alternate Code) on the Partner Definition dialog box of the Partner Editor.</li> <li>▶ <b>Partner by alternate code</b> indicates that the key is the partner's Application Code and enables you to map from one of three fields for this partner (Name, EDI Code, Alternate Code) on the Partner Definition dialog box of the Partner Editor.</li> <li>▶ <b>Partner by partner key</b> indicates that the key is the partner's Profile ID and enables you to map from one of three fields for this partner (Name, EDI Code, Alternate Code) on the Partner Definition dialog box of the Partner Editor.</li> <li>▶ <b>Division</b> enables you to map from one of three fields (Name, EDI Code, Alternate Code) for the &lt;Internal System User&gt; (system partner) on the Partner Definition dialog box of the Partner Editor.</li> </ul>
5	<p>Do you want the system to generate an error if the lookup fails?</p> <ul style="list-style-type: none"> <li>▶ If yes, select the <b>compliance error</b> check box. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul>
6	<p>From the map from list, select the field (Name, EDI Code, Application Code) from which you want the contents mapped.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>
7	From the map to list, select the element to which the information from the Partner Editor is mapped.
8	Click <b>OK</b> to add the standard rule.

---

## Using Information from Location Tables

### Introduction

Each Partner Profile and the Internal System Partner Profile can have many associated location tables. The location table can be used to contain address-related information about the partner. You can store your partner's store addresses, warehouse addresses, or "invoice to" addresses. For a screen entry translation object, the system displays a list which allows you to select the entry from the table. For a print translation object, the system prints the information on the report.

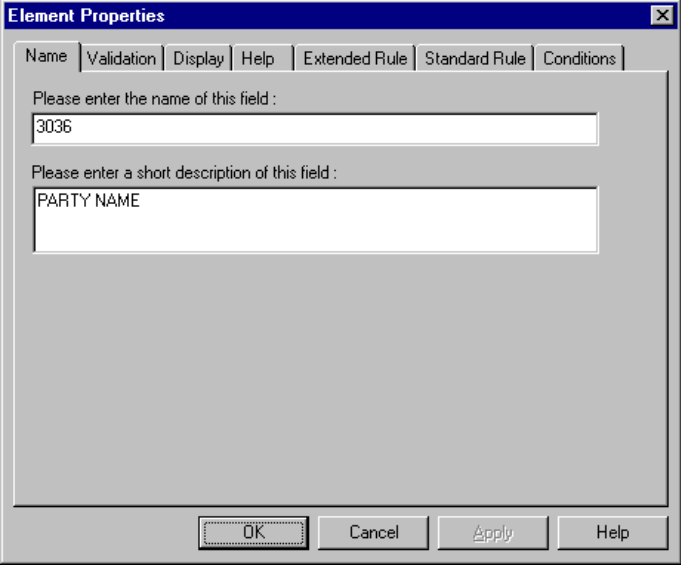
To use information from a location table, you must have created that table already in the Partner Editor.

### Reference

See the *Gentran:Server for Windows User's Guide* for more information on creating a Partner location table.

### Procedure

Use this procedure to load information from a Location Table into a screen entry or print translation object.

Step	Action
1	<p>Double-click the element to which the standard rule is applied.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
2	Click the <b>Standard Rule</b> tab.  <b>System Response</b> You access standard rule options.
3	From the standard rule list, choose <b>Select</b> .
4	From the table list, select the key that is used to look up the Partner Location entry. You are allowed to map from any box on the Locations dialog box. Valid selections are:  <ul style="list-style-type: none"> <li>▶ <b>Partner location by name</b> indicates that the name on the Locations dialog box (for the current partner) of the Partner Editor is the key and enables you to map from any field on that dialog box.</li> <li>▶ <b>Division location by name</b> indicates that the key is the name on the Locations dialog box of the Partner Editor for the &lt;Internal System User&gt; and enables you to map from any field on the Locations dialog box.</li> </ul>
5	Do you want the system to generate an error if the lookup fails?  <ul style="list-style-type: none"> <li>▶ If yes, select the compliance error check box. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul>
6	From the map from list, select the field from which the contents are mapped. You can select from the following fields:  <ul style="list-style-type: none"> <li>▶ Name, Address 1</li> <li>▶ Address 2, Address 3</li> <li>▶ City</li> <li>▶ State</li> <li>▶ Zip</li> <li>▶ Country</li> <li>▶ Telephone Number</li> <li>▶ Fax Number</li> <li>▶ Contact Name</li> <li>▶ Primary Reference</li> <li>▶ Secondary Reference.</li> </ul>
7	From the map to list, select the element to which the information from the Partner Editor is mapped.
8	Click <b>OK</b> to add the standard rule.

## Using Information from Lookup Tables

### Introduction

A lookup table is used to select information related to a value in inbound or outbound data. Each Partner Profile and the Internal System Partner Profile can have many lookup tables associated with it. For a screen entry translation object, the system displays a list which allows you to select the entry from the table. For a print translation object, the system prints the information on the report.

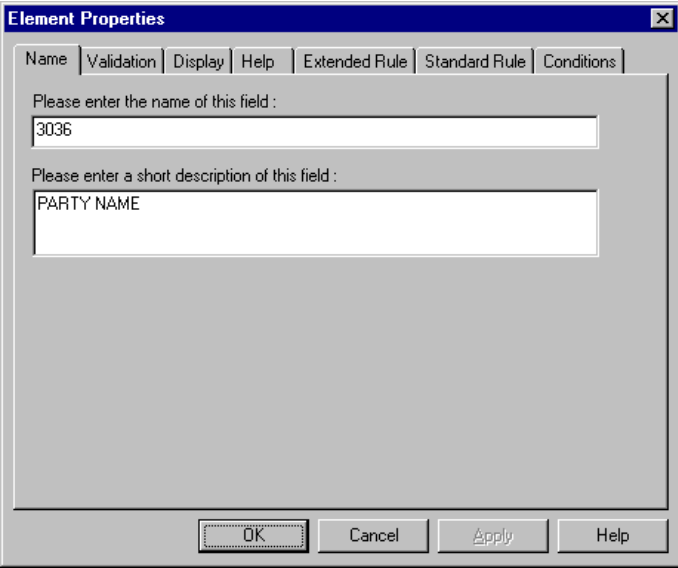
To use information from a lookup table, you must have created that table already in the Partner Editor.

### Reference

See the *Gentran:Server for Windows User's Guide* for more information on creating Partner lookup tables.

### Procedure

Use this procedure to load information from a Lookup Table into a screen entry or print translation object.

Step	Action
1	<p>Double-click the element to which the standard rule is applied.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, choose <b>Select</b> .

(Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
4	<p>From the table list, select the key from which the system looks up the trading partner. Valid selections are:</p> <ul style="list-style-type: none"> <li>▶ <b>Partner lookup (activates Sub Table box)</b> indicates that the key is the partner lookup table name for this partner and enables you to map from any box on the Lookup Entry dialog box of the Partner Editor.</li> <li>▶ <b>Division lookup (activates Sub Table box)</b> indicates that the key is the partner lookup table name for the &lt;Internal System User&gt; partner and enables you to map from any box on the Lookup Entry dialog box of the Partner Editor.</li> </ul>
5	In the sub table box, type the name of the lookup table.
6	If you want the system to generate an error if the lookup fails, select the compliance error check box.
7	From the map from list, select the field (Item, Description, Text 1, Text 2, Text 3, Text 4) from which the contents are mapped.
8	From the map to list, select the element to which the information from the Partner Editor is mapped.
9	Click <b>OK</b> to add the standard rule.

## Using Information from Cross-Reference Tables

### Introduction

Cross-reference tables are used to convert your values to your trading partner's values during outbound processing, or your partner's values to your values during inbound processing. Each Partner Profile and the Internal System Partner Profile can have many lookup tables associated with it. For a screen entry translation object, the system displays a list which allows you to select the entry from the table. For a print translation object, the system prints the information on the report.

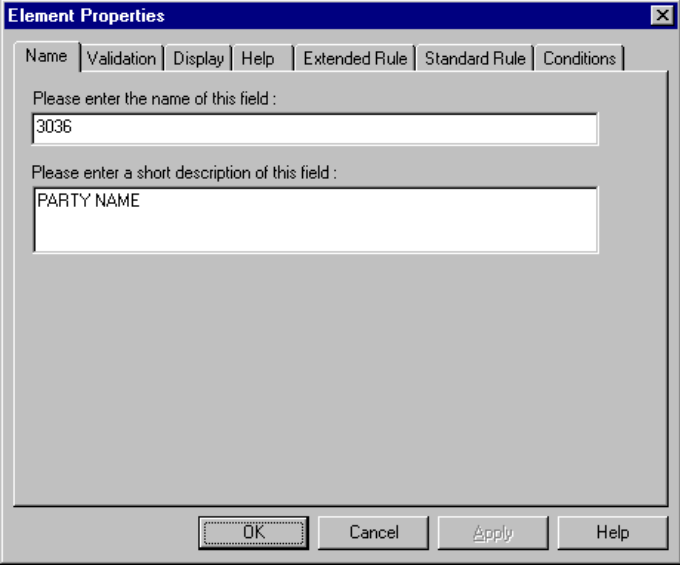
To use information from a cross-reference (XREF) table, you must have created that table already in the *Gentran:Server Partner Editor*.

### Reference

See the *Gentran:Server for Windows User's Guide* for more information on creating Partner cross-reference tables.

### Procedure

Use this procedure to load information from a Cross-Reference (XREF) Table into a screen entry or print translation object.

Step	Action
1	<p>Double-click the element to which the standard rule is applied.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, choose <b>Select</b> .

(Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
4	<p>From the table list, select the key from which the system performs the look up. For Cross-Reference Tables, valid selections are:</p> <ul style="list-style-type: none"> <li>▶ <b>Partner cross-reference by my item</b> indicates that the key is your item (value) on the partner cross-reference table for this partner and enables you to map from any box on the CrossRef Entry dialog box of the Partner Editor.</li> <li>▶ <b>Partner cross-reference by partner</b> indicates that the key is your partner's item (value) on the partner cross-reference table for this partner and enables you to map from any box on the CrossRef Entry dialog box of the Partner Editor.</li> <li>▶ <b>Division cross-reference by my item (activates Sub Table box)</b> indicates that the key is your item (value) on the partner cross-reference table for the &lt;Internal System User&gt; partner and enables you to map from any box on the CrossRef Entry dialog box of the Partner Editor.</li> <li>▶ <b>Division cross-reference by partner item (activates Sub Table box)</b> indicates that the key is your partner's item (value) on the partner cross-reference table for the &lt;Internal System User&gt; partner and enables you to map from any box on the CrossRef Entry dialog box of the Partner Editor.</li> </ul>
5	In the sub table box, type the name of the cross-reference table.
6	If you want the system to generate an error if the look up fails, select the compliance error check box.
7	From the map from list, select the field (My Item, Partner Item, Description, Text 1, Text 2, Text 3, Text 4) from which the contents are mapped.
8	From the map to list, select the element to which the information from the Partner Editor is mapped.
9	Click <b>OK</b> to add the standard rule.

# Using the Update Standard Rule

## Overview

### Introduction

---

The Update function enables you to update a specific field in a document record, envelope segment, interchange, group, current partner, or document, with the contents of the element.

### **WARNING**

**PLEASE NOTE THAT THIS FUNCTION UPDATES THE INTERNAL GENTRAN:SERVER DATABASE TABLES. WE RECOMMEND THAT YOU USE THIS FUNCTION ONLY IF YOU ARE SURE YOU REALLY WANT TO UPDATE THE INTERNAL DATABASE TABLES.**

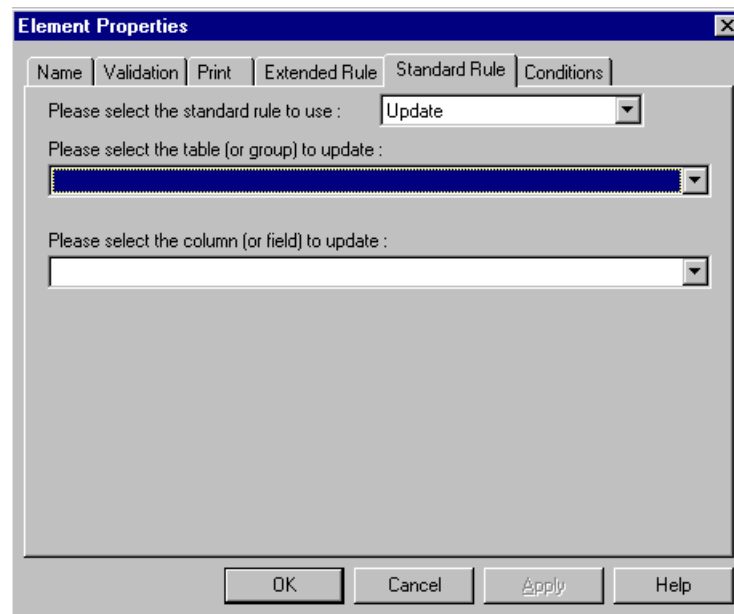
**TYPICALLY, YOU *ONLY* USE THIS FUNCTION IF YOU WANT TO UPDATE THE DOCUMENT NAME AND REFERENCE IN THE DOCUMENT TABLE. ANY OTHER USE OF THIS FUNCTION COULD HAVE DISASTROUS CONSEQUENCES!**

---



### Standard Rule tab (Update)

This illustration describes the Update function.



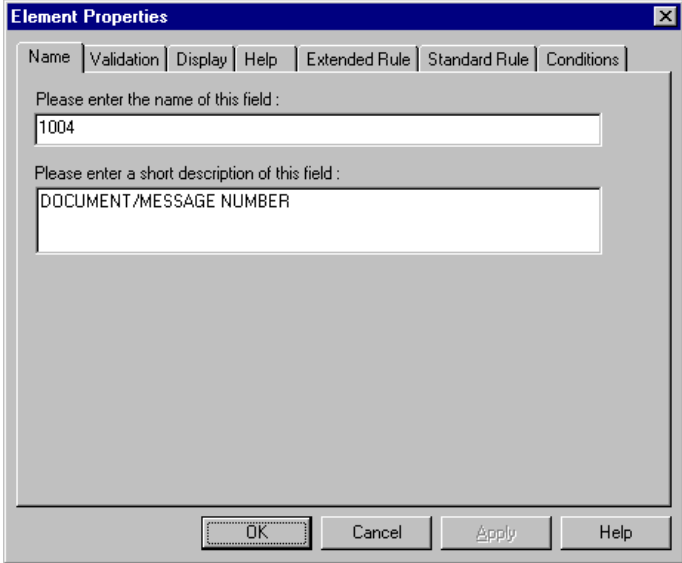
### Parts and Functions

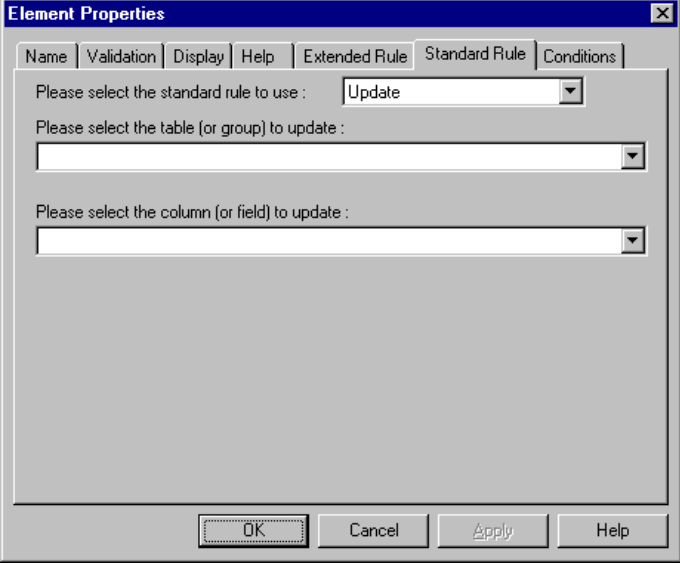
This table describes the parts and functions of the Update Standard Rule.

Part	Function
Please select a standard rule to use list	Identifies standard rule options.
Please select the table or group to update	Identifies tables or groups available for update.
Please select the column or field to update	Identifies which column or field available for update.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Apply</b>	Unavailable.
<b>Help</b>	Launches the Forms Integration online Help system.

## Using Update in a Form

**Procedure** Use this procedure to select the Update function.

Step	Action
1	<p>Double-click an existing element, or create a new element.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, choose <b>Update</b>.</p> 
4	<p>Select the table from the table list that the system updates with the contents of the current element. Valid values are:</p> <ul style="list-style-type: none"> <li>▶ Document record (use to update the document name and reference)</li> <li>▶ Generic envelope segment (advanced use only)</li> <li>▶ Interchange (advanced use only)</li> <li>▶ Group (advanced use only)</li> <li>▶ Current partner (advanced use only)</li> <li>▶ Document (advanced use only)</li> </ul>
5	<p>Select the element from the column (or field) list that the system updates with the contents of the current element. The elements that are available in this list depend on the table selected.</p>

## Update Example: Document Name

### Introduction

We recommend that you set up a Document Name for each form, to make the identification of a document created by this form easier in Gentran:Server. This allows you to differentiate between documents in the document browsers in Gentran:Server.

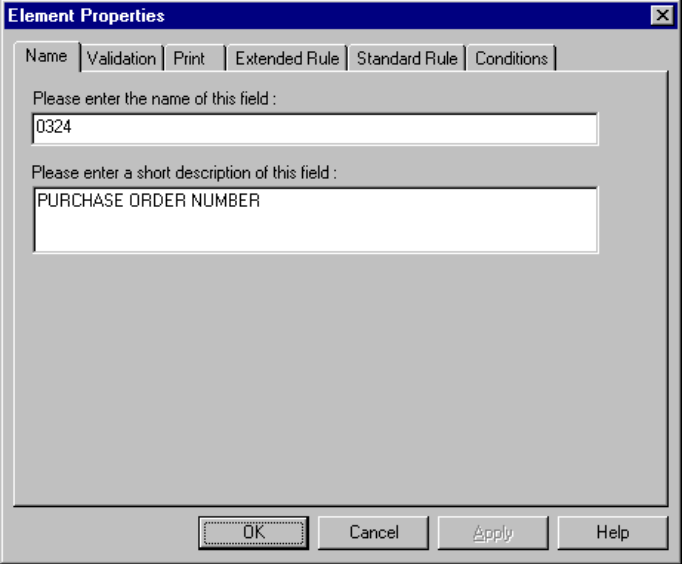
To set up a Document Name, you must select an element in the form that contains data that distinguishes the documents translated by this form. Then, the Name column of the document browsers that contain this document contains the data from the selected element.

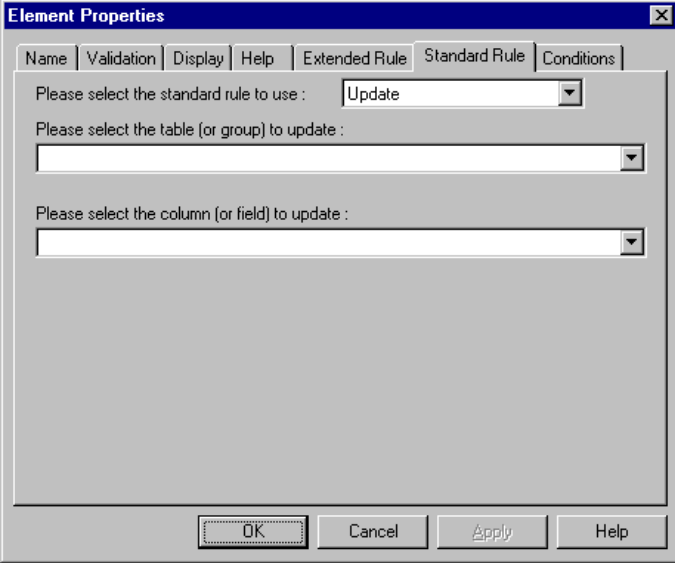
### Note

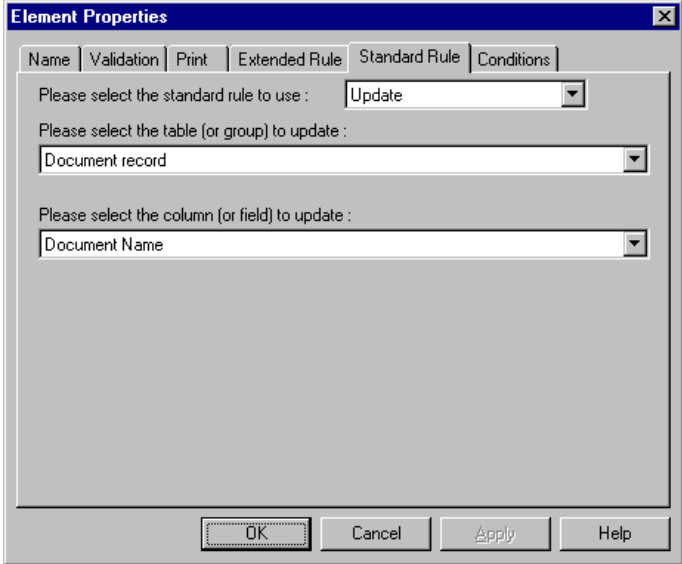
You can choose any element as the Document Name, but the element must only occur once in the document and must have a data type of "String." For a Purchase Order, the document name is typically the Purchase Order Number.

### Procedure

Use this procedure to set up the Document Name for an ANSI X12 Purchase Order.

Step	Action
1	<p>Double-click the BEG03 (0324) element.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, choose <b>Update</b>.</p> 
4	<p>From the table list, choose <b>Document Record</b>. This indicates that you are updating the document record in Gentran:Server.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

(Contd) Step	Action
5	<p>From the column list, choose <b>Document Name</b>. This indicates that you are updating the Name column in the document browsers with the contents of this element.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like the following:</p> 
6	Click <b>OK</b> to set up the document name.

# Using the Use System Variable Standard Rule

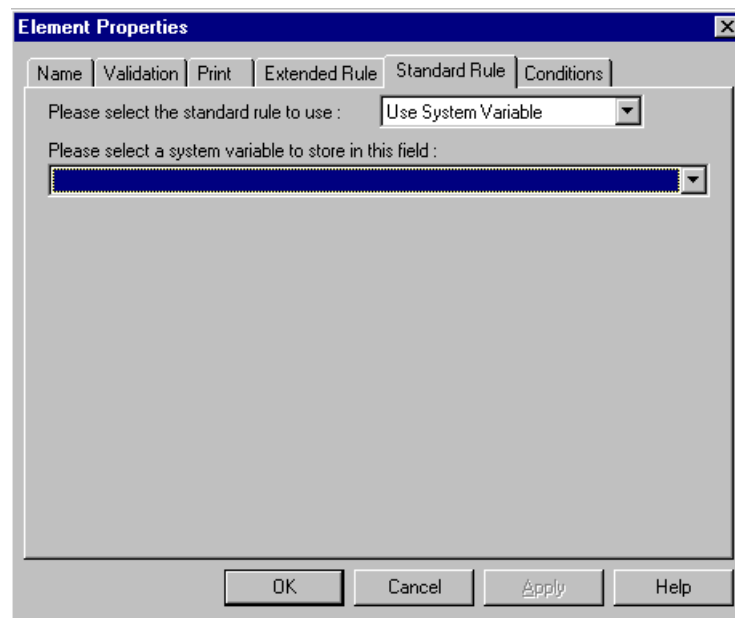
## Overview

### Introduction

The Use System Variable function enables you to set a variable that maps the current date and time to a selected element with a data type of “Date/Time.”

### Standard Rule tab (Use System Variable)

This illustration describes the Use System Variable function.



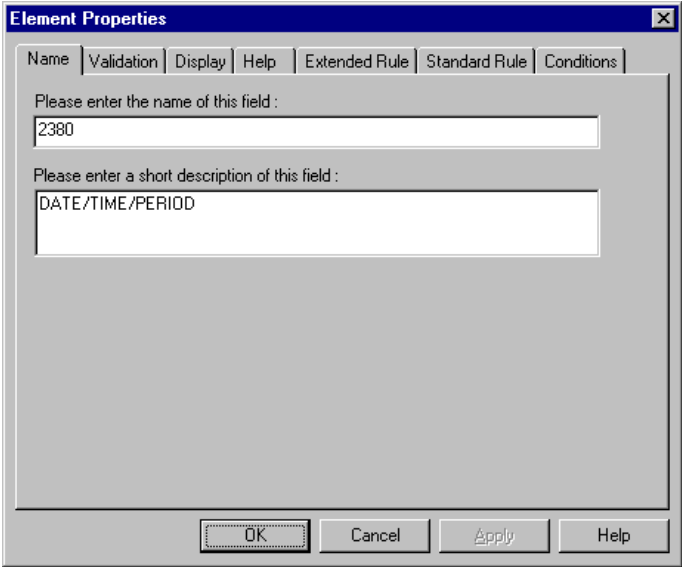
### Parts and Functions

This table describes the parts and functions of the Use System Variable Standard Rule.

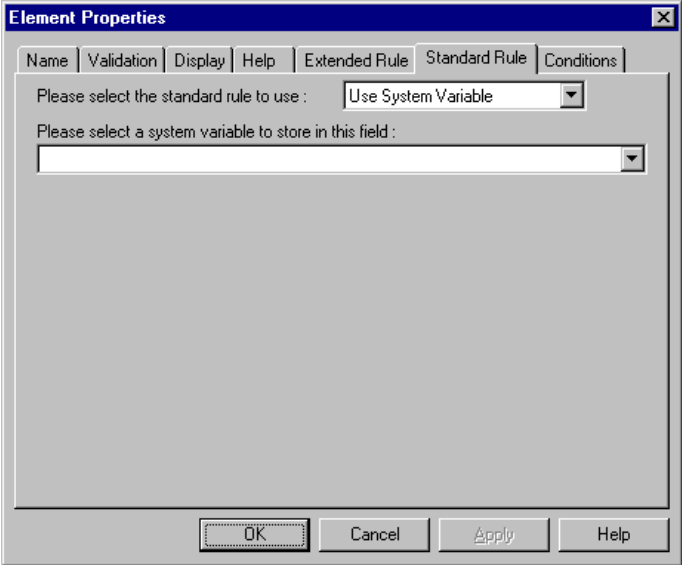
Part	Function
Please select a standard rule to use list	Identifies standard rule options.
Please select a system variable to store in this field list	Identifies available system variables.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Apply</b>	Unavailable.
<b>Help</b>	Launches the Forms Integration online Help system.

## Using a System Variable in a Form

**Procedure** Use this procedure to use a system variable.

Step	Action
1	<p>Double-click an existing element.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

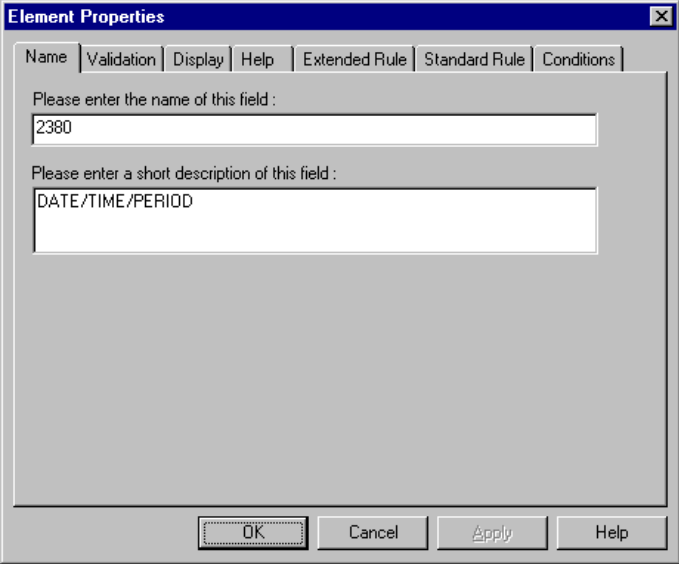


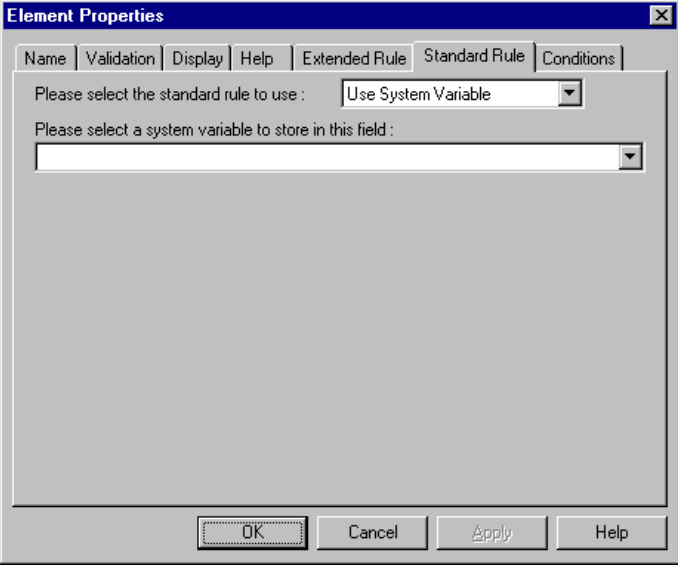
<b>(Contd) Step</b>	<b>Action</b>
3	From the standard rule list, choose <b>Use System Variable</b> . 
4	Select <b>Current date and time</b> from the system variable list to map those variables to the element.

## Using the System Date and Time

**Introduction**    Gentran:Server enables you to update any element (that has a data type of “Date/Time”) with the current system date or time.

**Procedure**    Use this procedure to update an element with the current system date or time.

Step	Action
1	Double-click the desired element to display the Element Properties dialog box (Name tab).  
2	Click the <b>Standard Rule</b> tab to access standard rule options.  (Continued on next page)

(Contd) Step	Action
3	<p>From the standard rule list, choose <b>Use System Variable</b>.</p> 
4	<p>From the system variable list, select <b>Current date and time</b> to update the element with the current system date and time.</p>
5	<p>Click <b>OK</b> to set up the system variable.</p>

# Using the Use Constant Standard Rule

## Overview

### Introduction

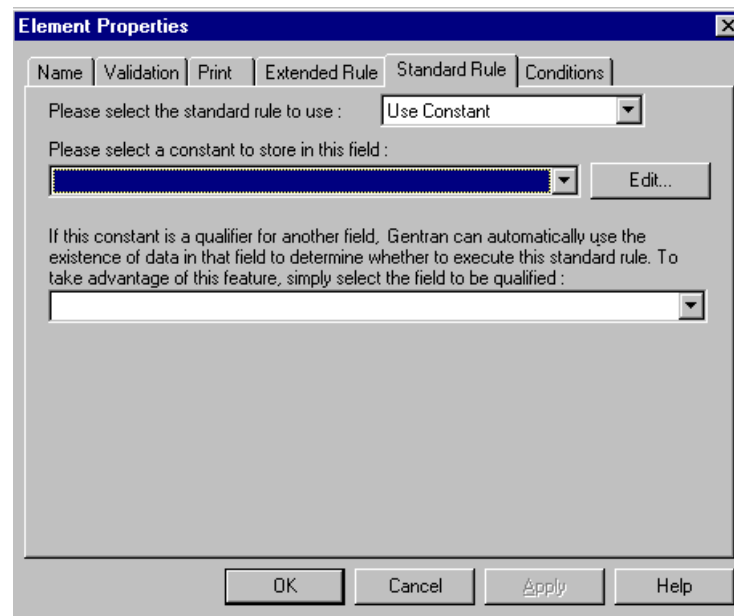
The Use Constant function enables you to move a literal constant value to the specified element and indicate a qualifying relationship with another element.

### Why constants are used

Constants are used in forms to hold information that is needed later in the form in a conditional statement. Typically, EDI qualifiers are typically generated from constants (to indicate a qualifying relationship with another element).

### Standard Rule tab (Use Constant)

This illustration describes the Use Constant function.



### Parts and Functions

This table describes the parts and functions of the Use Constant Standard Rule.

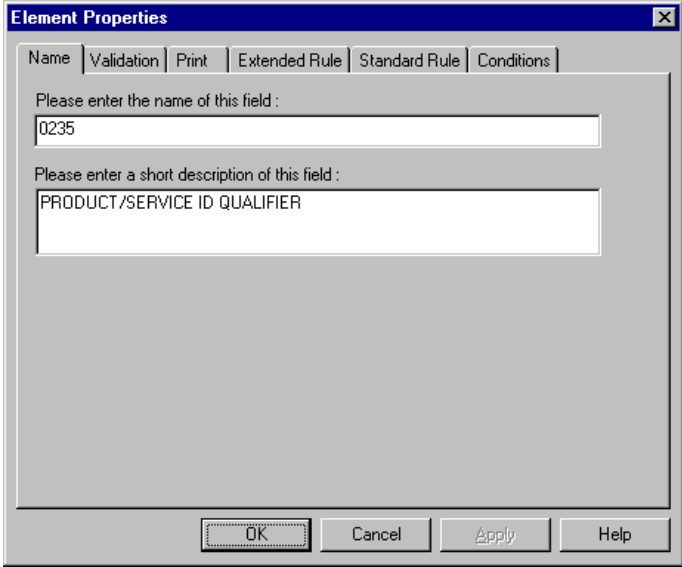
Part	Function
Please select a standard rule to use list	Identifies standard rule options.
Please select a constant to store in this field list.	Displays available literal constants. (Continued on next page)

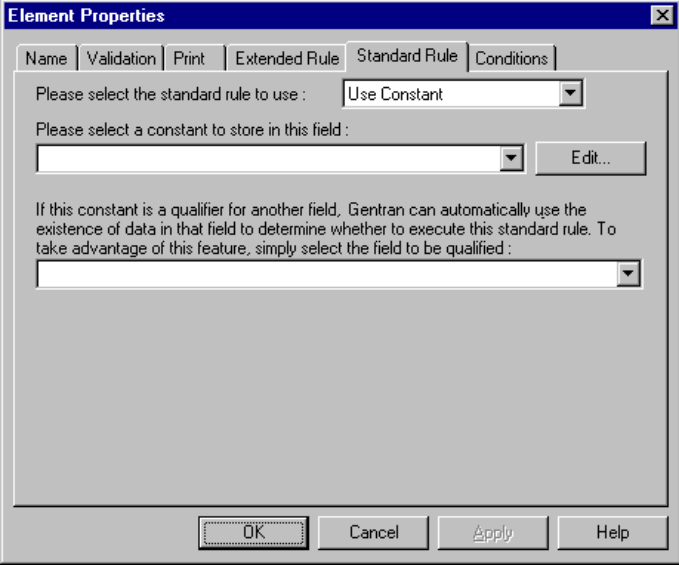
<b>Part</b>	<b>Function</b>
<b>Edit</b>	Accesses the Map Constants dialog box. This dialog box enables you to add, change or delete a literal constant.
Select the field to be qualified	Identifies the field to which you want to establish a qualifying relationship. <b>Note</b> If the qualified element is not generated due to lack of data, the constant is not moved to the current element or field.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Apply</b>	Unavailable.
<b>Help</b>	Launches the Forms Integration online Help system.

---

## Using Constants in a Form

**Procedure** Use this procedure to use a constant.

Step	Action
1	<p>Double-click an existing element.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, select <b>Use Constant</b>.</p> 
4	<p>Select a constant from the constant list. This constant is mapped to the current element.</p> <p><b>Note</b> If the necessary constant is not present in the constant list, you need to create it by using the Map Constants dialog, which you can access by clicking the <b>Edit</b> button.</p> <p>This dialog also allows you to edit and delete literal constants.</p>
5	<p>Select an element from the qualifies list if you want to indicate that it qualifies the selected element (establish a qualifying relationship between the two elements).</p> <p><b>Note</b> If the qualified element is not generated because of a lack of data, the constant is not moved to the current element.</p>

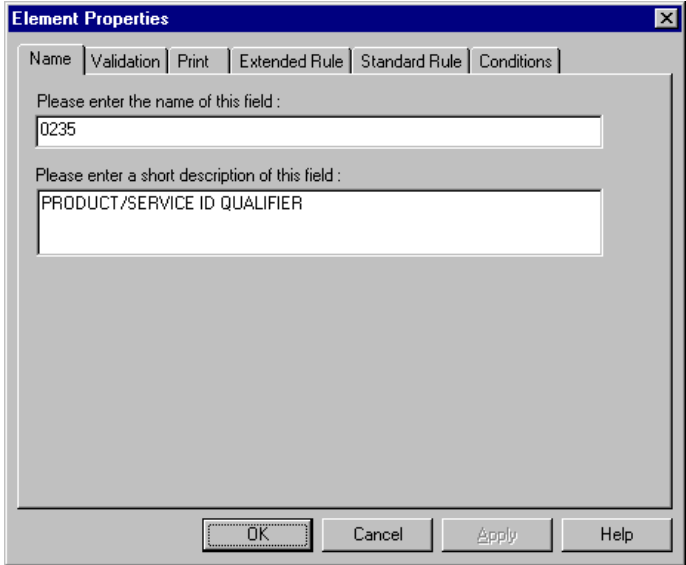
## Use Constant Example

### Introduction

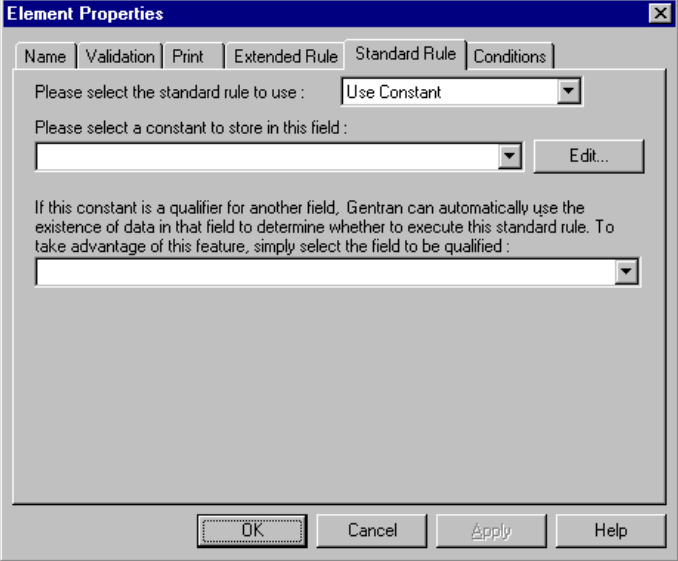
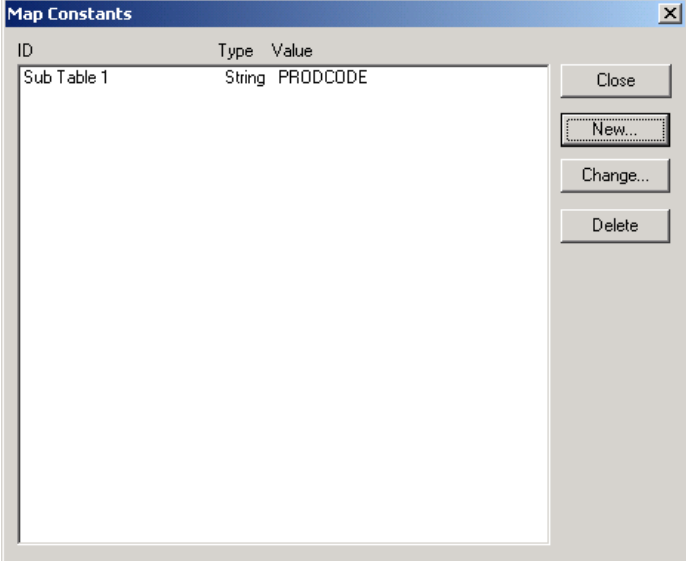
The system uses literal constants as a repository to store information that is used at a later point in the form. Typically, constants are used in a form to generate a qualifier. A qualifier is an element that has a value expressed as a code that gives a specific meaning to the function of another element. A qualifying relationship is the interaction between an element and its qualifier. The function of the element changes depending on which code the qualifier contains.

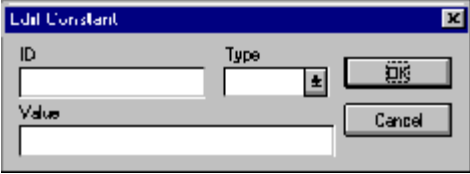
### Procedure

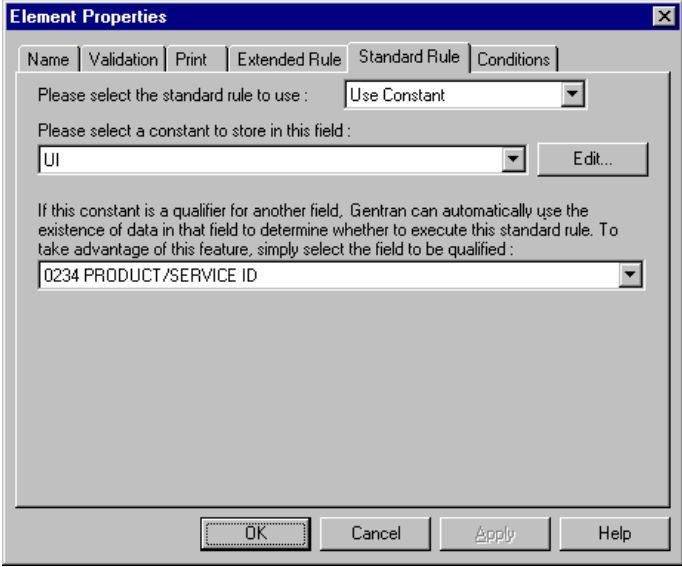
In this example, you use a constant to define a qualifier for a Product/Service ID.

Step	Action
1	<p>Double-click the Product/Service ID Qualifier that you want to use to further define (qualify) the Product/Service ID.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right;">(Continued on next page)</p>



(Contd) Step	Action
3	<p>From the standard rule list, select <b>Use Constant</b>.</p> 
4	<p>Click <b>Edit</b> to access the Map Constants dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

(Contd) Step	Action
5	Click <b>New</b> to access the Edit Constant dialog box. 
6	In the ID box, type the literal constant identifier. This is typically a description of the element in which the constant is used. If you need to refer to the constant in an extended rule, you should use the data from this element.
7	From the Type list, select <b>String</b> because the Product/Service ID Qualifier is formatted as a string data type. This indicates the category of the literal constant.
8	In the Value box, type <b>UI</b> to indicate that the Product/Service ID element must contain the UPC Code. This is the value of the literal constant.
9	Click <b>OK</b> to add the constant to the system.
10	Click <b>Close</b> to exit the Map Constants dialog box. <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
11	<p>The <b>Standard Rule</b> tab of the Element Properties dialog box should now look like the following.</p>  <p><b>Note</b> When you set a constant value for a qualifier field, you know that you never need to override the value in the field. In this instance, you would probably hide the field, as well.</p> <p><b>Reference</b> Please see <i>Hiding Fields</i> on page 7 - 3 for more information.</p>
12	Click <b>OK</b> on the Element Properties dialog box and the qualifying relationship between the two elements is established.

# Using Literal Constants and Qualifying Relationships

## Overview

---

**Introduction** This section describes how to use literal constants and qualifying relationships.

---

**Literal Constants** Literal constants are used by the system as a repository to store information that is used later in the form. You create the literal constant and name it. You can “hard code” (enter) a value that is loaded into the element that uses the constant. However, to *use* a constant (the information that is stored in it), you need to use a standard rule, as explained below. Constants are typically used to define qualifying relationships.

---

**Qualifying Relationships** A qualifying relationship establishes a relationship between an element and its qualifier. A qualifier contains a code that further defines the element. Qualifying relationships are defined using an standard rule.

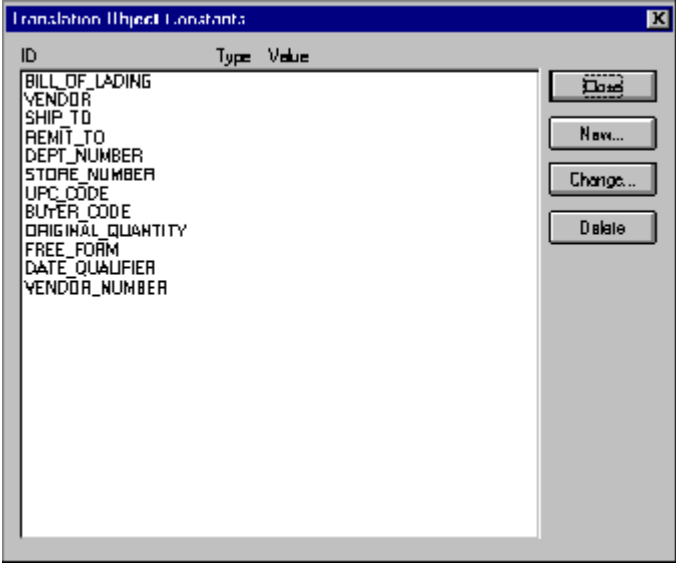
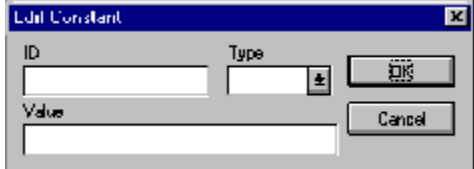
---

**Function Overview** You can perform the following functions with literal constants and qualifying relationships:

- Defining Literal Constants
  - Editing Literal Constants
  - Deleting Literal Constants
  - Mapping Literal Constants
  - Generating Qualifiers
-

## Defining Literal Constants

**Procedure** Use this procedure to create a literal constant so you can use it to store information.

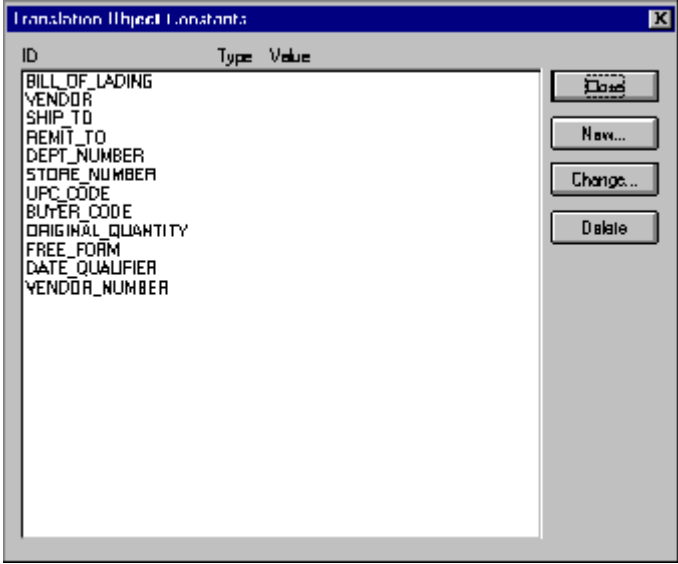
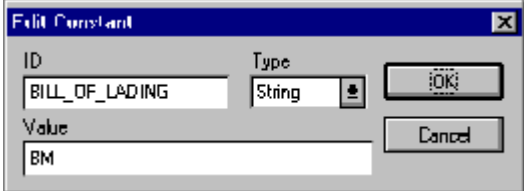
Step	Action
1	<p>Select Constants from the <b>Edit</b> menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Map Constants dialog box.</p> 
2	<p>Click <b>New</b>.</p> <p><b>System Response</b> The system displays the Edit Constant dialog box.</p> 
3	<p>In the ID box, type the literal constant identifier. This is typically a description of the element in which the constant is used. If you need to refer to the constant in an extended rule, you should use the data from this box.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
4	From the Type list, select the category of this literal constant. <ul style="list-style-type: none"><li>▶ <b>Integer</b> (select for numeric constants that are a positive or negative natural (non-fraction) number or “0” (zero))</li><li>▶ <b>Real</b> (select for numeric constants that are a positive or negative integer with an explicit decimal point)</li><li>▶ <b>String</b> (select for alphanumeric constants)</li></ul>
5	In the Value box, type the actual constant expression. This is the value of the literal constant.
6	Click <b>OK</b> to add the constant to the system.
7	Click <b>Close</b> to exit the Map Constants dialog box.

---

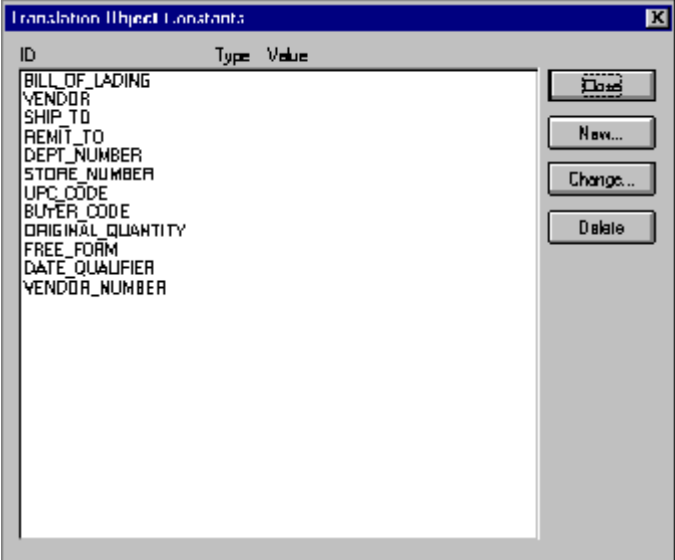
## Editing Literal Constants

**Procedure** Use this procedure to edit a literal constant.

Step	Action
1	<p>Select Constants from the <b>Edit</b> menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Map Constants dialog box.</p> 
2	<p>Highlight the constant you want to modify, and click <b>Change</b>.</p> <p><b>System Response</b> The system displays the Edit Constant dialog box.</p> 
3	<p>Make the changes to the constant, and click <b>OK</b> to accept the changes.</p>
4	<p>Click <b>Close</b> to exit the Map Constants dialog box.</p>

## Deleting Literal Constants

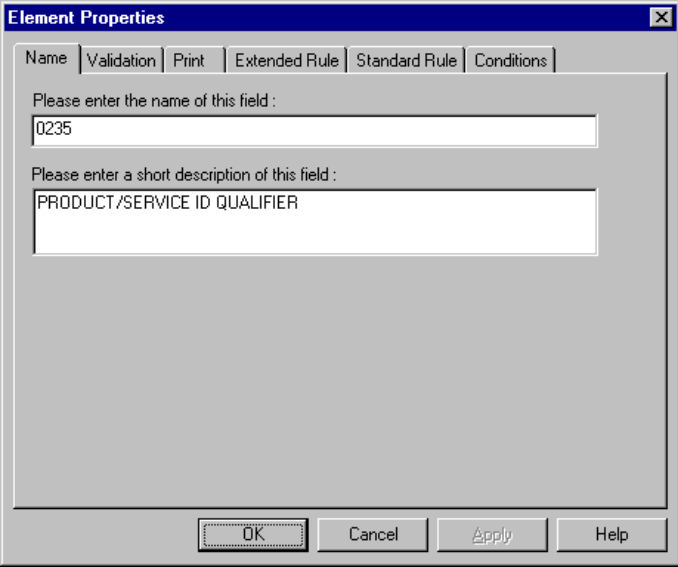
**Procedure** Use this procedure to delete a literal constant.

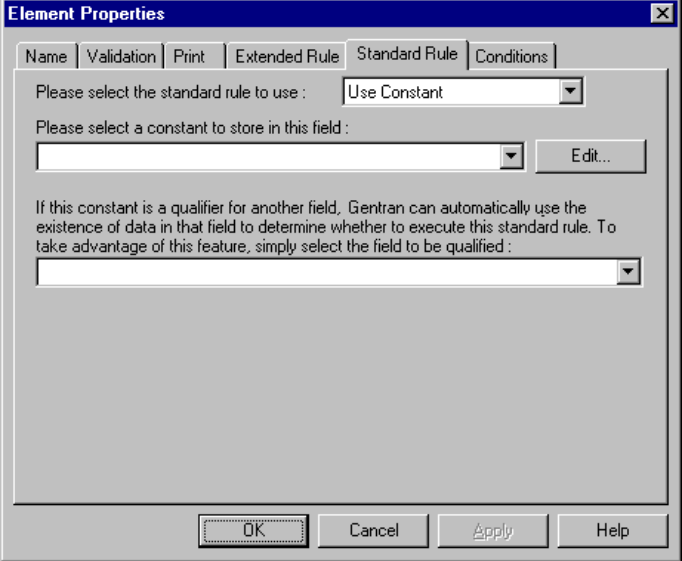
Step	Action
1	<p>Select Constants from the <b>Edit</b> menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Map Constants dialog box.</p> 
2	Highlight the constant you want to delete.
3	<p>Click <b>Delete</b> to remove the constant.</p> <p><b>WARNING</b> <b>THE CONSTANT IS DELETED WITHOUT WARNING.</b></p>
4	Click <b>Close</b> to exit the Map Constants dialog box.



## Using Literal Constants

**Procedure** Use this procedure to use a constant in which you previously stored information using an extended rule.

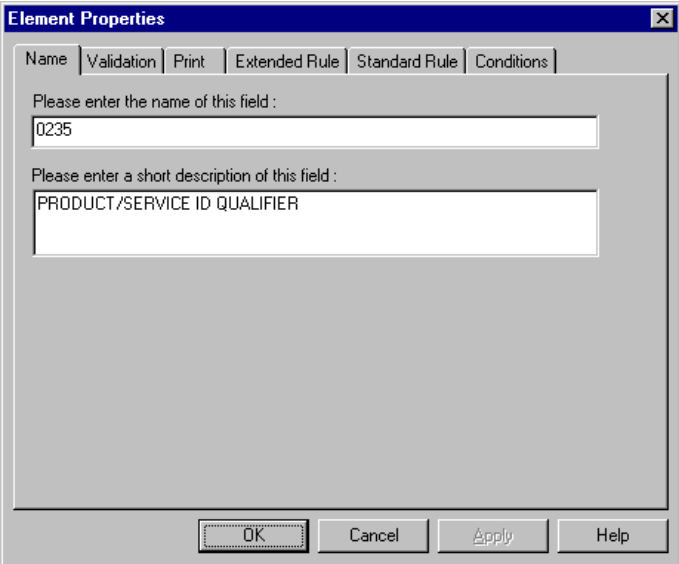
Step	Action
1	<p>Double-click the element in which you want to use the constant.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

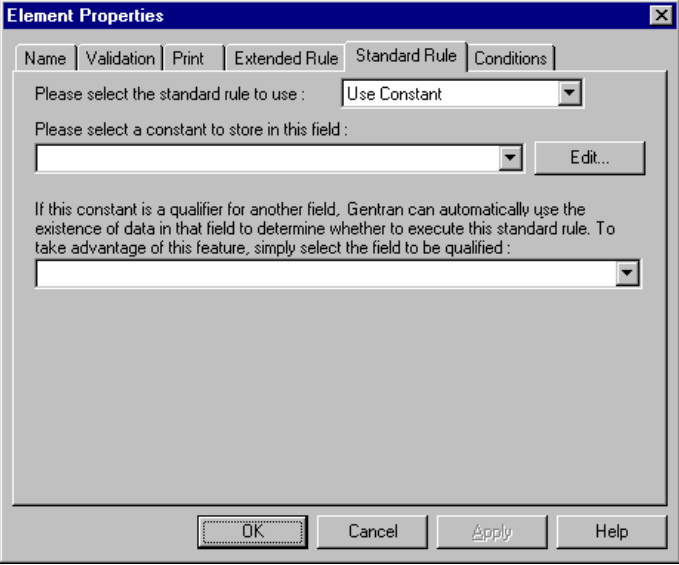
(Contd) Step	Action
3	<p>From the standard rule list, select <b>Use Constant</b>.</p> 
4	From the constant list, select the constant that you want to use.
5	Click <b>OK</b> and the data that was stored in the selected constant is loaded in the element.

# Generating Qualifiers

**Introduction** A qualifier is an element that has a value expressed as a code that gives a specific meaning to the function of another element. A qualifying relationship is the interaction between an element and its qualifier. The function of the element changes depending on which code the qualifier contains.

**Procedure** Use this procedure to define qualifying relationships.

Step	Action
1	<p>Double-click the element that you want to use to further define (qualify) another element.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, select <b>Use Constant</b>.</p> 
4	<p>From the qualifies list, select the element that this element qualifies.</p> <p><b>Note</b> This list contains only the other <i>active</i> elements in the same record or segment as the qualifying element.</p>
5	<p>Click <b>OK</b> and the qualifying relationship between the two elements is established.</p> <p><b>Note</b> When you set a constant value for a qualifier field, you know that you never need to override the value in the field. In this instance, you would probably hide the field, as well.</p> <p><b>Reference</b> See <i>Hiding Fields</i> on page 7 - 3 for more information.</p>

# Using the Use Accumulator Standard Rule

## Overview

### Introduction

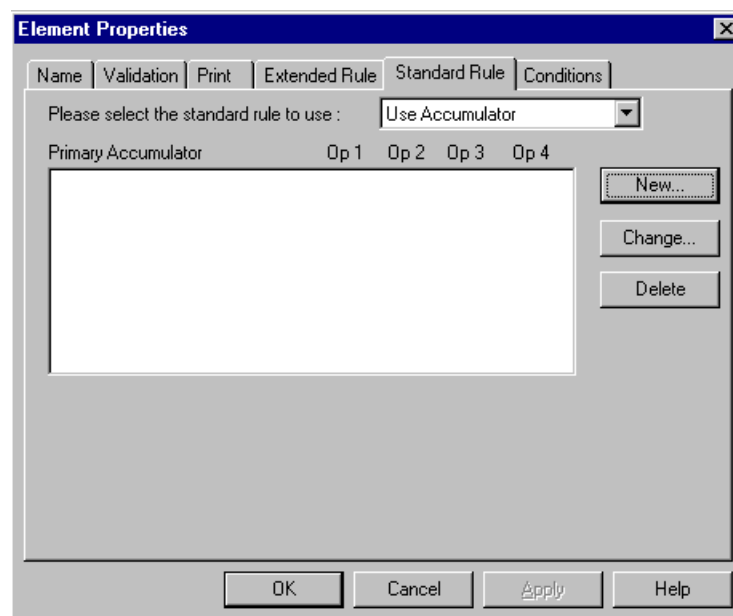
The Use Accumulator function gives you access to a set of numeric variables that you can manipulate via numeric operations, and then transfer to and from elements. This function enables you to add, change, or delete calculations for the element, including hash totals (used to accumulate numeric element values, i.e., quantity, price, etc.). This function also enables you to load the accumulated total into a control total element, and use accumulators. Accumulators are used generally for counting the occurrences of a specific element or generating increasing or sequential segment or line item numbers.

### Notes

- Accumulators are global variables.
- Accumulators are set to zero prior to being used in calculations.
- The order in which accumulator operations are performed depends on the hierarchical order of the form components.

### Standard Rule tab (Use Accumulator)

This illustration describes the Use Accumulator function.



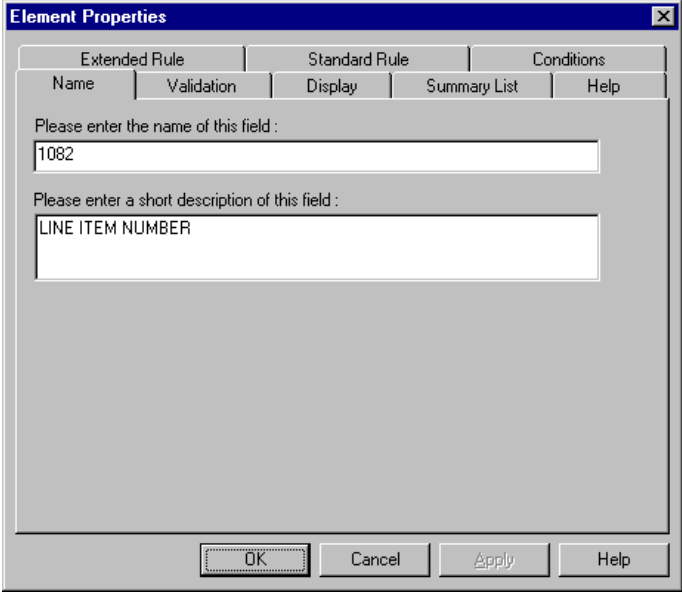
## Parts and Functions

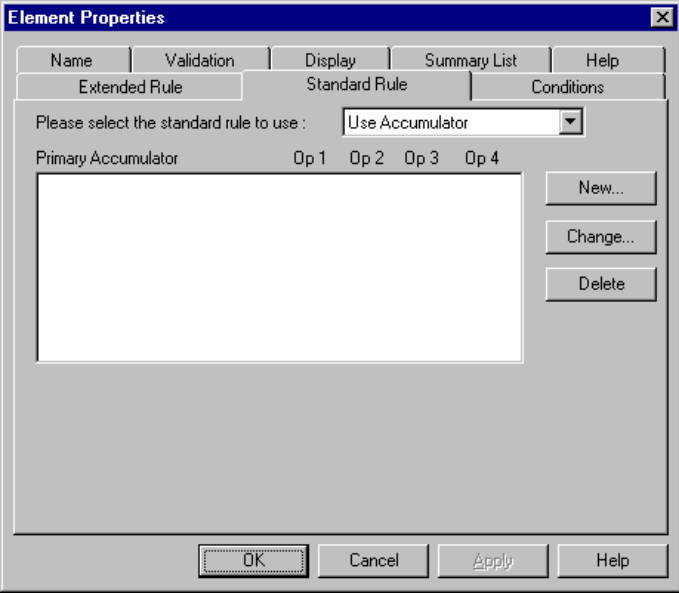
This table describes the parts and functions of the Use Accumulator Standard Rule.

<b>Part</b>	<b>Function</b>
Please select a standard rule to use list	Identifies standard rule options.
Primary Accumulator Op1, Op 2, Op 3, Op 4 list	Lists existing accumulators and their associated operations that were created for this element.
<b>New</b>	Accesses the Edit Accumulator Entry dialog box; Enables you to define an accumulator.
<b>Change</b>	Enables you to modify an existing accumulator.
<b>Delete</b>	Deletes a selected accumulator.  <b>Note</b> The selected accumulator is deleted without warning.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Apply</b>	Unavailable.
<b>Help</b>	Launches the Forms Integration online Help system.

## Using an Accumulator in a Form

**Procedure** Use this procedure to use the Use Accumulator function.

Step	Action
1	<p>Double-click an existing element, or create a new element.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, select <b>Use Accumulator</b>.</p> <p><b>System Response</b> The screen should look like the following.</p> 
4	<p>The Primary Accumulator list contains all existing calculations that were created for this element. Do you want to create a new calculation for this entry?</p> <ul style="list-style-type: none"> <li>▶ If yes, click <b>New</b> to access the Edit Accumulator Entry dialog box. Proceed to the next step.</li> </ul> <p><b>Reference</b> Please see <i>Using an Accumulator in a Form</i> on page 5 - 51 for more information.</p> <ul style="list-style-type: none"> <li>▶ If no, proceed to the next step.</li> </ul>
5	<p>Do you want to change an existing Primary Accumulator entry?</p> <ul style="list-style-type: none"> <li>▶ If yes, highlight a calculation in the Primary Accumulator list and click <b>Change</b> to access the Edit Accumulator Entry dialog box to edit the selected calculation. Proceed to the next step.</li> </ul> <p><b>Reference</b> Please see <i>Using an Accumulator in a Form</i> on page 5 - 51 for more information.</p> <ul style="list-style-type: none"> <li>▶ If no, proceed to the next step.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>



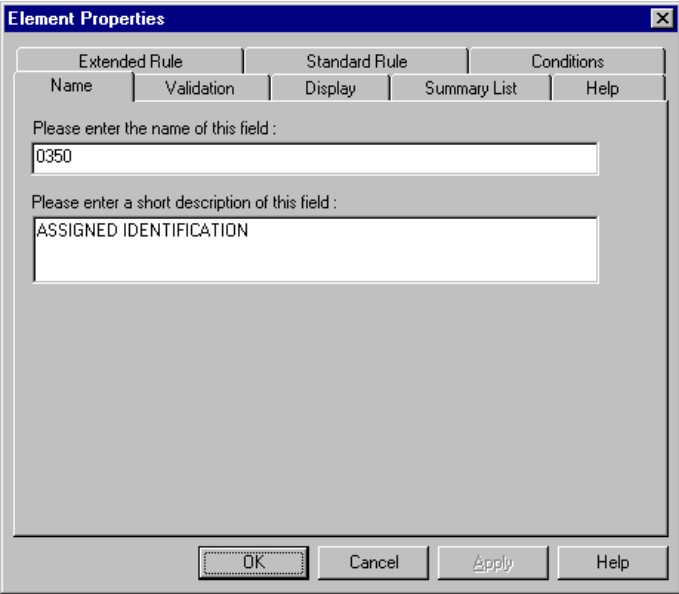
<b>(Contd) Step</b>	<b>Action</b>
6	<p>Do you want to delete an existing Primary Accumulator entry?</p> <ul style="list-style-type: none"><li>▶ If yes, highlight a calculation in the Primary Accumulator list and click <b>Delete</b>.</li></ul> <p><b>WARNING</b></p> <p><b>THE SELECTED CALCULATION IS DELETED WITHOUT WARNING.</b></p> <ul style="list-style-type: none"><li>▶ If no, proceed to the next step.</li></ul>
7	<p>Verify that your entries are accurate, and click <b>OK</b>.</p> <p><b>System Response</b> You exit the Element Properties dialog box.</p>

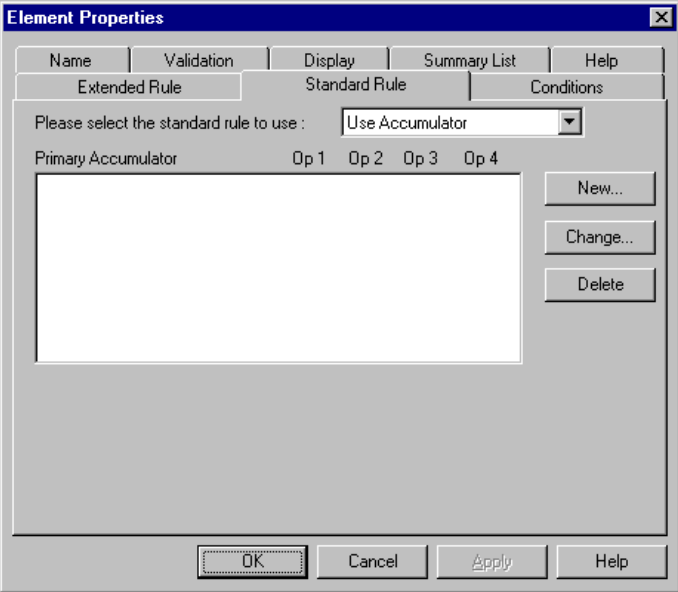
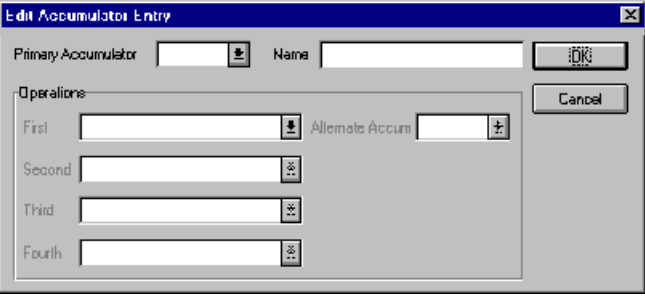
---

## Use Accumulator Example: Counting Line Items

**Introduction** In this example, you want an incremental count of the number of line items for a screen entry translation object, and you want to use that total value in the Number of Line Items Total element.

**Procedure** Use this procedure to count line items and generate a control total for an ANSI X12 purchase order.

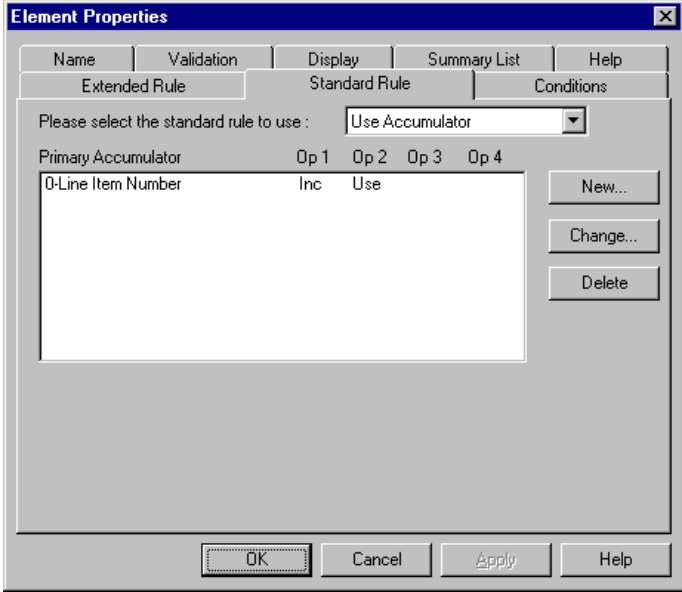
Step	Action
1	<p>Double-click the <b>P0101</b> element (in the P01 segment in the P01 group). This is the element that you typically use to count the line items.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right;">(Continued on next page)</p>

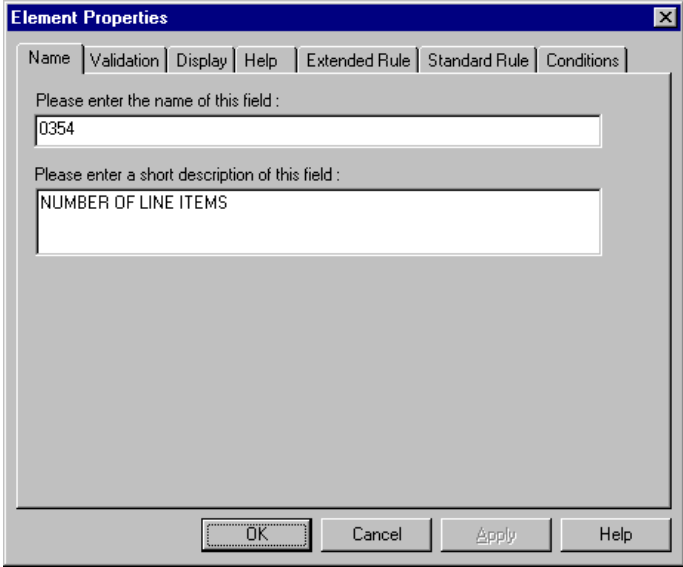
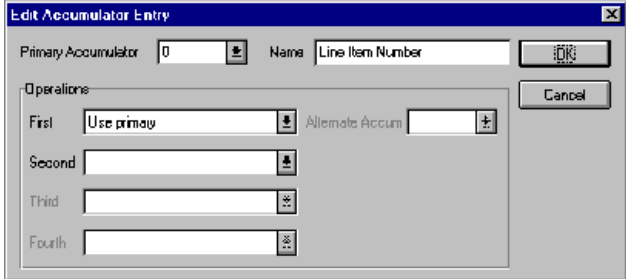
<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, select <b>Use Accumulator</b>.</p> 
4	<p>Click <b>New</b>.</p> <p><b>System Response</b> The system displays the Edit Accumulator Entry dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

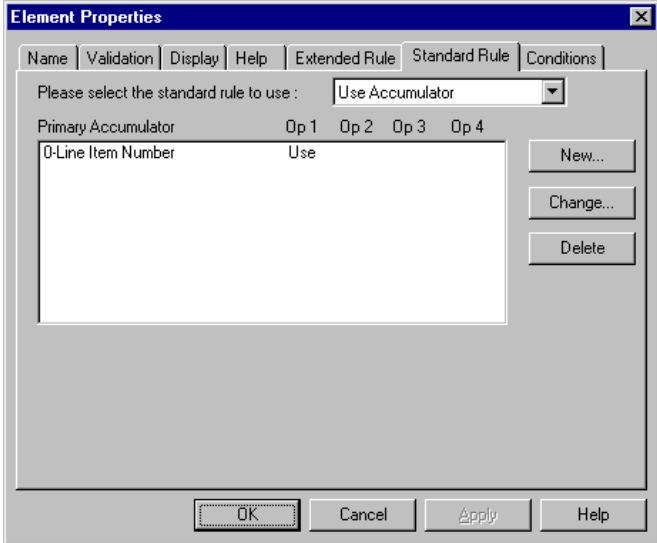


<b>(Contd) Step</b>	<b>Action</b>
5	<p>Select a Primary Accumulator from the list. Before any calculations are performed on an accumulator, its content is “0” (zero). When you use an accumulator, the system adds a new accumulator to the bottom of this list.</p> <p>For this example, select primary accumulator “0.”</p> <p><b>Note</b> There is <i>only one</i> set of accumulators for each form. This means that accumulator “0,” whether it is used in the Primary Accumulator or Alternate Accum box is the same accumulator with the same contents. If you assign calculations to accumulator “0” at the beginning of the form and then use accumulator “0” again later in the form, the content of that accumulator is the result of the earlier calculation. Any additional calculations you assign to that accumulator are performed on the contents resulting from an earlier calculation.</p>
6	<p>In the Name box, type <b>Line Item Number</b>. This is a descriptive alias that enables you to differentiate what the accumulators you create are used for.</p>
7	<p>From the First list, select <b>Increment Primary</b>. This is the first operation that the system performs. This specifies that the system will increment the PO101 element by one.</p> <p><b>Note</b> The First box is active only after you select a Primary Accumulator.</p>
8	<p>The valid selections for the First box are defined as follows:</p> <ul style="list-style-type: none"> <li>▶ <b>Increment primary</b> adds “1” (one) to the contents of the Primary Accumulator (i.e., Primary = Primary + 1).</li> <li>▶ <b>Decrement primary</b> subtracts “1” (one) from the contents of the Primary Accumulator (i.e., Primary = Primary - 1).</li> <li>▶ <b>Sum in primary</b> adds the numeric value (takes the positive or negative sign of the numbers into account) of the field to the contents of the Primary Accumulator (i.e., Primary = (+/-)Primary + (+/-)Element).</li> <li>▶ <b>Hash sum in primary</b> adds the absolute value (does not take the positive or negative sign of the numbers into account) of the element to the contents of the Primary Accumulator (i.e., Primary + Element).</li> <li>▶ <b>Load primary</b> loads the contents of the element into the Primary Accumulator (i.e., Primary = Element).</li> <li>▶ <b>Use primary</b> loads the contents of the Primary Accumulator into the element (i.e., Element = Primary).</li> <li>▶ <b>Zero primary</b> sets the value of the Primary Accumulator to zero (i.e., Primary = 0).</li> </ul> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
8 (cont.)	<ul style="list-style-type: none"> <li>▶ <b>Multiply with primary</b> multiplies the element with the contents of the Primary Accumulator, and stores the result in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Primary} * \text{Element}</math>).</li> <li>▶ <b>Divide by primary</b> divides the element with the contents of the Primary Accumulator, and stores the result in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Element} / \text{Primary}</math>).</li> <li>▶ <b>Divide primary by field</b> divides the contents of the Primary Accumulator with the element, and stores the result in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Primary} / \text{Element}</math>).</li> <li>▶ <b>Modulo with primary</b> divides the contents of the element with the contents of the Primary Accumulator, and stores the <i>remainder</i> of that operation in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Primary} \% \text{Element}</math>).</li> <li>▶ <b>Modulo with field</b> divides the contents of the Primary Accumulator with the contents of the element, and stores the <i>remainder</i> of that operation in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Element} \% \text{Primary}</math>).</li> <li>▶ <b>Negate primary</b> makes the contents of the Primary Accumulator negative (i.e., <math>\text{Primary} = \text{Primary} * -1</math>).</li> </ul> <p><b>Note</b> The only way to subtract the Primary Accumulator from the element is to “Negate” the Primary Accumulator and then use the “Sum in primary” operation to add the negative Primary Accumulator to the element.</p> <ul style="list-style-type: none"> <li>▶ <b>Move primary to alternate</b> copies the contents of the Primary Accumulator to the Alternate Accum. This overwrites the current contents of the Alternate Accum field (i.e., <math>\text{Alternate} = \text{Primary}</math>).</li> <li>▶ <b>Add primary to alternate</b> adds the contents of the Primary Accumulator to the contents of the Alternate Accum, and stores the result in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Primary} + \text{Alternate}</math>).</li> <li>▶ <b>Multiply primary by alternate</b> multiplies the contents of the Primary Accumulator with the contents of the Alternate Accum, and stores the result in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Primary} * \text{Alternate}</math>).</li> <li>▶ Accumulator (i.e., <math>\text{Primary} = \text{Primary} \% \text{Alternate}</math>).</li> <li>▶ <b>Divide primary by alternate</b> divides the contents of the Primary Accumulator with the contents of the Alternate Accum, and stores the result in the Primary Accumulator (i.e., <math>\text{Primary} = \text{Primary} / \text{Alternate}</math>).</li> <li>▶ <b>Modulo primary with alternate</b> divides the contents of the Primary Accumulator with the contents of the Alternate Accum, and stores the <i>remainder</i> of that operation in the Primary A</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
9	<p>From the Second list, select <b>Use primary</b>. This is the second operation that the system performs, after the First operation is completed. This specifies that the system loads the current value of the accumulator into the P0101 (Assigned Identification) element.</p> <p>The values for the Second, Third, and Fourth boxes are listed above.</p> <p><b>Note</b> The Second box is active only after you select a First operation that does not involve the Alternate Accum. The Third box is active only after you select a Second operation. The Fourth box is active only after you select a Third operation.</p>
10	<p>Click <b>OK</b> to add the accumulator.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like this:</p> 
11	<p>Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the P0101 element.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
12	<p>Double-click the <b>CTT01</b> element (in the CTT segment). This is the element that typically contains the total number of line items.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
13	Click the <b>Standard Rule</b> tab to access standard rule options.
14	From the standard rule list, select <b>Use Accumulator</b> .
15	Click <b>New</b> to access the Edit Accumulator Entry dialog box to create a new calculation for this element.
16	From the Primary Accumulator list, select primary accumulator <b>0</b> . This accumulator currently contains the total number of line items.
17	<p>From the First list, select <b>Use primary</b> to specify that the system loads the current value of the accumulator into the CTT01 (Number of Line Items Total) element.</p> <p>The Edit Accumulator Entry dialog box should now look like this:</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

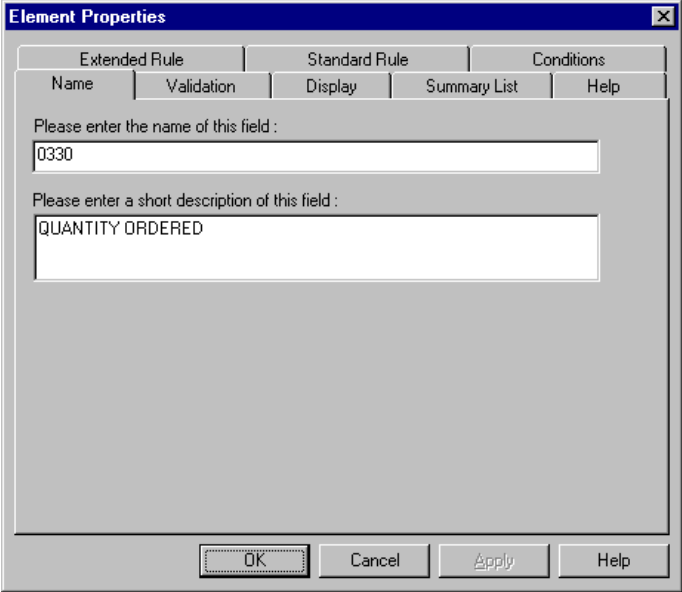
<b>(Contd) Step</b>	<b>Action</b>
18	<p>Click <b>OK</b> to add the accumulator.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like this:</p> 
19	<p>Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the CTT01 element.</p> <p>The CTT01 element now contains the total number of line items in the purchase order.</p> <p><b>Note</b> If you are adding this accumulator to a screen entry translation object, you would typically set the CTT01 field to display-only.</p>

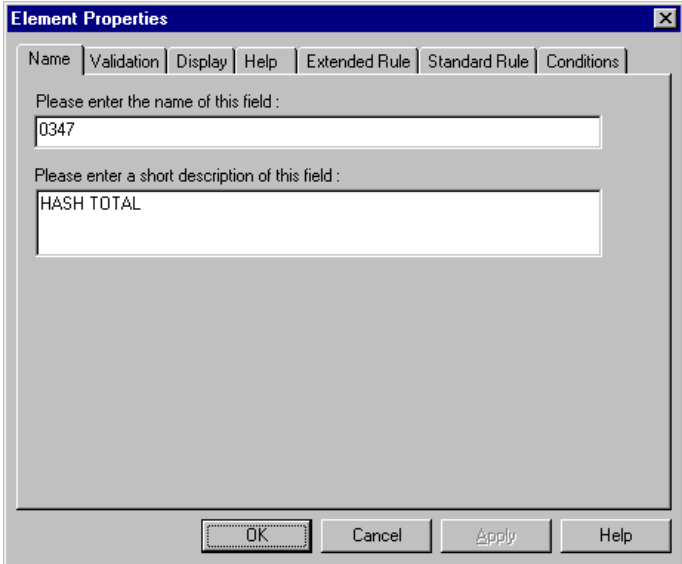


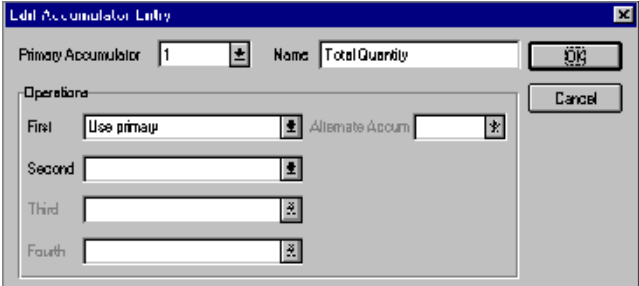
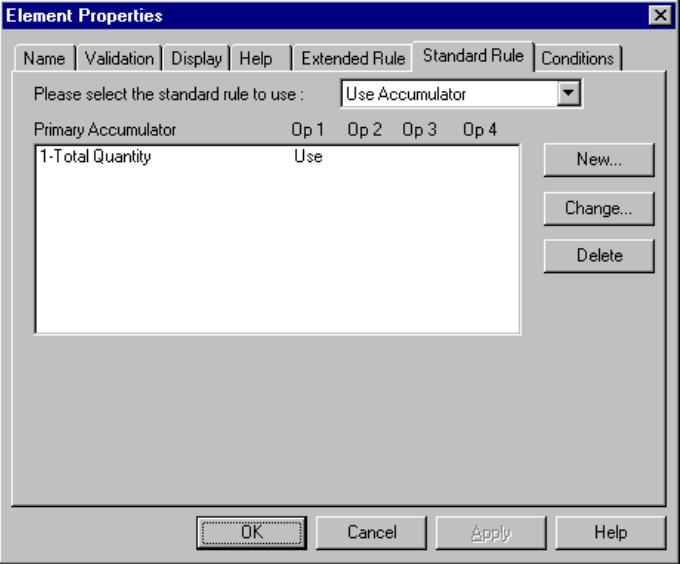
## Use Accumulator Example: Calculating Hash Totals

**Introduction** In this example, you want to count the quantity ordered for each line item and load the total quantity in the CTT02 (Hash Total) element for a screen entry translation object.

**Procedure** Use this procedure to calculate a hash total for an ANSI X12 purchase order.

Step	Action
1	<p>Double-click the <b>P0102</b> element (in the P01 segment in the P01 group). This is the element that you typically use to count the line items.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, select <b>Use Accumulator</b> .
4	Click <b>New</b> to access the Edit Accumulator Entry dialog box to create a new calculation for this element.
5	<p>Select a Primary Accumulator from the list. Before any calculations are performed on an accumulator, its content is “0” (zero). When you use an accumulator, the system adds a new accumulator to the bottom of this list.</p> <p>For this example, select primary accumulator “1.”</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
6	In the Name box, type <b>Total Quantity</b> . This is a descriptive alias that enables you to differentiate what the accumulators you create are used for.
7	From the First list, select <b>Hash Sum in Primary</b> .  <b>Note</b> This is the first operation that the system performs. This specifies that the system will add the numeric value of the PO102 element to the contents of the Primary Accumulator.
8	Click <b>OK</b> to add the accumulator.
9	Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the P0102 element.
10	Double-click the <b>CTT02</b> element (in the CTT segment). This is the element that typically contains the total quantity of the purchase order.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).  
11	Click the <b>Standard Rule</b> tab to access standard rule options.
12	From the standard rule list, select <b>Use Accumulator</b> .
13	Click <b>New</b> to access the Edit Accumulator Entry dialog box to create a new calculation for this element.
14	From the Primary Accumulator list, select primary accumulator <b>1</b> . This accumulator currently contains the total quantity.  (Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
<p>15</p>	<p>From the First list, select <b>Use primary</b>. The system loads the current value of the accumulator into the CTT02 (Hash Total) element.</p> <p><b>System Response</b> The Edit Accumulator Entry dialog should now look like this:</p> 
<p>16</p>	<p>Click <b>OK</b> to add the accumulator.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like this:</p> 
<p>17</p>	<p>Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the CTT02 element. The CTT02 element now contains the total quantity of the purchase order.</p>

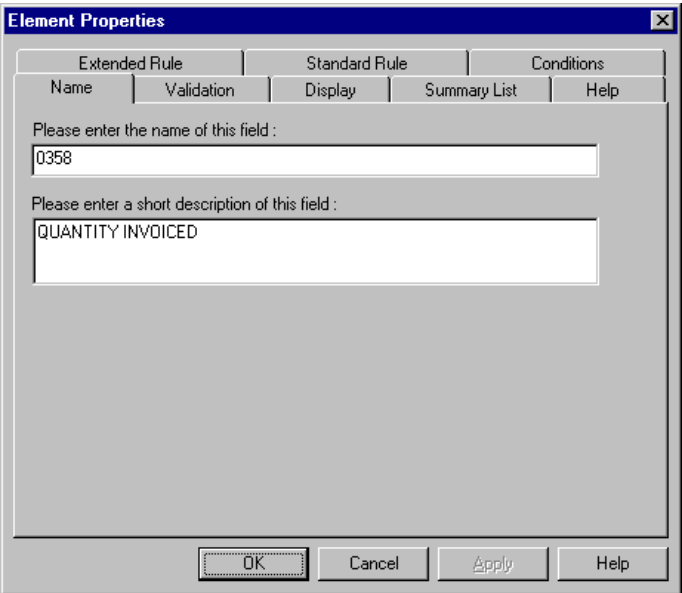
## Use Accumulator Example: Resetting and Calculating a Value Total

### Introduction

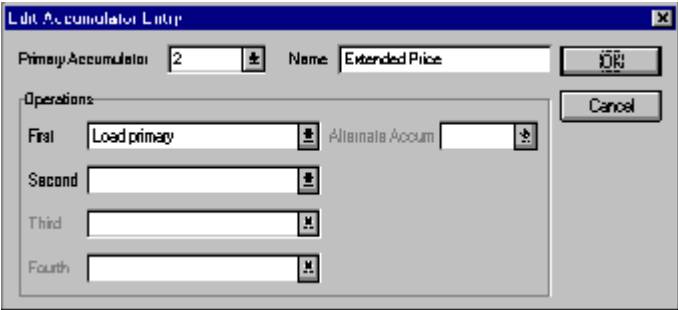
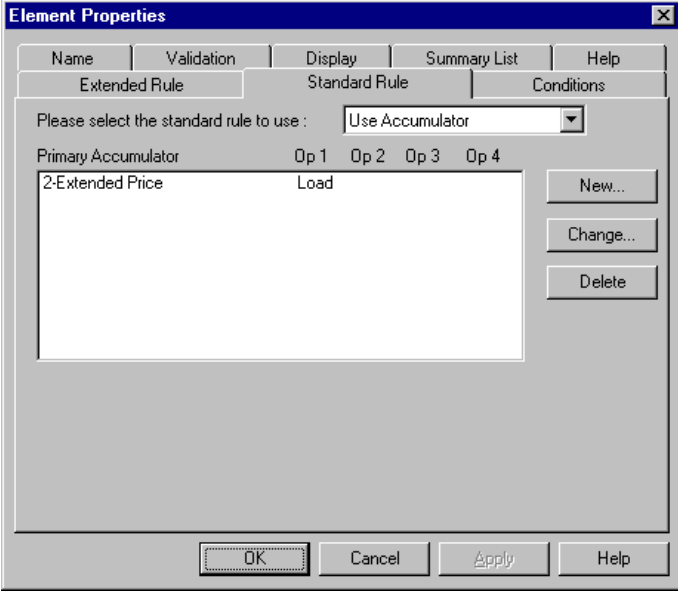
In this example, you want to multiply the quantity invoiced for each line item by the unit price to obtain the extended price for a screen entry translation object. Then, you want to generate a running total of extended price and load the final total into the TDS01 (Total Invoice Amount) element.

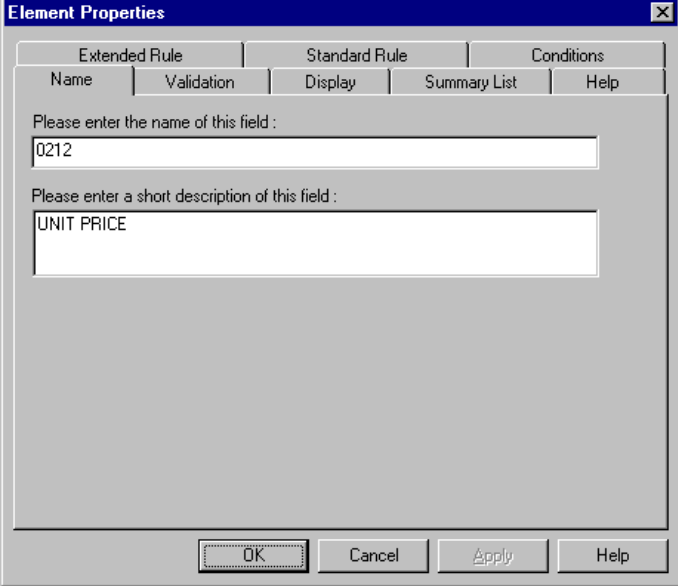
### How to multiply quantity invoiced by unit price

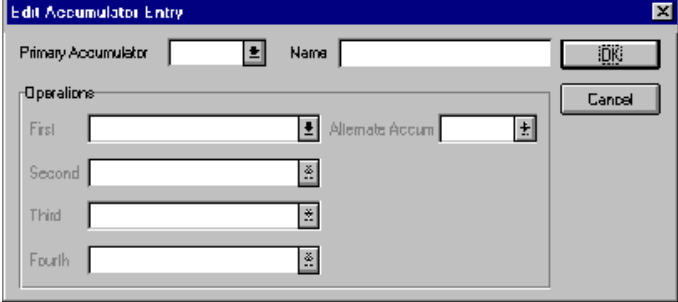
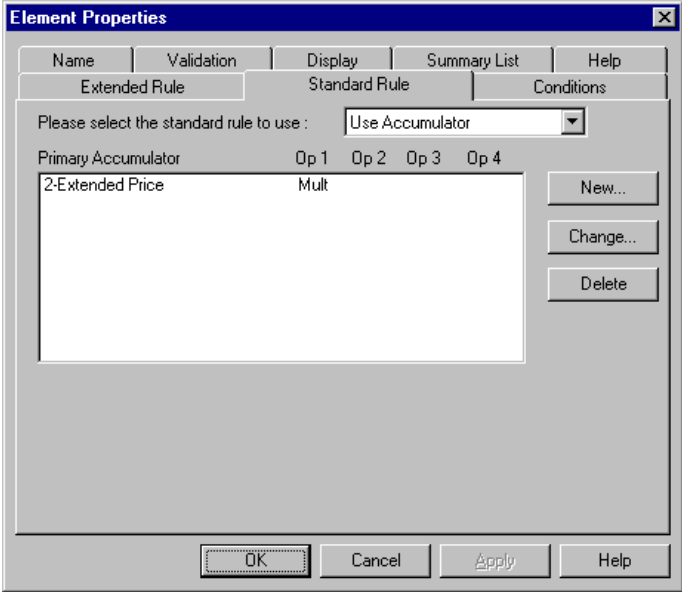
Use this procedure to multiply the quantity invoiced for each line item by the unit price to obtain the extended price for an ANSI X12 invoice.

Step	Action
1	<p>Double-click the <b>IT102</b> element (in the IT1 segment in the IT1 group). This is the element that you typically use to count the quantity invoiced.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, select <b>Use Accumulator</b> .
4	Click <b>New</b> to access the Edit Accumulator Entry dialog box to create a new calculation for this element.
5	Select primary accumulator <b>2</b> .

(Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
6	In the Name box, type <b>Extended Price</b> . This is a descriptive alias that enables you to differentiate what the accumulators you create are used for.
7	<p>From the First list, select <b>Load primary</b>.The system loads the contents of the element into the Primary Accumulator for each iteration of the IT1 group.</p> <p><b>System Response</b> The Edit Accumulator Entry dialog box should now look like this:</p> 
8	<p>Click <b>OK</b> to add the accumulator.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like this:</p> 
9	<p>Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the IT102 element.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

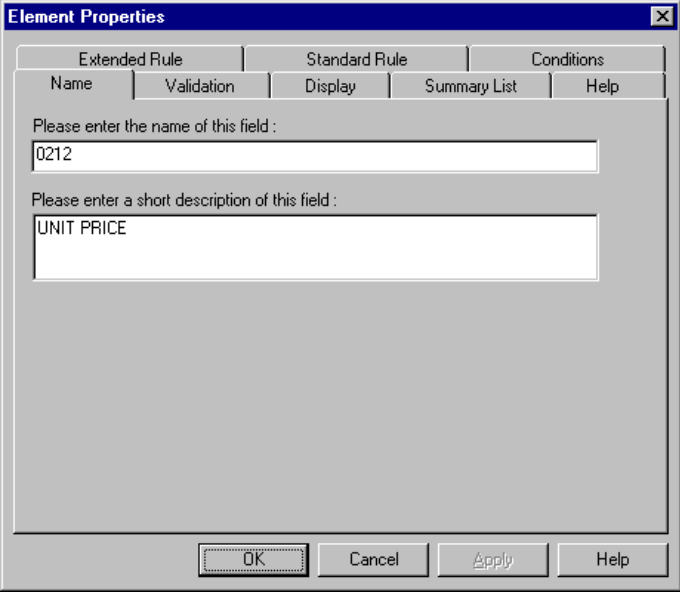
<b>(Contd) Step</b>	<b>Action</b>
10	<p>Double-click the <b>IT104</b> element (in the IT1 segment in the IT1 group). This is the element that contains the unit price for each line item.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
11	Click the <b>Standard Rule</b> tab to access standard rule options.
12	From the standard rule list, select <b>Use Accumulator</b> .
13	Click <b>New</b> to access the Edit Accumulator Entry dialog box to create a new calculation for this element.
14	(Continued on next page)Select primary accumulator 2.

<b>(Contd) Step</b>	<b>Action</b>
15	<p>From the First list, select <b>Multiply with primary</b>. The system multiplies the value of the IT104 (Unit Price) element with the contents of the primary accumulator, and stores the result in the primary accumulator for each iteration of the IT1 group.</p> <p><b>System Response</b> The Edit Accumulator Entry dialog box should now look like this:</p> 
16	<p>Click <b>OK</b> to add the accumulator.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like this:</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

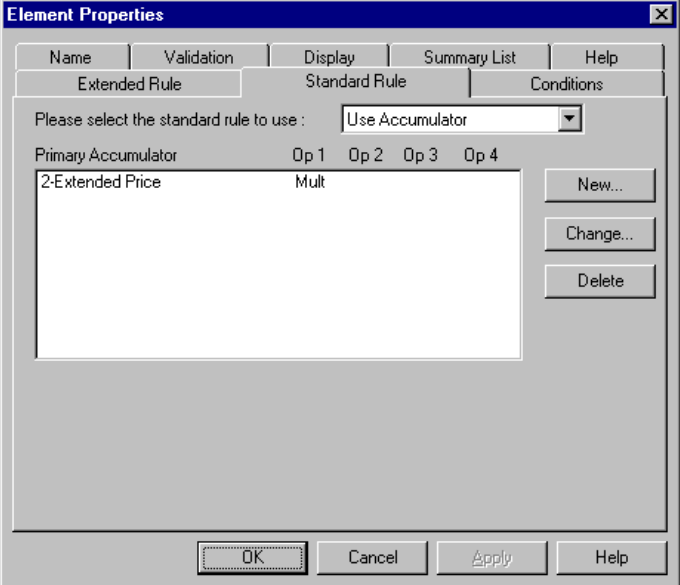
<b>(Contd) Step</b>	<b>Action</b>
17	<p>Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the IT104 element.</p> <p><b>Note</b> If there is an extended price element in your EDI file, you could load the total from the extended price calculation into that element. To do this, you need to use an accumulator on that extended price element that specifies “Use primary” for accumulator 2.</p>

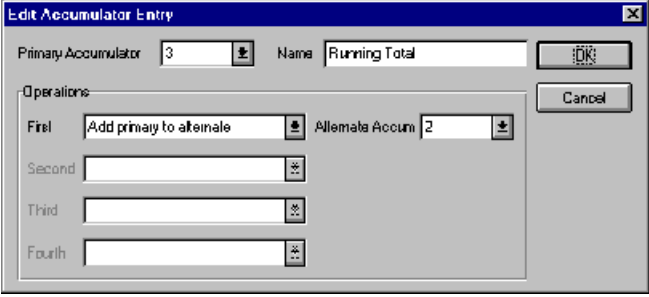
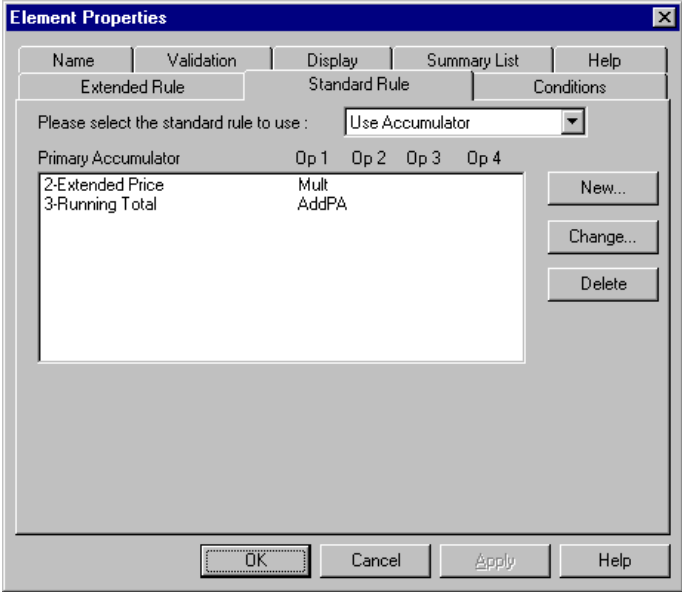
### How to generate a running total of extended price

Use this procedure to generate a running total of the extended price.

<b>Step</b>	<b>Action</b>
1	<p>Double-click the <b>IT104</b> element (in the IT1 segment in the IT1 group). This is the element that contains the unit price for each line item.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

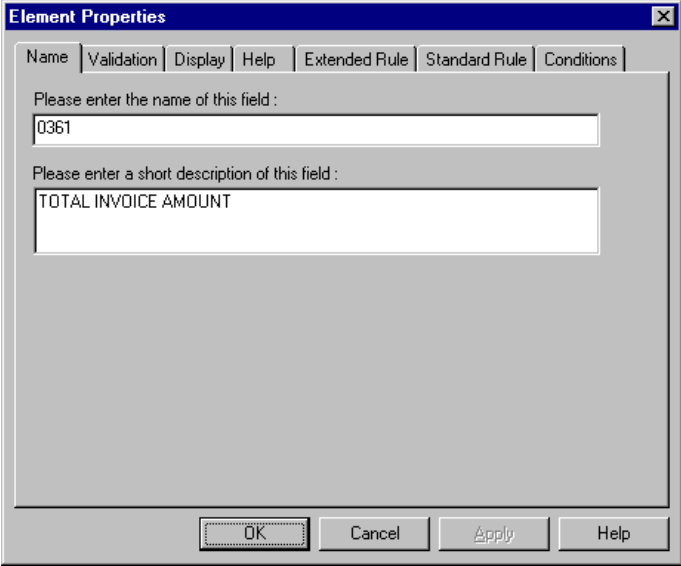


<b>(Contd) Step</b>	<b>Action</b>
2	<p>On the Standard Rule tab, you already established one accumulator that is displayed in the list.</p> 
3	Click <b>New</b> to access the Edit Accumulator Entry dialog box to create another calculation for this element.
4	Select primary accumulator <b>3</b> .
5	In the Name box, type <b>Running Total</b> .
6	<p>From the First list, select <b>Add primary to alternate</b>. The system adds the contents of the primary accumulator to the contents of the alternate accumulator, and stores the result in the primary accumulator for each iteration of the IT1 group.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

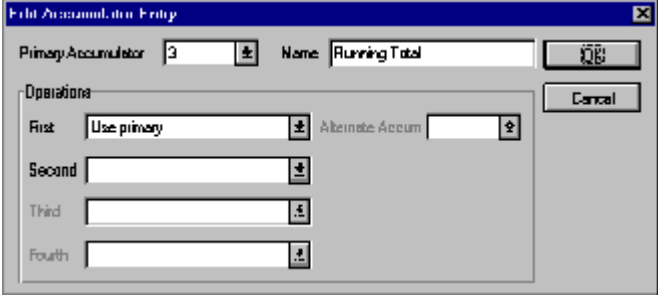
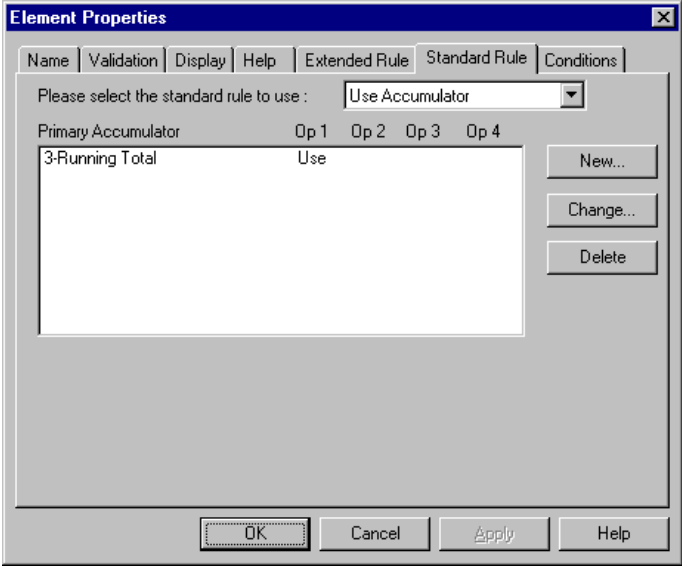
<b>(Contd) Step</b>	<b>Action</b>
7	<p>From the Alternate Accum list, select <b>2</b>. The system adds the value of accumulator “2” (which contains the extended price for a line item) to the value of accumulator “3.” The system stores the sum in accumulator “3,” which therefore contains a running total of the extended price with each iteration of the IT1 group.</p> <p><b>System Response</b> The Edit Accumulator Entry dialog box should now look like this:</p> 
8	<p>Click <b>OK</b> to add the accumulator.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like this:</p> 
9	<p>Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the IT104 element.</p>

### How to load a running total of extended price

Use this procedure to load the running total of the extended price into the TDS01 (Total Invoice Amount) element.

Step	Action
1	<p>Double-click the <b>TDS01</b> element. This is the element that contains the total invoice amount for each line item.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, select <b>Use Accumulator</b> .
4	Click <b>New</b> to access the Edit Accumulator Entry dialog box to create a new calculation for this element.
5	Select primary accumulator <b>3</b> .

(Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
6	<p>From the First list, select <b>Use primary</b>. The system loads the contents of the primary accumulator into the TDS01 (Total Invoice Amount) element.</p> <p><b>System Response</b> The Edit Accumulator Entry dialog box should now look like this:</p> 
7	<p>Click <b>OK</b> to add the accumulator.</p> <p><b>System Response</b> The Standard Rule tab of the Element Properties dialog box should now look like this:</p> 
8	<p>Click <b>OK</b> on the Element Properties dialog box to add the standard rule to the TDS01 element.</p> <p><b>Note</b> If you are adding this accumulator to a screen entry translation object, you would typically set the TDS01 element to display-only.</p>

# Using the Loop Count Standard Rule

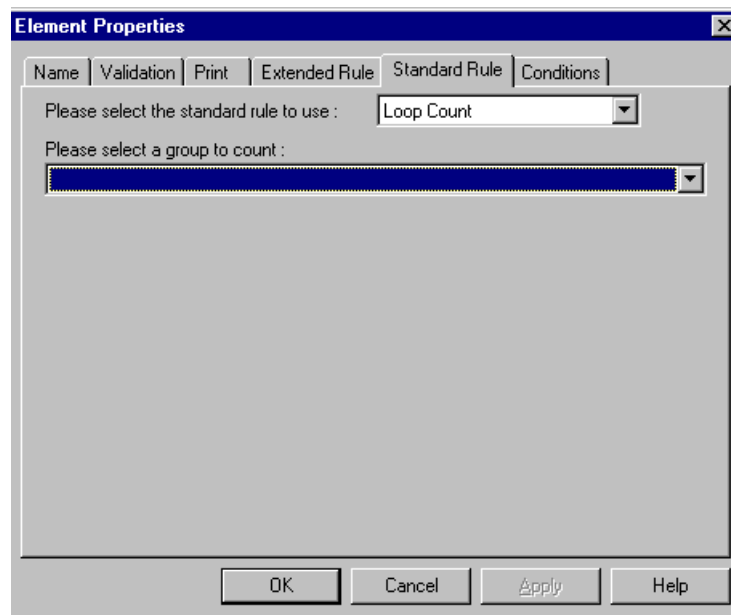
## Overview

### Introduction

The Loop Count function enables you to obtain the value of the current loop count, if the element is part of a loop. If the loop is a nested loop, you can track the current loop *or* the outer loop. For example, if the Y loop is nested within the X loop, and the Y loop has cycled through 15 iterations and the X loop has cycled through 3 iterations, you can choose to count either the “15” (Y loop) or the “3” (X loop).

### Standard Rule tab (Loop Count)

This illustration describes the Loop Count function.



### Parts and Functions

This table describes the parts and functions of the Loop Count Standard Rule.

Part	Function
Please select a standard rule to use list	Identifies standard rule options.
Please select a group to count	Identifies the group for which you want to obtain a loop count. If a loop is nested, you can track the current loop or the outer loop.
<b>OK</b>	Saves changes; exits the dialog box.

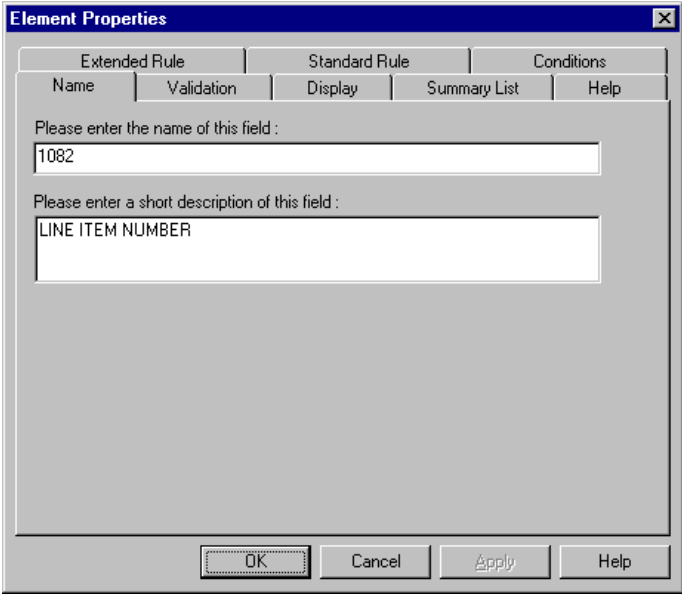
(Continued on next page)

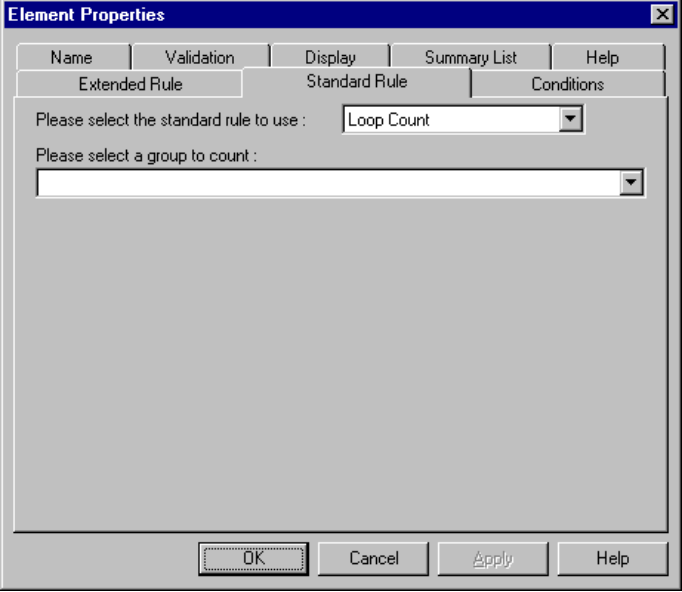
<b>(Contd) Part</b>	<b>Function</b>
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Apply</b>	Unavailable.
<b>Help</b>	Launches the Forms Integration online Help system.

---

## Using Loop Count in a Form

**Procedure** Use this procedure to use the Loop Count function.

Step	Action
1	<p>Double-click an existing element, or create a new element</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, select <b>Loop Count</b>.</p> 
4	<p>In the Please select a group to count text box, select the loop that you want to obtain the current value of the loop count.</p> <p><b>Note</b> If the loop is a nested loop, you can track the current loop <i>or</i> the outer loop.</p> <p><b>Example</b> If the Y loop is nested within the X loop, and the Y loop has cycled through 15 iterations and the X loop has cycled through 3 iterations, you can choose to count either the “15” (Y loop) or the “3” (X loop).</p>
5	<p>Verify that your entries are accurate, and click <b>OK</b>.</p> <p><b>System Response</b> You exit the Element Properties dialog box.</p>



# Using the Use Code Standard Rule

## Overview

---

**Introduction**

The Use Code function enables you to select an element from a predefined code table and specify whether or not a compliance error is generated if the element does not contain one of the values in the code table for screen entry translation objects. The Use Code function enables you to extract and print the description of a code for print translation objects.

---

**Creating code tables in Forms Integration**

The Forms Integration subsystem enables you to create code tables to be used with the current form. You can set up code tables in the Forms Integration subsystem to function like the partner cross-reference and lookup tables in Gentran:Server. However, code tables that are set up in the Forms Integration subsystem can be used *only* for the current form. Code tables that you create in Gentran:Server can be used globally for all forms.

---

**Using the Use Code function**

An element with a Use Code rule enables values to either be checked against or selected from the codes in a specified code table. Codes are typically used to further qualify another element.

**Example**

If the XX element contains address information, you can further qualify that element by choosing the code “SU” from the 0222 table. In the 0222 table, the code “SU” is described as a “supplier's address.” Therefore, by using this code with the XX element, you are indicating that the XX element is not just address information, but address information for the *supplier*.

---

**Related topic**

See *Defining a Code List* on page 5 - 88 for more information.

---

## Standard Rule tab (Use Code)

This illustration describes the Use Code function.

## Parts and Functions

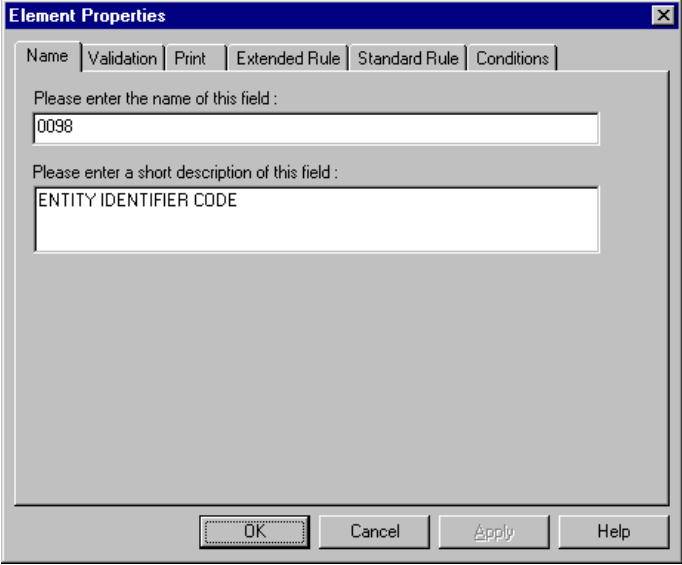
This table describes the parts and functions of the Use Code Standard Rule.

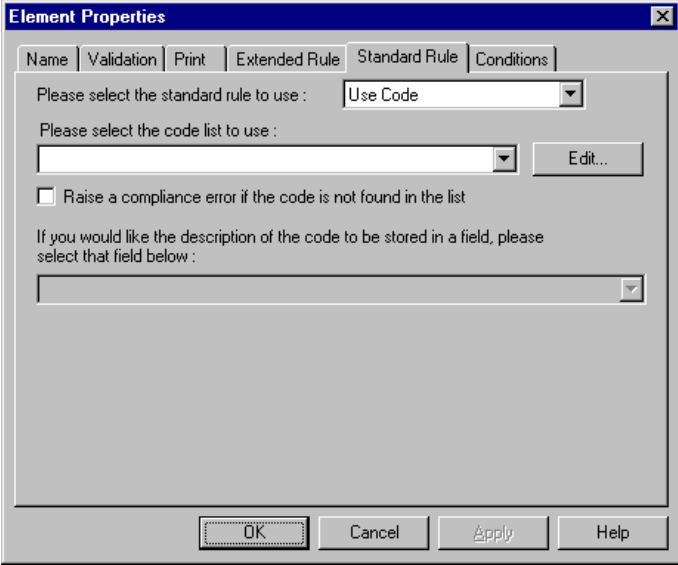
Part	Function
Please select a standard rule to use list	Identifies standard rule options.
Please select a code list to use list.	Displays available code lists.
<b>Edit</b>	Accesses the Edit Code List dialog box. This dialog box enables you to add, change, delete or load a code list.
Raise a compliance error if the code is not found in the list check box	Specifies whether you want the system to generate a compliance error if the element does not contain one of the codes from the specified table (nothing else is valid for that field).  (Continued on next page)

<b>(Contd) Part</b>	<b>Function</b>
If you would like a description of the code to be stored in a field, please select the field below list box	Displays elements where you want the description of the code that is used to appear when the selection is made.  <b>Example</b> If the code is SU, it much more useful to view the description of the code (suppliers address). If you selected element XX from the store description list, the description for the code use is mapped to element XX.
<b>OK</b>	Saves changes; exits the dialog box.
<b>Cancel</b>	Cancels changes; exits the dialog box.
<b>Apply</b>	Unavailable.
<b>Help</b>	Launches the Forms Integration online Help system.

## Using Use Code in a Form

**Procedure** Use this procedure to use the Use Code function.

Step	Action
1	<p>Double-click an existing element, or create a new element .</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Standard Rule</b> tab to access standard rule options.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

(Contd) Step	Action
3	<p>From the standard rule list, select <b>Use Code</b>.</p> 
4	<p>Is the code list that you want to use listed in the Please select the code list to use drop-down list?</p> <ul style="list-style-type: none"> <li>▶ If yes, select that code list and proceed to the next step.</li> <li>▶ If no, click <b>Edit</b> to access the Edit Code List dialog box. This enables you to load or create a code table. Proceed to the next step.</li> </ul>
5	<p>Must the element contain, for compliance reasons, one of the codes from the specified table (nothing else is valid for that field)?</p> <ul style="list-style-type: none"> <li>▶ If yes, select the compliance error check box. Proceed to the next step.</li> </ul> <p><b>Example</b> If an element is defined as containing only “YES” or “NO,” you can set up an exclusive code table that contains only YES and NO. Then if you receive a “MAYBE” in that element, the system flags it as an error.</p> <ul style="list-style-type: none"> <li>▶ If no, verify that the compliance error check box is <i>not</i> selected. Proceed to the next step.</li> </ul> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
6	<p data-bbox="630 342 1230 369">Do you want the code description to be stored in a field?</p> <ul data-bbox="630 388 1409 478" style="list-style-type: none"><li data-bbox="630 388 1409 478">▶ If yes, select an element from the drop-down list where you want the description of the code (that is used) to appear when the selection is made. Proceed to the next step.</li></ul> <p data-bbox="678 499 792 527"><b>Example</b></p> <p data-bbox="678 531 1409 646">If the code is "SU," it is much more useful to view the description of the code ("Supplier's Address). If you selected element "XX" from the store description list, the description for the code used is mapped to element "XX."</p> <ul data-bbox="630 667 1003 695" style="list-style-type: none"><li data-bbox="630 667 1003 695">▶ If no, proceed to the next step.</li></ul>

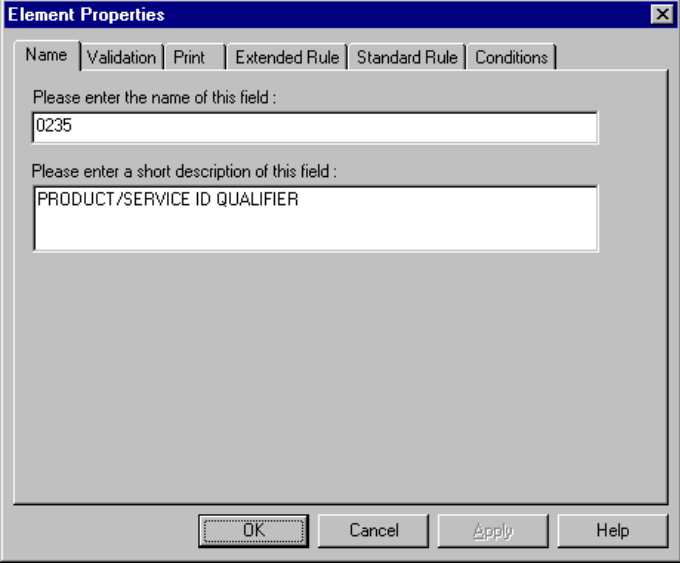
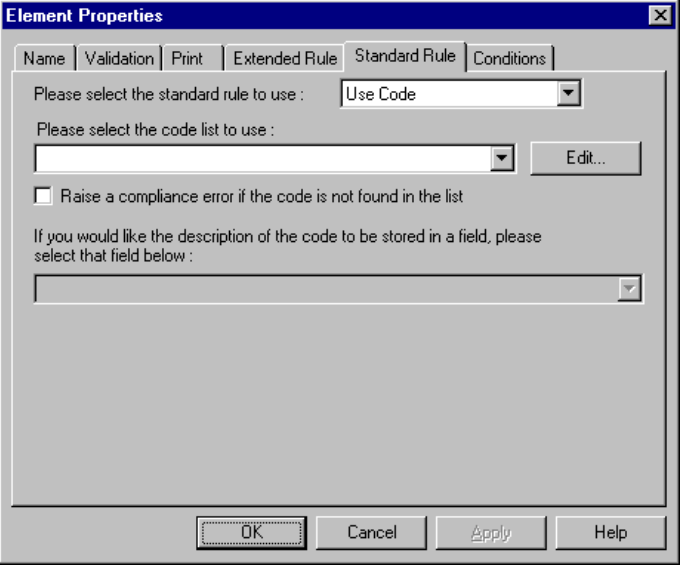
---

## Use Code Example: Selecting a Code List Item

**Introduction** For this example you need to have a list displayed during data entry, so the user can select your partner's customer product code. Therefore, you need to establish and use a code table with the Product Service ID *Qualifier*.

**Procedure** Use this procedure to select from a code list table.

Step	Action
1	<p>Load the code list table from the standard for the Product/Service ID Qualifier element.</p> <p><b>Reference</b> Please see <i>Loading a Code List Table from the Standard</i> on page 5 - 99 or more information.</p> <p><b>Note</b> Please note that the standards provide code list tables only for elements that use them. For example, in the TD4 segment, the TD401 (Special Handling Code) and TD403 (Hazardous Material Class Code) have code tables provided by the standard.</p>
2	<p>Define a code list table for the Product/Service ID Qualifier element that only contains the code value "BP" (Buyer's Part Number).</p> <p><b>Reference</b> Please see <i>Defining a Code List</i> on page 5 - 88 for more information.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>Double-click the element for which you need to validate data against a code table (Product/Service ID Qualifier).</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
4	Click the <b>Standard Rule</b> tab to access standard rule options.
5	<p>From the standard rule list, select <b>Use Code</b>.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>
6	From the code list, select the code list table that the data in this element is validated against (0235).
7	If you need to specify that, for compliance reasons, the element <i>must</i> contain one of the codes from the specified table (nothing else is valid for that element), select the compliance error check box.
8	Click <b>OK</b> to add this standard rule to the element.

---

# Using Code List Tables

## Overview

---

### Introduction

Code List Tables are used by EDI standards as repositories for lists of codes. Each EDI standard provides a code list for each element that can be further defined with a code. Gentran:Server allows you to load code lists from the standard. You can either load all the codes in the table, or you can select only one or more codes from the table.

---

### Use Code Standard Rule

Once you load or create a code table, you can use a “Use Code” standard rule. For screen entry translation objects, you can use it to select an element from a list that contains a predefined code table. You can also specify whether or not a compliance error is generated if the element does not contain one of the values in the code table. For print translation objects, the Use Code function enables you to extract and print the description of a code and validate the contents of a element against the values in the code table for print translation objects.

---

### Code list function

The Code List function also enables you to store a code's description in an element/field, create a unique code table, use code values from a code table, and flag whether or not the system will generate an error if a validation against the code table fails. You can import and export code lists and copy and paste code lists between forms.

An element with a Use Code standard rule applied to it allows values to either be checked against or selected from the codes in a specified code table. Codes are typically used to further qualify the element.

### Example

The Product/Service ID (0234) element contains a product code that you expect to be the *customer's* product code. If you receive a value in that element that is something other than the customer's product code, you want the system to generate an error. So, you first use the code table for the Product/Service ID Qualifier (0235), but specify when you load it that the code table should only include the code “BP.” In this code table, the code “BP” is described as a “Buyer’s Part Number.” Then, you make the 0235 code table exclusive for the Product/Service ID Qualifier (0235) element. Therefore, by using this code with the qualifier for the Product/Service ID element, you are indicating that the Product/Service ID element must contain not just a product code, but product code from the *customer*.

---

---

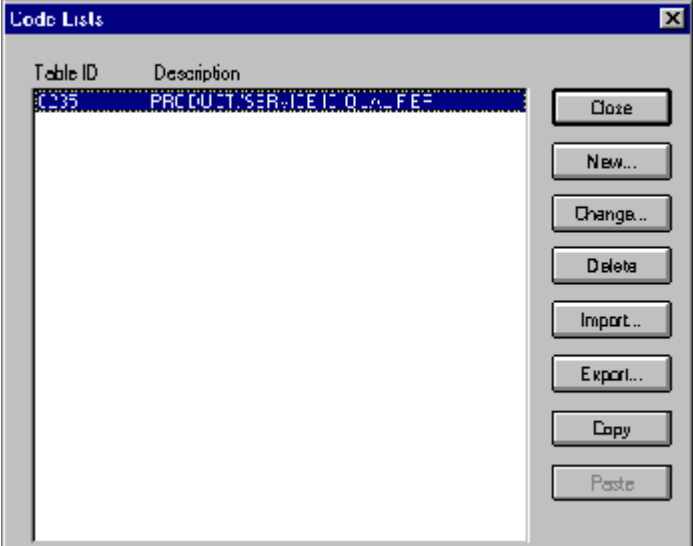
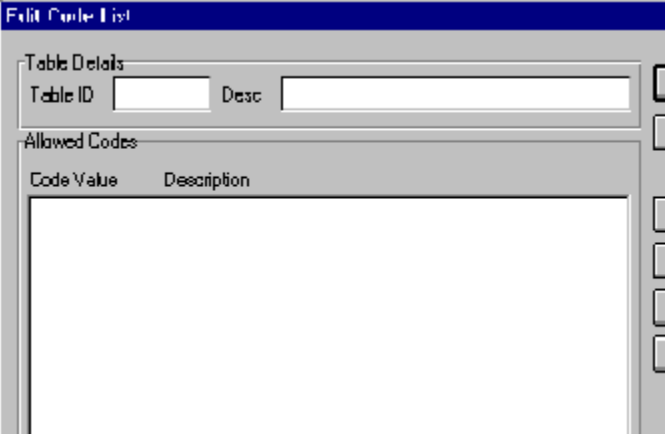
**Code List Table  
functions**


You can perform the following functions with Code List Tables:

- ▶ Defining a Code List
  - ▶ Modifying a Code List or Code List Entry
  - ▶ Deleting a Code List or Code List Entry
  - ▶ Importing a Code List
  - ▶ Exporting a Code List
  - ▶ Loading Code Tables from the Standard
  - ▶ Copying and Pasting Code Lists
  - ▶ Validating Data Against Code List Tables
  - ▶ Loading Code Item Descriptions
-

## Defining a Code List

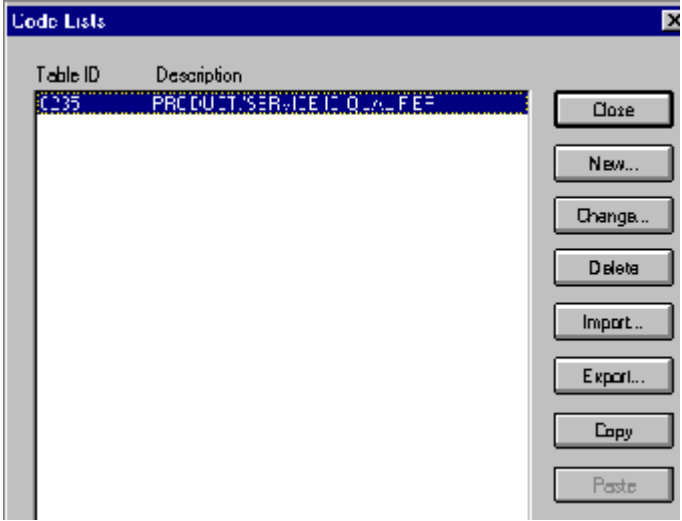
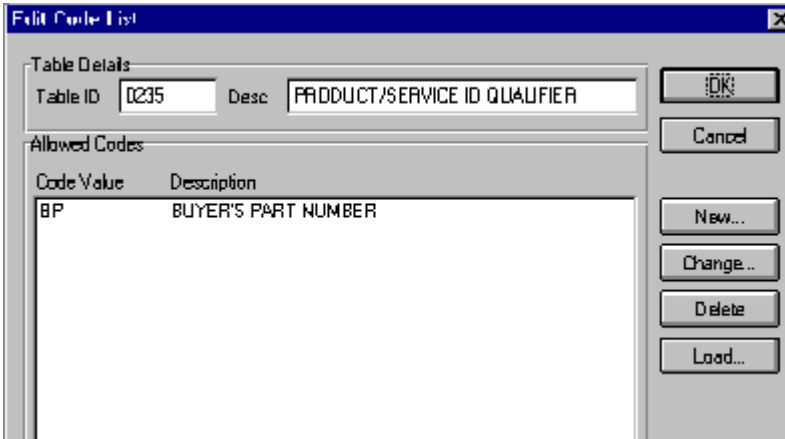
**Procedure** Use this procedure to define a code list table.

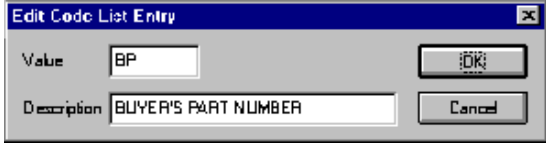
Step	Action
1	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
2	<p>Click <b>New</b> to access the Edit Code List dialog box.</p> 
3	<p>In the Table ID box, type the name of the element for which this code list table is used.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

(Contd) Step	Action
4	In the Description box, type the description of the element for which this code list table is used.
5	Click <b>New</b> to access the Edit Code List Entry dialog box. 
6	In the Value box, type the actual value of the code.
7	In the Description box, type a description of the code value. The code value description is used if you specify an element to which you want the code description mapped.
8	Click <b>OK</b> to save the code list entry.
9	Repeat <b>Step 5</b> through <b>Step 8</b> to add more code list entries to the code list table.
10	Click <b>Close</b> to save and exit the Edit Code List dialog box.
11	Click <b>Close</b> to exit the Code Lists dialog box.

## Modifying a Code List

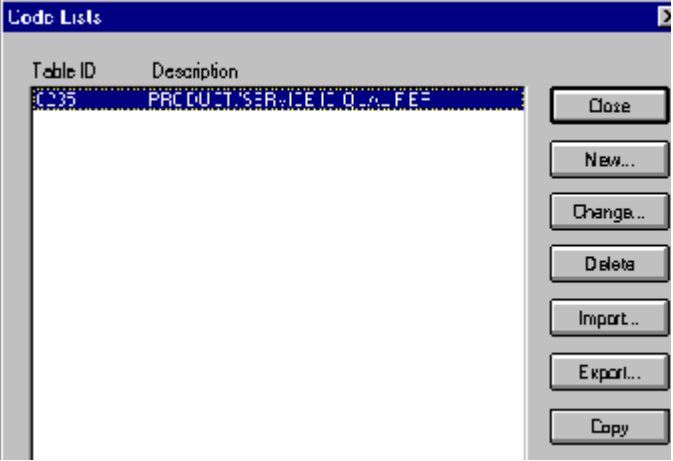
**Procedure** Use this procedure to modify a code list table or code list entry.

Step	Action
1	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
2	Highlight the code list that you want to modify.
3	<p>Click <b>Change</b> to access the Edit Code List dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

(Contd) Step	Action
4	<p>At this point, you can add new code list entries to the table, delete entries, or modify code list entries. To modify an entry, highlight it and click <b>Change</b> to access the Edit Code List Entry dialog box.</p> 
5	Make the necessary changes.
6	Click <b>OK</b> to save the code list entry.
7	Repeat <b>Step 2</b> through <b>Step 6</b> to modify other code list entries in the code list table.
8	Click <b>Close</b> to save and exit the Edit Code List dialog box.
9	Click <b>Close</b> to exit the Code Lists dialog box.

## Deleting a Code List or Code List Entry

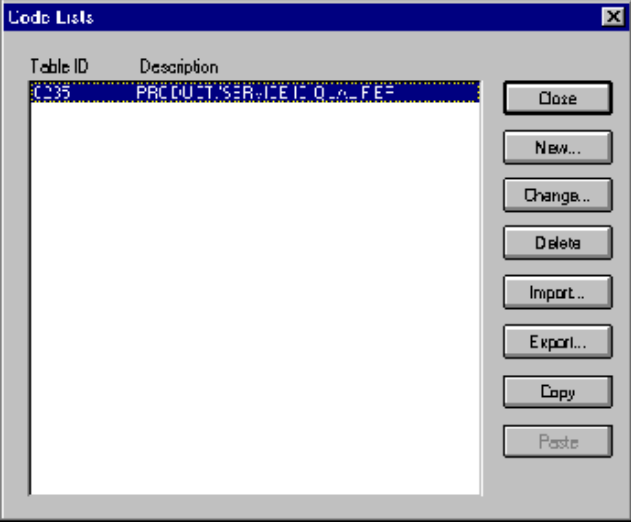
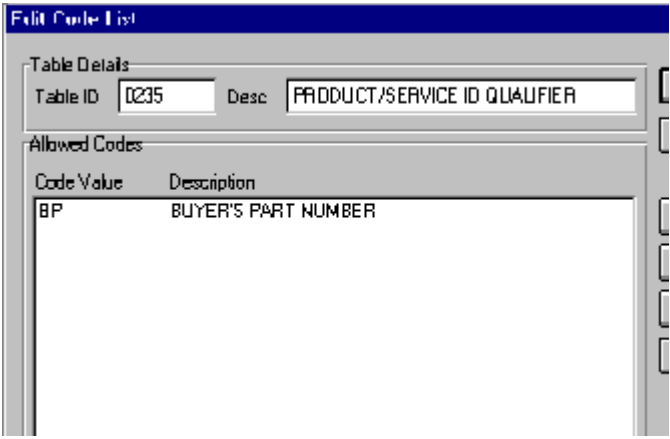
**Procedure** Use this procedure to delete a code list table.

Step	Action
1	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
2	If you want to delete a code list, highlight that list.
3	<p>Click <b>Delete</b> to delete the code list table.</p> <p><b>WARNING</b> <b>THE SELECTED TABLE DELETED WITHOUT WARNING.</b></p>



## How to delete a Code List Entry

Use this procedure to delete a code list entry.

Step	Action
1	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
2	Highlight the code list from which you want to delete an entry.
3	<p>Click <b>Change</b> to access the Edit Code List dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

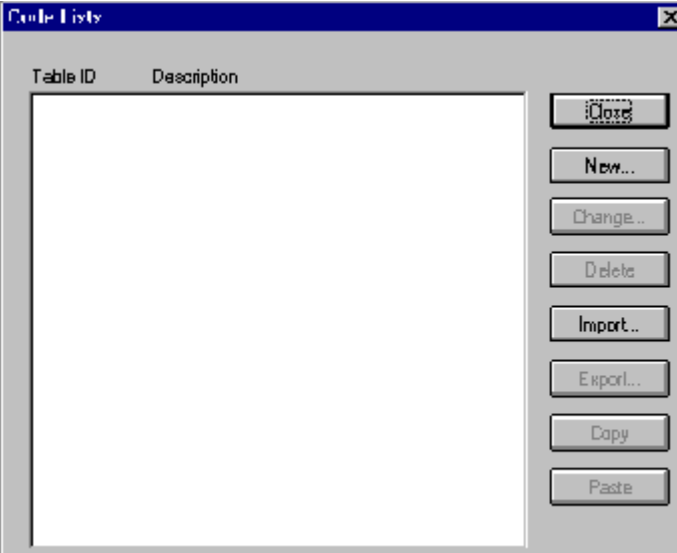
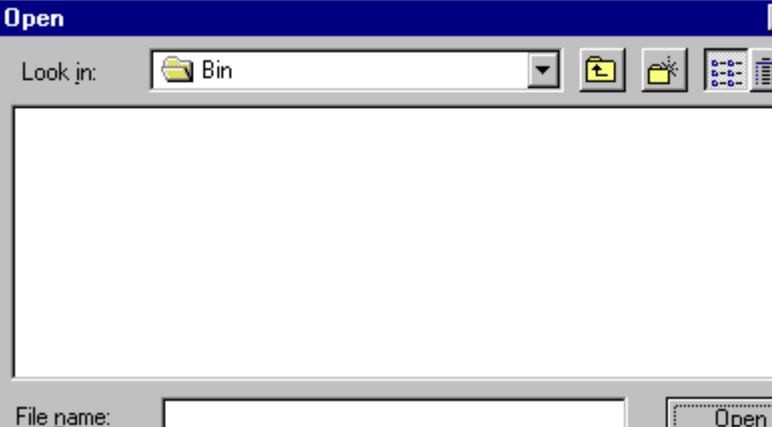
<b>(Contd) Step</b>	<b>Action</b>
4	To delete an entry, highlight it and click <b>Delete</b> to remove the entry. Delete any other entries you wish.  <b>WARNING</b> <b>THE SELECTED ENTRY IS DELETED WITHOUT WARNING.</b>
5	Click <b>OK</b> to save the code list table.

---

## Importing a Code List

**Introduction** The Code List Import function enables you to import code lists created for another map and share code lists with other users of Gentran:Server.

**Procedure** Use this procedure to import a code list table.

Step	Action
1	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
2	<p>Click <b>Import</b> to access the Open dialog box.</p>  <p style="text-align: right;">(Continued on next page)</p>

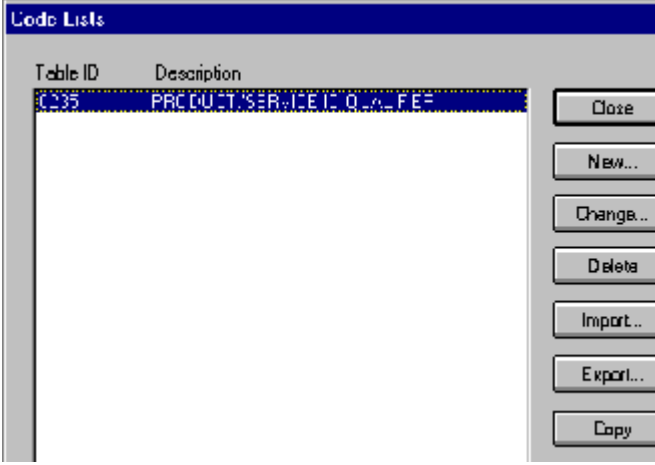
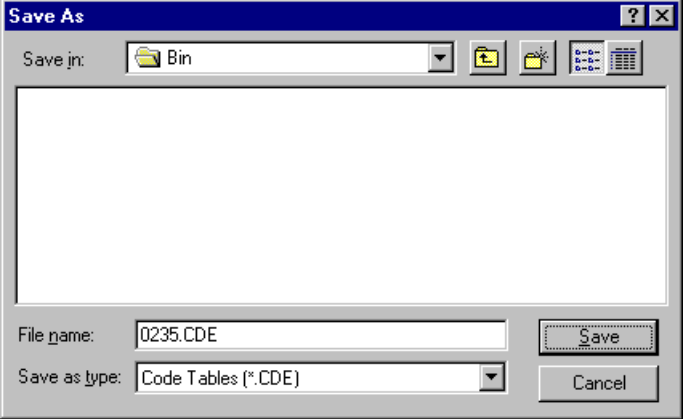
<b>(Contd) Step</b>	<b>Action</b>
3	Select the location of the code list file. The default location is Forms Integration install folder (the default is GENSRVNT\Bin).  <b>Note</b> The default file extension for code lists is .CDE.
4	The imported code list will be available for your use.
5	Click <b>Close</b> to exit the Code Lists dialog box.

---

## Exporting a Code List

**Introduction** The Code List Export function enables you to export code lists to file. This enables you to define a code list for one map and use that code list in another map. This function also allows you to share code lists with other users of Gentran:Server.

**Procedure** Use this procedure to export a code list table.

Step	Action
1	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
2	<p>Select a code list and click <b>Export</b>.</p> <p><b>System Response</b> The system displays the Save As dialog box.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>In the File name field, enter the file name and location. Verify that the correct folder is selected in the Save in drop-down list.</p> <p><b>Note</b> By default, the file name is listed as the table ID with a “.CDE” file extension. The default file location is the GENSRVNT\Bin folder.</p>
4	<p>Click <b>Save</b>.</p> <p><b>System Response</b> The code list is exported and you return to the Code Lists dialog box.</p>
5	<p>Click <b>Close</b>.</p> <p><b>System Response</b> You exit the Code Lists dialog box.</p>

---

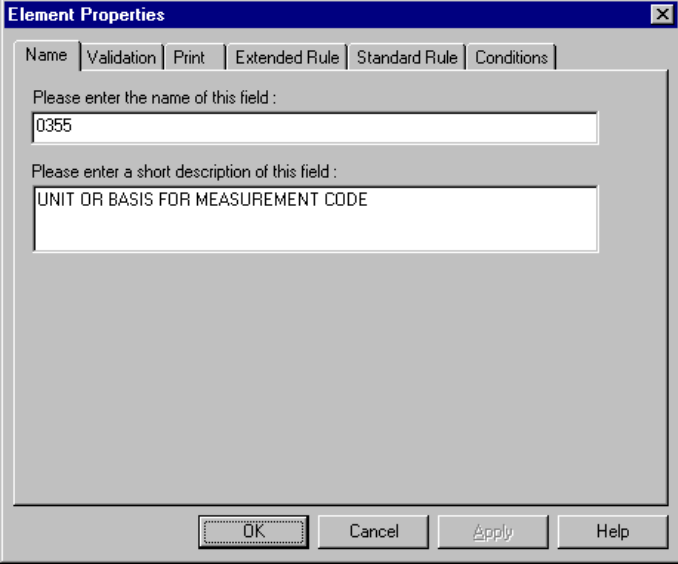
## Loading a Code List Table from the Standard

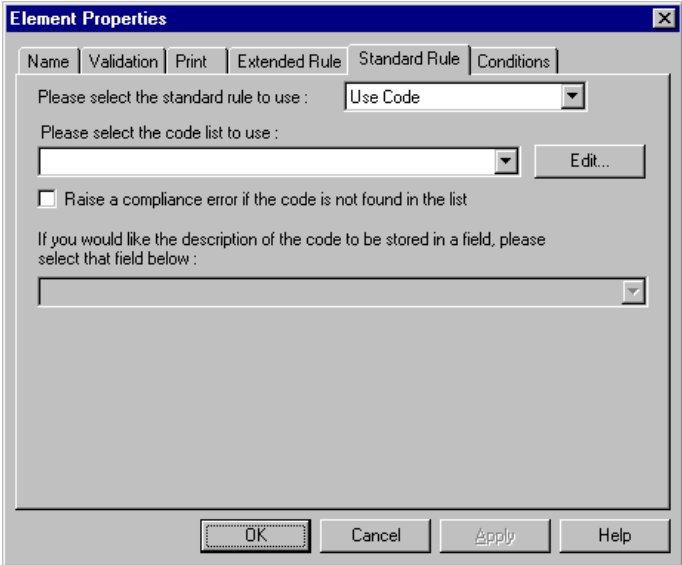
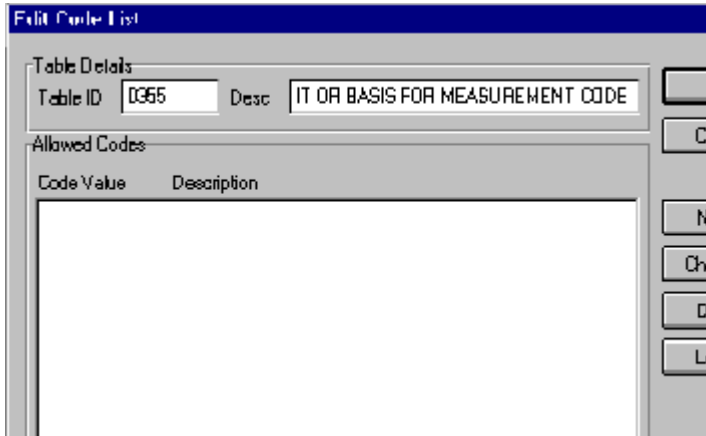
**Introduction** The EDI standards provide code list tables only for elements that use them.

**Example**

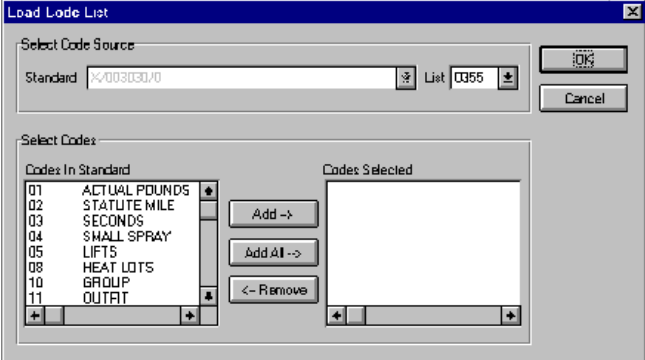
In the TD4 segment, the TD401 (Special Handling Code) and TD403 (Hazardous Material Class Code) have code tables provided by the standard.

**Procedure** Use this procedure to load a code list table from the standard.

Step	Action
1	Double-click the element for which you need to use a code table. The system displays the Element Properties dialog box (Name tab). 
2	Click the <b>Standard Rule</b> tab to access standard rule options. <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
3	<p>From the standard rule list, select <b>Use Code</b>.</p> 
4	<p>Click <b>Edit Table</b>.</p> <p><b>System Response</b> The system displays the Edit Code List dialog box.</p>  <p>The Table ID box contains the name of the element for which this code list table will be used. The Description box contains the description of the element for which this code list table will be used.</p> <p style="text-align: right;">(Continued on next page)</p>

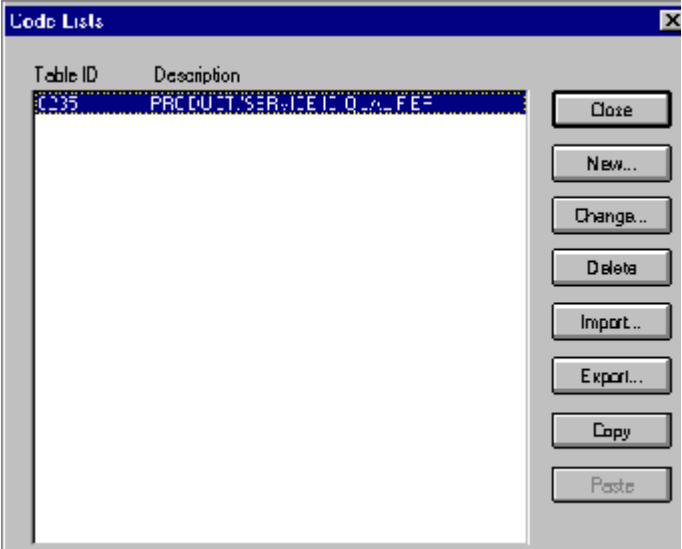


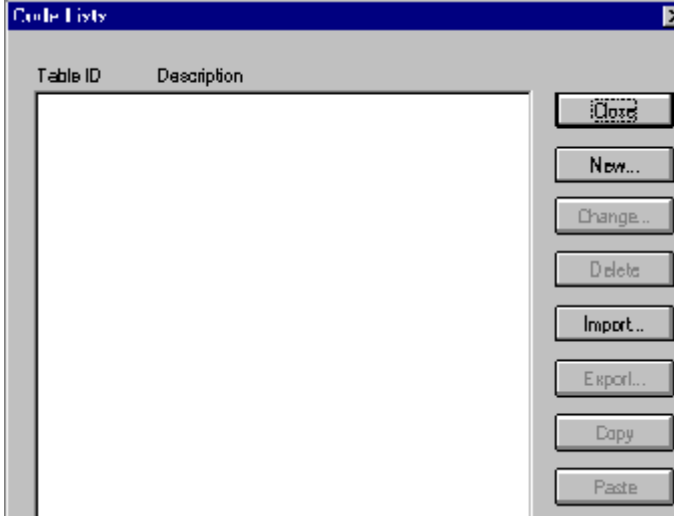
<b>(Contd) Step</b>	<b>Action</b>							
5	<p>Click <b>Load</b>.</p> <p><b>System Response</b> The system displays the Load Code List dialog box.</p> 							
6	<p>Use the following table to determine your next step.</p> <table border="1" data-bbox="618 919 1427 1608"> <thead> <tr> <th data-bbox="618 919 842 982"><b>IF...</b></th> <th data-bbox="842 919 1427 982"><b>THEN...</b></th> </tr> </thead> <tbody> <tr> <td data-bbox="618 982 842 1100">You want to select specific codes only,</td> <td data-bbox="842 982 1427 1100">Highlight each code from the Codes in Standard list that you want to be loaded. Click <b>Add</b> to move it to the Codes Selected list. Proceed to the next step.</td> </tr> <tr> <td data-bbox="618 1100 842 1608">To load the entire code list,</td> <td data-bbox="842 1100 1427 1608"> <ul style="list-style-type: none"> <li>▶ click <b>Add All</b> to select all the codes and move them to the Codes Selected list.</li> </ul> <p><b>WARNING</b></p> <p><b>WE RECOMMEND THAT YOU ONLY ADD THE CODES THAT YOU AND YOUR TRADING PARTNERS ARE ABLE TO CREATE OR ACCEPT. ADDING ALL THE CODES IN THE CODE TABLE (USING THE ADD ALL BUTTON) CREATES A MUCH LARGER TRANSLATION OBJECT THAN IF YOU ONLY USE SELECTED CODES.</b></p> <ul style="list-style-type: none"> <li>▶ Proceed to the next step.</li> </ul> </td> </tr> </tbody> </table>		<b>IF...</b>	<b>THEN...</b>	You want to select specific codes only,	Highlight each code from the Codes in Standard list that you want to be loaded. Click <b>Add</b> to move it to the Codes Selected list. Proceed to the next step.	To load the entire code list,	<ul style="list-style-type: none"> <li>▶ click <b>Add All</b> to select all the codes and move them to the Codes Selected list.</li> </ul> <p><b>WARNING</b></p> <p><b>WE RECOMMEND THAT YOU ONLY ADD THE CODES THAT YOU AND YOUR TRADING PARTNERS ARE ABLE TO CREATE OR ACCEPT. ADDING ALL THE CODES IN THE CODE TABLE (USING THE ADD ALL BUTTON) CREATES A MUCH LARGER TRANSLATION OBJECT THAN IF YOU ONLY USE SELECTED CODES.</b></p> <ul style="list-style-type: none"> <li>▶ Proceed to the next step.</li> </ul>
<b>IF...</b>	<b>THEN...</b>							
You want to select specific codes only,	Highlight each code from the Codes in Standard list that you want to be loaded. Click <b>Add</b> to move it to the Codes Selected list. Proceed to the next step.							
To load the entire code list,	<ul style="list-style-type: none"> <li>▶ click <b>Add All</b> to select all the codes and move them to the Codes Selected list.</li> </ul> <p><b>WARNING</b></p> <p><b>WE RECOMMEND THAT YOU ONLY ADD THE CODES THAT YOU AND YOUR TRADING PARTNERS ARE ABLE TO CREATE OR ACCEPT. ADDING ALL THE CODES IN THE CODE TABLE (USING THE ADD ALL BUTTON) CREATES A MUCH LARGER TRANSLATION OBJECT THAN IF YOU ONLY USE SELECTED CODES.</b></p> <ul style="list-style-type: none"> <li>▶ Proceed to the next step.</li> </ul>							
7	Click <b>OK</b> to load the code list.							
8	Click <b>Close</b> to exit the Code Lists dialog box.							
9	Click <b>OK</b> to exit the Element Properties dialog box.							

## Copying and Pasting Code Lists

**Introduction** The Code List Copy and Paste function enables you to copy code lists from one map to another.

**Procedure** Use this procedure to copy and to paste a code list table.

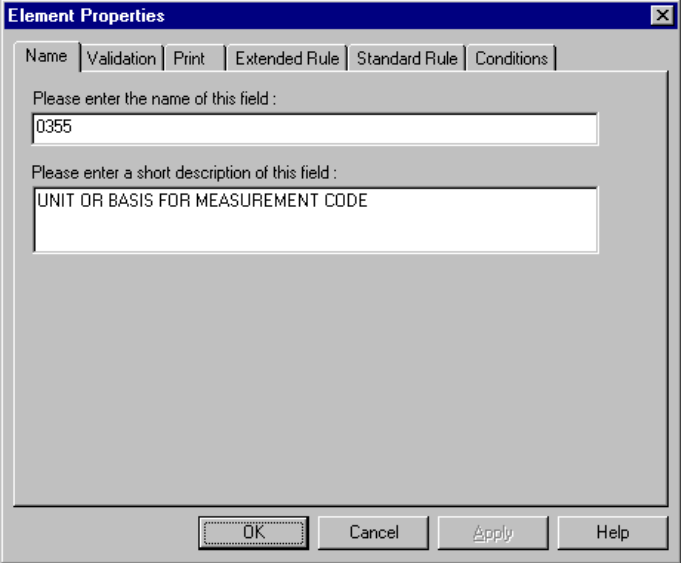
Step	Action
1	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
2	<p>Select a code list and click <b>Copy</b>.</p> <p><b>System Response</b> The selected code list is copied to the clipboard.</p>
3	<p>Click <b>Close</b> to exit the Code Lists dialog box.</p>
4	<p>Open the form in which you want to use the code list, if the form is not already open.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
5	<p>Select <b>Code Lists</b> from the Edit menu or the Main Toolbar.</p> <p><b>System Response</b> The system displays the Code Lists dialog box.</p> 
6	Click <b>Paste</b> to add the copied code list to this form.
7	Click <b>Close</b> to exit the Code Lists dialog box.

## Validating Data Against Code List Tables

**Introduction** For print translation objects, you can use code list tables to validate an element for compliance checking.

**Procedure** Use this procedure to validate data against a code list table.

Step	Action
1	<p>Double-click the element for which you need to validate data against a code table.</p> <p><b>Note</b> The standards provide code list tables only for elements that use them. For example, in the TD4 segment, the TD401 (Special Handling Code) and TD403 (Hazardous Material Class Code) have code tables provided by the standard.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, select <b>Use Code</b> .

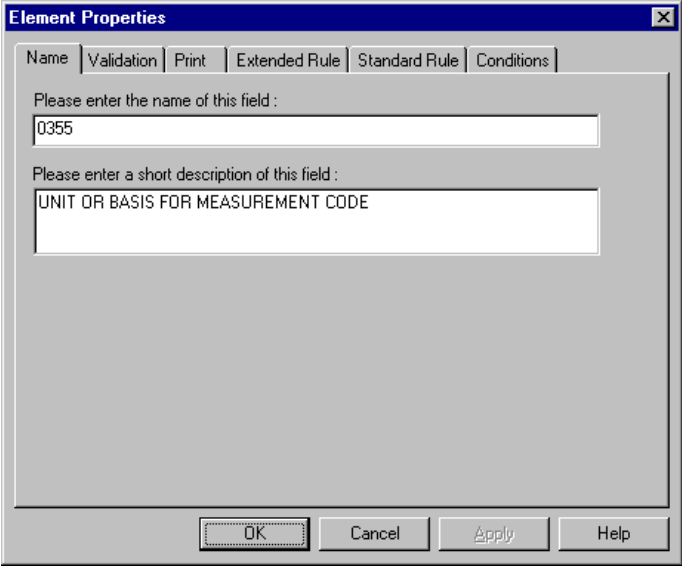
(Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
4	<p>Is the Tables list empty?</p> <ul style="list-style-type: none"> <li>▶ If yes, you need to load a code table.</li> </ul> <p><b>Reference</b> See <i>Loading a Code List Table from the Standard</i> on page 5 - 99 for instructions.</p> <ul style="list-style-type: none"> <li>▶ If no, from the code list, select the code list table that the data in this element is validated against.</li> </ul>
5	<p>Do you need to specify (for compliance reasons) that the element <i>must</i> contain one of the codes from the specified table (nothing else is valid for that field)?</p> <ul style="list-style-type: none"> <li>▶ If yes, select the compliance error check box. Proceed to the next step.</li> <li>▶ If no, proceed to the next step.</li> </ul>
6	Click <b>OK</b> to add this standard rule to the element.

# Loading Code Item Descriptions

**Introduction** Typically, you only load code item descriptions for print translation objects.

**Procedure** Use this procedure to load a code item description.

Step	Action
1	<p>Double-click the element for which you need to load a code item description.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, select <b>Use Code</b> .
4	<p>Is the code list table empty?</p> <ul style="list-style-type: none"> <li>▶ If yes, you need to load the code.</li> </ul> <p><b>Reference</b> See <i>Loading a Code List Table from the Standard</i> on page 5 - 99 for instructions</p> <ul style="list-style-type: none"> <li>▶ If no, from the code list, select the code list table that the data in this element will be validated against.</li> </ul> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
5	<p>Do you need to specify, for compliance reasons that the element <i>must</i> contain one of the codes from the specified table (nothing else is valid for that field)?</p> <ul style="list-style-type: none"><li>▶ If yes, select the compliance error check box. Proceed to the next step.</li><li>▶ If no, proceed to the next step.</li></ul>
6	<p>From the store description list, the element to which you want the description of the code item (that is used) to be mapped to when the selection is made.</p> <p><b>Example</b> If the code you used is “BP,” it would be useful to view the description of the code (“Buyer's Part Number”). If you selected element “0350” from the Store Field list, the description for the BP code is mapped to element “0350.”</p>
7	Click <b>OK</b> to add this standard rule to the element.

---





---

# Using Extended Rules

<b>Contents</b>	<b>Using Extended Rules . . . . .</b>	<b>6 - 4</b>
	▶ Overview . . . . .	6 - 4
	▶ Declarations Section . . . . .	6 - 5
	▶ Statements Section . . . . .	6 - 6
	▶ When Rules are Processed . . . . .	6 - 7
	<b>Defining Extended Rules. . . . .</b>	<b>6 - 9</b>
	▶ Overview . . . . .	6 - 9
	▶ Defining a Session Rule . . . . .	6 - 10
	▶ Defining a Form Component Extended Rule . . . . .	6 - 11
	<b>Extended Rule Syntax. . . . .</b>	<b>6 - 13</b>
	▶ Overview . . . . .	6 - 13
	▶ Keywords and Commands . . . . .	6 - 14
	▶ Operators . . . . .	6 - 16
	▶ Symbols. . . . .	6 - 17
	<b>Common Statements and Examples . . . . .</b>	<b>6 - 20</b>
	▶ Overview . . . . .	6 - 20
	▶ Assignment . . . . .	6 - 21
	▶ Datetime Expressions . . . . .	6 - 22
	▶ Conditional Logic . . . . .	6 - 24
	▶ String Conditions and Functions . . . . .	6 - 25
	▶ Numerical Functions . . . . .	6 - 27
	▶ Raise Compliance Error Function (CERROR) . . . . .	6 - 28
	▶ Remove Field Value Function (EMPTY) . . . . .	6 - 29
	▶ Existence of Data Function (EXIST) . . . . .	6 - 30
	▶ Count Function (COUNT) . . . . .	6 - 31
	▶ Delete Function (DELETE) . . . . .	6 - 32
	▶ File Pointer Functions (FSEEK, FTELL) . . . . .	6 - 33
	▶ Block of Data Functions (READBLOCK, WRITEBLOCK) . . . . .	6 - 34
	▶ Select Function (SELECT) . . . . .	6 - 35

▶ Update Function (UPDATE).....	6 - 36
▶ Insert Function (INSERT).....	6 - 37
▶ User Exit Function (EXEC).....	6 - 38
▶ ActiveX and User Exit Functions.....	6 - 39
<b>Alphabetic Language Reference .....</b>	<b>6 - 41</b>
▶ Overview.....	6 - 41
▶ atoi.....	6 - 42
▶ aton.....	6 - 43
▶ auditlog.....	6 - 44
▶ begin.....	6 - 46
▶ break.....	6 - 47
▶ cerror.....	6 - 48
▶ concat.....	6 - 55
▶ continue.....	6 - 56
▶ count.....	6 - 57
▶ createobject.....	6 - 58
▶ date.....	6 - 59
▶ delete.....	6 - 61
▶ deleteobject.....	6 - 62
▶ empty.....	6 - 63
▶ end.....	6 - 64
▶ exec.....	6 - 65
▶ exist.....	6 - 66
▶ fseek.....	6 - 67
▶ ftell.....	6 - 68
▶ get.....	6 - 69
▶ getiid.....	6 - 70
▶ if then...else.....	6 - 71
▶ index.....	6 - 72
▶ insert.....	6 - 73
▶ left.....	6 - 74
▶ len.....	6 - 75
▶ messagebox.....	6 - 76
▶ mid.....	6 - 78
▶ ntoa.....	6 - 79
▶ param.....	6 - 80
▶ queryobject.....	6 - 81
▶ readblock.....	6 - 82
▶ readbytes.....	6 - 83
▶ right.....	6 - 84
▶ select.....	6 - 85
▶ set.....	6 - 86

•	strdate .....	6 - 87
•	strstr .....	6 - 89
•	unreadblock .....	6 - 90
•	update .....	6 - 91
•	while...do .....	6 - 92
•	winexec .....	6 - 93
•	writeblock .....	6 - 95
•	writebytes .....	6 - 96
•	Select and Update Available Options .....	6 - 97

---

# Using Extended Rules

## Overview

---

### Introduction

Extended rules enable you to use a Gentran:Server proprietary programming language to perform virtually any operation you require. This is a full programming language that gives you access to the entire Gentran:Server internal storage area.

You can use these rules to define more complex translations than are available through the use of standard rules.

---

### Using Variables

You must declare any variable that is not already defined as part of the form specification before you use that variable with an extended rule.

---

### Extended rule sections

An extended rule generally consists of two sections: a *declarations* section followed by a *statements* section. This table describes extended rule sections.

Section Name	Description
Declarations	<p>The declarations section enables you to specify the names and types of any variables you use either in this rule or in any other rule that is within the scope of this rule.</p> <p><b>Note</b></p> <p>The declarations section is only required if you use additional variables.</p>
Statements	<p>The statements section is where you define the actions that you want the rule to execute.</p>

---

## Declarations Section

### Introduction

The variables that you define in the declarations section are used to store values. Variables consist of a name and a data type. Variable names can include alphanumerics and the “.” and “\_” characters. The first character in a variables name may not be a numeric. Please note that all variable names are case-sensitive.

### Note

A declaration must be terminated with a semicolon (;). To improve readability, you typically include a blank line in between the declaration and statement sections.

### Supported data types

This table describes supported data types.

Data Type	Description	Example
Integer	a whole number with no decimal component.	Declare <b>i</b> as an integer <b>integer i;</b>
Real	a whole number that may have a decimal component.	Declare <b>r</b> as a real number <b>real r;</b>
String	contains one or more printable characters.	Declare <b>s</b> as a 20-character string <b>string[20] s;</b>
Datetime	contains a date or time.	Declare <b>d</b> as a date or time <b>datetime d;</b>
Array	defines a table of multiple occurrences of a single data type.	Declare <b>a</b> as an array of 10 integers <b>integer a[10];</b>  Declare <b>p</b> as an array of 50 10-character strings <b>string[10] p[50];</b>

## Statements Section

---

**Introduction**

The actual work performed by an extended rule is defined in the statements section. A rule consists of a statement or a combination of statements (to perform more complex operations). A statement is a single operation that consists of a combination of expressions, keywords, commands, operators, and symbols.

---

**What is an expression?**

An *expression* is a logical unit (e.g.,  $A = B$  or  $A + B$ ) that the system evaluates. The statements section consists of a sensible combination of keywords, operators, and symbols.

---

## When Rules are Processed

### Introduction

You can specify pre- and post-session rules on the Session Rules dialog box. Pre-session extended rules are processed before the translation object, and are in scope for every extended rule defined in the translation object. Post-session rules are executed after the translation object is processed.

### Levels of extended rules

This table describes the form component level at which you can attach extended rules.

Form Component Level	Attach using the...
Groups, sub-groups, repeating segments, and repeating elements	Loop Extended Rule tab of the component's associated Properties dialog box.
Elements	Element Level Extended Rule dialog box.

### Scope

The *scope* of an extended rule determines which variables are accessible from within a given extended rule. The scope varies depending on the current state of the form. The scope of an extended rule is defined as:

- Pre-session extended rules (defined on the Session Rules dialog box) are in scope for every rule in the translation object.
- On New, On Store, On Open, and On Delete extended rules (defined on the Loop Extended Rules tab of the appropriate Properties dialog box) are in scope until the conclusion of the corresponding action (creating a new group, storing a group, opening an existing group, and deleting a group). On Open and On Delete extended rules are not available for print forms; only for screen entry.
- Field level extended rules are only in scope for the duration of the element.

An extended rule attached to the current form component depends on the type and state of the form component.

#### Example

If a current group to which the rule is attached is subordinate to another group, the parent group is automatically in scope for the duration of the entire child group, and the current hierarchical structure is also in scope for the duration of the child. An extended rule that is attached to an element is only in scope for the duration of the element. Field level extended rules are always processed after standard rules.

A variable is considered to be “in scope” if it was declared in the current rule or in the Pre-Session rule.

The translator builds the data storage area for a form based on the EDI structure. Therefore, extended rules address the form based on the hierarchy of the EDI file.

**Screen entry form processing**

The translator processes screen entry forms in the following manner:

Stage	Description
1	Reads the data into the Document Editor and processes all cumulative standard rules (e.g., accumulators, etc.). Does not process any extended or non-cumulative standard rules).
2	Executes group level extended rules in the following situations: <ul style="list-style-type: none"> <li>• On_New when a new group is created</li> <li>• On_Open when an existing group is edited</li> <li>• On_Store when a group is saved</li> <li>• On_Delete when a group is deleted</li> </ul>
3	Executes field level rules in the following sequence when focus is lost from the field: <ul style="list-style-type: none"> <li>• Non-cumulative standard rules</li> <li>• Extended rules</li> </ul>
4	When the form is saved or recalculated, all standard rules are executed.

**Print form processing**

The translator processes print forms in the following manner:

Stage	Description
1	Reads the data and processes all cumulative standard rules (e.g., accumulators). Does not process any extended or non-cumulative standard rules).
2	Executes group level extended rules in the following situations: <ul style="list-style-type: none"> <li>• On_New just prior to printing a group</li> <li>• On_Store just after printing a group</li> </ul>
3	Executes field level rules in the following sequence after the field is printed: <ul style="list-style-type: none"> <li>• Non-cumulative standard rules</li> <li>• Extended rules</li> </ul>



# Defining Extended Rules

## Overview

---

**Introduction**

The form component that an extended rule accesses depends on what you want the scope of the rule to be. You also need to determine when you want the rule to be executed (e.g., before or after the form component is processed).

The process you need to follow to define an extended rule varies slightly, depending on whether you are defining a session rule or a rule for a form component. You can define extended rules to access three different levels of form components.

---

## Defining a Session Rule

**Procedure** Use this procedure to define a pre- or post-session rule.

Step	Action
1	From the Edit menu, select <b>Session Rules</b> .  <b>System response</b> The system displays the Session Level Extended Rules dialog box.
2	Do you want to define a pre-session rule? <ul style="list-style-type: none"> <li>• If <i>yes</i>, click the Pre-session option.</li> <li>• If <i>no</i> (you want to define a Post-session rule), click the Post-session option.</li> </ul>
3	In the Editor list, type the extended rule.  <b>Note</b> The Session Level Extended Rules dialog box contains line and character number (within a line) indicators. These indicators, which are displayed to the lower right of the Editor box (the first indicator references the line number and the second references the character number), enable you to easily debug compile errors.
4	Do you want to check the rule for errors? <ul style="list-style-type: none"> <li>• If <i>yes</i>, click <b>Compile</b> to compile the extended rule.</li> </ul> <b>System response</b> Any warnings or errors are displayed in the Errors list. Double-click an error to instantly navigate to the line containing the error. <ul style="list-style-type: none"> <li>• If <i>no</i>, continue with step 6.</li> </ul> <b>Note</b> The Compile function gives you immediate feedback about the accuracy of your rule. The rule is compiled when you compile the entire translation object.
5	Correct any errors that the system flagged and click <b>Compile</b> again.  <b>Note</b> Repeat this process until there are no errors generated.
6	Click <b>OK</b> to add the extended rule.

## Defining a Form Component Extended Rule

**Procedure** Use this procedure to define an extended rule for a form component.

Step	Action										
1	Click the form component with the right mouse button to access the shortcut menu.										
2	<p>Select <b>Extended Rules</b> (or Extended Rule if the form component is an element) from the shortcut menu.</p> <p><b>System Response</b></p> <ul style="list-style-type: none"> <li>• If the form component is an EDI file, repeating group, repeating segment, the system displays the Loop Extended Rules tab of the appropriate Properties dialog box.</li> <li>• If the form component is an element, the system displays the Extended Rule tab of the Element Properties dialog box.</li> </ul>										
3	<p>Are you defining the rule for a group?</p> <ul style="list-style-type: none"> <li>• If yes, proceed to the next step.</li> <li>• If no, proceed to Step 5.</li> </ul>										
4	<p>Use this table to determine which extended rule execution option to select.</p> <p><b>Note</b> You can define an On New, On Store, On Open, and On Delete rule for a single group</p> <table border="1"> <thead> <tr> <th>IF you want to execute the extended rule when...</th> <th>THEN...</th> </tr> </thead> <tbody> <tr> <td>you create a new group in the translation object,</td> <td>click the <b>On New</b> option.</td> </tr> <tr> <td>you store a group,</td> <td>click the <b>On Store</b> option.</td> </tr> <tr> <td>you open an existing group,</td> <td>click the <b>On Open</b> option.</td> </tr> <tr> <td>If you want the extended rule to be executed when you delete a group,</td> <td>click the <b>On Delete</b> option.</td> </tr> </tbody> </table> <p style="text-align: right;">(Continued on next page)</p>	IF you want to execute the extended rule when...	THEN...	you create a new group in the translation object,	click the <b>On New</b> option.	you store a group,	click the <b>On Store</b> option.	you open an existing group,	click the <b>On Open</b> option.	If you want the extended rule to be executed when you delete a group,	click the <b>On Delete</b> option.
IF you want to execute the extended rule when...	THEN...										
you create a new group in the translation object,	click the <b>On New</b> option.										
you store a group,	click the <b>On Store</b> option.										
you open an existing group,	click the <b>On Open</b> option.										
If you want the extended rule to be executed when you delete a group,	click the <b>On Delete</b> option.										

<b>(Contd) Step</b>	<b>Action</b>
5	<p>In the extended rule list, type the extended rule.</p> <p><b>Reference</b> See <i>Alphabetic Language Reference</i> on page 6 - 41 for more information about rule syntax.</p> <p><b>Note</b> The Extended Rules dialog boxes contain line and character number (within a line) indicators. These indicators, which are displayed to the lower right of the Editor box (the first indicator references the line number and the second references the character number), enable you to easily debug compile errors.</p>
6	<p>Do you want to check the rule for errors?</p> <ul style="list-style-type: none"> <li>▶ If <i>yes</i>, click <b>Compile</b> to compile the extended rule.</li> </ul> <p><b>System response</b> Any warnings or errors are displayed in the Errors list. Double-click an error to instantly navigate to the line containing the error.</p> <ul style="list-style-type: none"> <li>▶ If <i>no</i>, continue with step 7.</li> </ul> <p><b>Note</b> The Compile function gives you immediate feedback about the accuracy of your rule. The rule is compiled when you compile the entire translation object.</p>
7	<p>Correct any errors that the system flagged and click <b>Compile</b> again. Repeat this process until there are no errors generated.</p>
8	<p>Click <b>OK</b> to add the extended rule.</p> <p><b>Note</b> When an element contains an extended rule a black asterisk appears to the right of the element icon.</p>

# Extended Rule Syntax

## Overview

---

**Introduction**

The statements section of an extended rule consists of a sensible combination of keywords, operators, and symbols. The correct syntax for each of these component is explained below.

**Note**

You use spaces and operators to separate keywords and symbols. You cannot string two keywords sequentially together without an operator.

---

# Keywords and Commands

---

## Introduction

A *keyword* is a fixed defined use of a word that indicates how the programming language should be interpreted. There are two types of keywords. The first type of keyword controls the flow of execution of the defined rule. These keywords are used to evaluate conditions and perform looping operations.

The second type of keyword is a *command*. Commands perform actions on variables and are responsible for the movement of data.

---

## Valid keywords

A list of Gentran:Server execution control keywords is as follows:

- ▶ IF
  - ▶ THEN
  - ▶ ELSE
  - ▶ BEGIN
  - ▶ END
  - ▶ WHILE
  - ▶ DO
  - ▶ CONTINUE
  - ▶ BREAK
- 

## Valid commands

A list of Gentran:Server commands is as follows:

- ▶ AUDITLOG
  - ▶ GET
  - ▶ SET
  - ▶ STRDATE
  - ▶ CONCAT
  - ▶ LEN
  - ▶ ATOI
  - ▶ ATON
  - ▶ CERROR
  - ▶ EMPTY
  - ▶ EXIST
  - ▶ INDEX
- 

(Continued on next page)

**Valid commands  
(contd.)**

- ▶ NTOA
- ▶ COUNT
- ▶ DELETE
- ▶ FSEEK
- ▶ FTELL
- ▶ LEFT
- ▶ RIGHT
- ▶ MID
- ▶ SELECT
- ▶ INSERT
- ▶ UPDATE
- ▶ READBLOCK
- ▶ UNREADBLOCK
- ▶ READBYTES
- ▶ WRITEBYTES
- ▶ CREATEOBJECT
- ▶ DELETEOBJECT
- ▶ QUERYOBJECT
- ▶ GETID
- ▶ WINEXEC
- ▶ DATE
- ▶ EXEC
- ▶ PARAM

**Note**

A statement must be terminated with a semicolon (;).

# Operators

## Introduction

Operators define the simplest operation in an expression. This table describes valid extended rule operators.

Operator	Description
+	addition, concatenation
-	subtraction
*	multiplication
/	division
=	assignment, equality
>	greater-than
<	less-than
>=	greater-than or equal to
<=	less-than or equal to
!=	not equal to
!	logical not
&	logical and
	logical or
<<	date modification



# Symbols

---

## Introduction

Operations are performed on symbols. The symbols that you can use in Gentran:Server extended rules are variables, constants, form components/internal storage, arrays, and accumulators. You can address existing form components and you also have the ability to create additional instances of form components, as long as the form component is originally defined in internal storage. You can use this function to create, for example, extra line items when one line item field is already defined in internal storage.

---

## Symbol syntax

You must address each type of symbol in the proper syntax.

### String constant

To address a string constant, you must enclose the constant value in quotes:

```
#fieldname = "HDR";
```

where HDR is the constant value

### Addressing or creating a field in internal storage

To address a field or create a field in internal storage, within the scope of the current mapping action, the syntax is #FIELD\_NAME

```
#field_1 = 2;
```

where 2 is a numeric constant value

### Addressing or creating a field in a group

To address a field within a group or create a field within a group in internal storage within the scope of the current hierarchy, the syntax is \$GROUP.#FIELD\_NAME

```
$N1.#0234
```

---

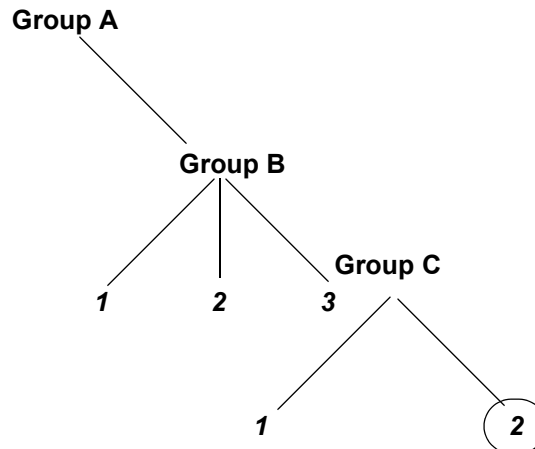
(Continued on next page)

**Symbol syntax  
(cont.)****Addressing or creating a group in internal storage**

To fully address a group in the entire internal storage area or create a group in internal storage, the syntax is \$LOOP[index1][index2][index3] where the index entries indicate the hierarchical structure of the loop and enable you to address specific instances of a group:

**\$Group\_C[3][2].#Field\_2**

where you are specifying the second instance of Group\_C within the third instance of Group\_B:

**Addressing an array**

To address an array (of any type), you address each element of the array individually. For example, if array\_1 is an array (of integers) and is declared:

**integer array\_1[5]**

with variables 0 through 4, each element of the array is addressed individually as follows:

```

array_1[0]
array_1[1]
array_1[2]
array_1[3]
array_1[4]

```

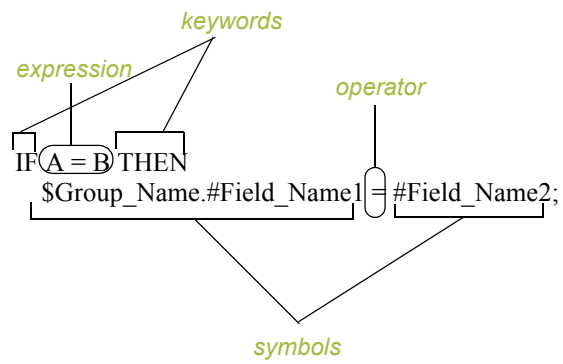
**Accessing an accumulator**

An accumulator can be accessed in the same manner as variables or internal storage. To address an accumulator, use the syntax accum(n) where “n” is the number (not the name) of the accumulator:

**accum(2) = 5;**

**Example of a simple statement**

This diagram illustrates an example of a simple statement.



# Common Statements and Examples

## Overview

---

**Introduction**

Some of the common statements are as follows:

- assignment
  - datetime expressions
  - conditional logic
  - string conditions and functions
  - numerical functions
  - raise compliance error function
  - remove field value function
  - existence of data function
-

# Assignment

---

## Introduction

The assignment statement is the most powerful and most often used extended rule statement. In the most simple form, it is written as follows:

```
variable=expression
```

However, you can use this statement in more flexible and complex ways, defined as follows:

```
numeric_variable=numeric_expression  
numeric_field=numeric_expression  
string_variable=string_expression  
string_field=string_expression  
datetime_variable=datetime_expression  
datetime_field=datetime_expression
```

---

## Numeric expressions

A numeric expression can consist of number, numeric fields, numeric variables, and numeric functions combined with the standard arithmetic operators.

---

## String expression

A string expression can consist of string constants, string fields, string variables, and string functions concatenated with the “+” operator.

---

## Datetime expression

A datetime expression can consist of a datetime constant, datetime field or datetime variable.

---

## Examples

Some examples of assignment expressions are as follows:

```
a = 5;  
a = b + c;  
s = "hello";  
s = s + "world";
```

---

# Datetime Expressions

## Introduction

Datetime expressions consist of a datetime variable and (optionally) datetime modifiers. Datetime expressions can be written using datetime constants if you are using the standard syntax, as follows:

```
year/month/day
hour:minute:second
year/month/day/hour:minute:second
```

Datetime expressions can also be written with datetime fields, variables, or using *date* and *time* functions.

## Date syntax

Date functions are written as follows (month specified as 0-11):

```
datetime d;

d = date(1995,4,6);
d = date(1995,4,6,12,0);
d = date("%y/%m/%d", "95/4/6");
```

The `d = date("%y/%m/%d", "95/4/6");` format enables you to convert any string format type into a datetime format type by indicating a format mask ("`%y/%m/%d`") along with the string ("`01/4/6`") you want to convert. You use this function if you are using non-standard syntax and need to specify the syntax you are using.

## << operator

You can use the << operator to modify your datetime variable by adding time increments (e.g., days, weeks, years). For example:

```
datetime d;
d=d <<weeks(2);
//This adds 2 weeks to d.
```

## Time syntax

Time functions are written as follows:

```
d = time(12,0);
d = time(12,0,59);
```

You can use the << operator to modify your datetime variable by adding time increments (e.g., seconds, minutes, years). For example:

```
datetime d;
d=d <<seconds(1);
//This adds 1 second to d.
```

---

**Get and set functions**

The *get* and *set* functions enable you to access (get) or modify (set) individual components of a datetime type. These functions are used as follows:

```
integer a;  
datetime d;  
a = get days (d);  
a = get hours (d);  
set hours (d,a);  
set days (d,a);
```

---

## Conditional Logic

---

### Introduction

Gentran:Server uses conditional logic to test conditions and then, depending on the results of the test, perform different operations. Conditions can be nested to any level. Do not end conditions with a semicolon (;) – this terminating syntax is necessary for statements *only*.

---

### IF...THEN... ELSE

You can use the IF/THEN keywords to execute one or more statements conditionally. The condition is typically a comparison, but it can be any expression that concludes with a numeric value. Gentran:Server interprets the value as either *true* or *false*. The system interprets a zero value as false and a nonzero value as true.

#### Note

If you include more than one statement in the body of an IF/THEN loop, you must surround the statements with the BEGIN/END keywords. If you only use a single statement, you can omit the BEGIN and END.

Gentran:Server evaluates the IF/THEN condition, and if it is true, the system executes all the statements that follow the THEN keyword.

You can use the ELSE keyword in conjunction with IF/THEN to define several blocks of statements, one of which is executed. Gentran:Server tests the first IF/THEN condition. If the condition is false, the system proceeds to test each sequential condition until it finds one that is true. The system executes the corresponding block of statements for the true condition. If none of the IF/THEN conditions are true, the system executes the statements following the ELSE keyword. The IF/THEN/ELSE statement is written as follows:

```
IF condition THEN
BEGIN
    statement1;
    statement2;
END
ELSE
BEGIN
    statement3;
    statement4;
END
```

---



# String Conditions and Functions

---

## Introduction

You can use string conditions in IF/THEN and IF/THEN/ELSE statements to perform comparisons between strings. Examples of the syntax are as follows:

```
IF s1 = s2 THEN
IF s1 < s2 THEN
    IF s1 > s2 THEN
```

The following string functions are also available for you to use:

- left
- right
- mid
- strdate
- concat
- strstr

---

## Left, right, mid syntax

The *left*, *right*, and *mid* functions enable you to extract substrings from a string. The left function extracts a specified number of characters from the left of the string variable or field and returns the result as a string. The right function extracts a specified number of character from the right of the string variable and returns the result as a string. The mid functions extracts from a specified position in the string to the right, for a specified number of characters. This is an example of how the statements are used.

```
string[10] s;
string[3] s1;
string[3] s2;
string[4] s3;
string[7] s4;

s = "abcdefghij";
s1 = left(s,3);
s2 = right(s,3);
s3 = mid(s,3,4);
```

---

## Strdate syntax

The *strdate* function converts a datetime type into a string using a format that you specify. This function allows you to include static characters such as a slash (/), which gives you access to full date support.

### Syntax

```
datetime d;
string[8] s;

strdate(d, "%y/%m/%d", s);
```

---

**Concat syntax**

The *concat* function concatenates a specified number of characters from one string onto the end of another string. The following example demonstrates the syntax to concatenate five characters from string “s2” onto the end of string “s1”:

```
string[10] s1,s2;  
concat(s1,s2,5);
```

---

**Strstr syntax**

The *strstr* function finds a substring inside a string. This function returns the position of the first instance of the designated substring. If this function does not find the specified substring inside the string, it returns a value of -1.

**Syntax**

```
integer d;  
  
d = strstr("hello", "el");
```

---

# Numerical Functions

---

## Introduction

The numerical functions enable you to convert one data type to another. The numerical functions that are available for you to use are the following:

- `len`
  - `atoi`
  - `atof`
  - `ntoa`
- 

## Len syntax

The `len` function counts and returns the number of characters in a string.

### Example

```
integer a;  
a = len("hello");
```

---

## Atoi, atof, ntoa syntax

The `atoi` function converts strings into integers.

The `atof` function converts string into real numbers.

The `ntoa` function converts integers and real numbers into strings.

### Example

```
integer a;  
real b;  
string[8] s;  
a = atoi("5");  
b = atof("5.5");  
ntoa(5.5, s);
```

---

## Raise Compliance Error Function (CERROR)

---

### Introduction

The *cerror* function raises a compliance error and reports the target statement (the statement you specify) on the translation report.

### Syntax

```
cerror(100,$ILD[0][1][1].#VATC);
```

The above example raises compliance error 100 on the VATC element of the specified instance of the ILD group. You typically specify this function as a action to be performed if a condition is false.

---

## Remove Field Value Function (EMPTY)

---

### Introduction

The *empty* function sets the value of a field in internal storage to null. This function is not the same as setting the value of a field to a zero length string (" ") or to zero.

### Syntax

```
empty($ILD[0][1][1].#VATC);
```

The above example sets the value of the specified instance of the VATC element to null. This function is typically used to prevent output to the specified field.

---

## Existence of Data Function (EXIST)

---

### Introduction

The *exist* function tests to determine if a field is empty (null). It returns a non-zero (true) value if there is data in a specified field in internal storage. If data is not present in the specified field, this function returns a zero (false) value. This function is typically used as a part of a condition.

### Syntax

```
IF exist($ILD[0][1][1].#VATC) THEN
```

The above example returns a non-zero value if the condition is true (data is present in the specified instance of the VATC element). A zero value is returned if the condition is false (no data is present in the specified instance of the VATC element).

There are some situations when the IF EXIST returns a non-zero (true) value whether or not the condition is true. You can work around this by only using IF EXIST for date- and number-type fields. Be certain that all references to the field which is interrogated are nested within the IF EXIST begin block. For string-type fields, use the following format:

```
IF Field1 = " "
```

---

## Count Function (COUNT)

---

### Introduction

The *count* function returns the total number of times a group (or repeating record) has iterated.

### Syntax

```
count (GROUPNAME [3] [*] );
```

The above example returns the total iterations of the GROUPNAME group within the third iteration of the parent group.

---

## Delete Function (DELETE)

---

**Introduction**    The *delete* function deletes the specified iteration of a repeating record or group.

**Syntax**

```
delete (GROUPNAME [iteration]);
```

---



## File Pointer Functions (FSEEK, FTELL)

---

### Introduction

The *fseek* function moves the file pointer to a new location which is a specified number of bytes (offset) from the designated point of origin in the file (the point of origin may be the beginning of the file or relative to either the end of the file or the current position).

To invoke the *fseek* function against the input file, the value for *current\_file* is "0." To invoke the *fseek* function against the output file, the value for *current\_file* is "1." The *fseek* function is typically only used in conjunction with the *ftell*, *readblock*, and *writeblock* functions.

### Syntax

```
fseek(current_file,offset,origin);
```

The *ftell* function obtains the current position of the file pointer and returns it as an integer. To invoke the *ftell* function against the input file, the value for *current\_file* is "0." To invoke the *ftell* function against the output file, the value for *current\_file* is "1." The *fseek* function is typically only used in conjunction with the *fseek*, *readblock*, and *writeblock* functions.

### Syntax

```
numeric_variable = ftell(current_file);
```

---

## Block of Data Functions (READBLOCK, WRITEBLOCK)

---

### Introduction

The *readblock* function reads a block of data (segment or record) from the input file and places it into the argument of a string variable. The *writeblock* function writes the data contains in the argument of a string variable to the output file.

The *readblock* and *writeblock* functions are used in conjunction with each other to pass a block of data from the input file to the output file without compliance checking or testing for proper EDI syntax. Together these functions provide a more efficient alternative of using “wildcard” segments, which are typically implemented in build and break maps.

### Syntax

```
readblock(string_variable);  
writeblock(string_variable);
```

---

## Select Function (SELECT)

---

### Introduction

The *select* function allows information to be retrieved from the database tables. Only the tables and fields available in the Select standard rule are available for the select extended rule. In the command syntax “expression” and “receiverlist” can be a string field, string variable, or string literal. It is important to note that the table and field names for the select extended rule are slightly different than those depicted in the standard rule. A listing of these table and field names can be found at the end of this chapter.

### Syntax

```
select fieldname into receiverlist from tablename where key =  
expression [and key = expression];
```

---

## Update Function (UPDATE)

---

### Introduction

The *update* function allows information in the database tables to be updated. This function is similar to the Update standard rule, except that it provides more flexibility. Only the tables and fields available in the Update standard rule are available for the update extended rule. In the command syntax “expression” can be a string field, string variable, or string literal. It is important to note that the table and field names for the update extended rule are slightly different than those depicted in the standard rule. A listing of these table and field names can be found at the end of this chapter.

### Syntax

```
update tablename set fieldname =expression [fieldname =expression]  
where key = expression [and key = expression];
```

---

# Insert Function (INSERT)

---

**Introduction** The *insert* function allows information in the database tables to be updated.

**Syntax**

```
insert into tablename [ (fieldlist) ] [ (valuelist) ];
```

---

## User Exit Function (EXEC)

---

### What is a user exit?

A user exit is an extended rule that enables the map to temporarily exit translation to enhance your functionality or fulfill specific requirements that Gentran:Server does not perform during normal translation.

#### Reference

See User Exits, appendix B, for more information.

---

### Placing the extended rule

You can use the exec function in an extended rule for a map component at any hierarchical level, including field level extended rules, if appropriate.

#### Note

When you apply a user exit to a map component, subordinate map components may also be able to execute the same user exit.

---

### Allowable data types

These are the extended rule data types that you can use with exec:

- INTEGER
- REAL
- STRING

#### Note

You cannot use date and time data unless you process it as a string data type.

---

# ActiveX and User Exit Functions

---

## Introduction

Gentran:Server now supports additional extended rule functionality to allow you to use Microsoft's ActiveX Data Objects (ADO) from within extended rules, as well as providing enhancements to user exit support.

---

## ActiveX indexed properties

Gentran:Server now supports ActiveX indexed properties, specifically these types of index:

- variant
- numeric
- string

The index support allows you to use these syntaxes (if they are supported by your Automation server):

```
n = ob.property[1];
n = ob.property["Count"];
n = ob.property[ob.method()];
```

Gentran:Server also supports chaining of ActiveX method calls and properties, which simplifies extended rules for Automation servers with moderately complex object models (e.g., ADO).

### Example

The following is an example of chaining statements:

```
recordset.fields.item["MessageId"].value
```

---

## Createobject syntax

The *createobject* function enables you to create an instance of an ActiveX Automation Server.

### Syntax

```
object = createobject("ProgID");
```

ProgID is the programmatic identifier. An example of a ProgID is:

**"InternetExplorer.Application"**

---

---

**Deleteobject  
syntax**

The *deleteobject* function enables you to delete an instance of an ActiveX Automation Server. An object must be deleted before the end of the map that uses it. It is more efficient to delete the object immediately on completion (using the DELETEOBJECT command), although the Gentran:Server translator will delete the object automatically at the end of the map. Also, if you assign one object to another one, both copies of the object must be deleted for that object to be properly unloaded.

**Syntax**

```
deleteobject (object);
```

---

**Getiid syntax**

The *getiid* function enables you to obtain the unique identifier for an interface, by using the string-character name of the interface to return the globally unique identifier that is used by software to run the interface.

**Syntax**

```
string_variable = getiid("ProgID", "InterfaceID");
```

InterfaceID (IID) is the interface identifier. An example of an IID is:  
**"IWebBrowser2"**

---

**Queryobject  
syntax**

The *queryobject* function is used to request a different interface on an existing object.

**Syntax**

```
object2 = queryobject (object1, "{IID}");
```

**Note**

You may not use extended rules that compare two ActiveX properties or method results, or any combination of properties or method results. This type of comparison is invalid because property and method types are unknown prior to compilation and thus it is not possible to generate the correct comparison code.

---



# Alphabetic Language Reference

## Overview

### Introduction

This section contains an alphabetic reference for the Gentran:Server proprietary programming language keywords and commands. This enables you to quickly and easily refer to information on specific keywords or commands.

### Typeface conventions

This table lists the typeface conventions used in this section.

IF the typeface follows this example...	THEN it is used to indicate...
While	what you need to type.
condition	in syntax, information that you need to provide.
[1]	that, in syntax, items inside square brackets are optional.

### Typeface conventions

The following programming guidelines are utilized in this section:

- Keywords and commands are shown in all lower case letters.
- Map components are shown in all upper case letters.
- A double slash (//) introduces comments.
- Lines that are too long to fit on one line in this section may be continue on the next line using a line-continuation character (Â).

# atoi

---

**Introduction**

The **atoi** function is a numerical function that converts strings into integers. The numerical functions enable you to convert one data type to another.

---

**Syntax**

```
int = atoi(string);
```

where:     int    =   integer variable  
          string =   string variable

---

**Example**

```
integer a;  
string[20] s;  
s = "5";  
a = atoi(s);  
// "a" contains the value 5
```

---

# aton

---

**Introduction**

The **aton** function is a numerical function that converts strings into real numbers. The numerical functions enable you to convert one data type to another.

---

**Syntax**

```
real = aton(string);
```

where: real = real number variable  
string = string variable

---

**Example**

```
real a;  
string[20] s;  
s = "5.5";  
a = aton(s);  
// "a" contains the value 5.5
```

---

# auditlog

---

<b>Introduction</b>	<p>The <b>auditlog</b> function enables you to write user-defined audit messages to the Gentran:Server Audit Log. When an extended rule that calls an auditlog operation is executed, it writes the specified user-defined message to the Audit Log.</p> <p><b>Note</b> This function does not return a value.</p>
<b>Syntax</b>	<pre>auditlog(MessageID, Type, Key, [, string1] [, string2] [, string3] [, string4] [, string5] [, string6] [, string7]);</pre>
<b>MessageID</b>	<p>The first parameter, MessageID, is the user-defined audit message identifier, which must be an integer value.</p>
<b>Type</b>	<p>The second parameter, Type, is a keyword that identifies the type of audit message. These pre-defined keywords are valid for Gentran:Server:</p> <ul style="list-style-type: none"><li>• AL_PROC (processing)</li><li>• AL_MSG (message)</li></ul> <p><b>Note</b> You may also supply an integer value to account for a type for which a keyword is not currently defined.</p>
<b>Key</b>	<p>The third parameter, Key, is a keyword that is either zero (if the type of parameter two is “AL_PROC”) or the identification of a piece of data of the specified type. These pre-defined keywords are valid for Gentran:Server:</p> <ul style="list-style-type: none"><li>• AL_KEY_INPUT</li><li>• AL_KEY_OUTPUT</li></ul> <p><b>Note</b> You may also supply an integer value to account for a key for which a keyword is not currently defined.</p>

---

(Continued on next page)

---

**Parameters 4 through 10**

Parameters four through ten are optional string values that the user-defined message may require to fill in variables defined in the audit message.

---

**Example1**

```
auditlog(MessageID, AL_PROC, 0, ...);  
//Issues a processing message in any map.
```

---

**Example2**

```
auditlog(MessageID, AL_MSG, Oatoi(param(1)), ...);  
//Issues a data audit for the message which is currently  
//processing.
```

---

# begin

---

## Introduction

Gentran:Server uses conditional logic to test conditions and then, depending on the results of the test, perform different operations. Conditions can be nested to any level. You can use the **if/then** keywords to execute one or more statements conditionally. If you include more than one statement in the body of an **if/then** loop, you must surround the statements with the **begin/end** keywords. If you only use a single statement, you can omit the **begin** and **end**.

## Note

Do not end conditions with a semicolon (;) – this terminating syntax is necessary for statements *only*.

---

## Syntax

```
if condition then
begin
    statement1;
    statement2;
end
```

---

# break

---

## Introduction

The **break** function terminates the execution of the nearest enclosing **while** loop, and passes control to the statement that follows the **end** keyword. The **break** keyword is generally used in complex loops to terminate a loop before several statements have been executed.

---

## Example

```
integer i
  i = 0;

While i<10 do
Begin
  #Total = Total + 50;
  i = i + 1;

  If #Total > 100000
    Break;
  Else
    Continue;
End
//While the value contained in the variable "i" is less than ten,
//50 will be added to the field Total.
//If the value in the field Total becomes greater than 100000
//before "i" equals 10, break out of the while loop else continue
//processing until "i" equals 10.
```

---

## cerror

### Introduction

The *cerror* function raises a compliance error and reports the target statement (the statement you specify) on the translation report. You typically specify this function as an action to be performed if a condition is false. This table describes when the cerror function is supported in Gentran:Server.

IF the map/form is of type...	THEN the cerror function is...
Screen entry	Not valid.
Print	Not valid.
Export	Only valid on the input side of the map.
Import	Valid on the input or output side of the map.
Break (Interchange, Group, or Transaction Set)	Only valid on the input side of the map.
Build (Interchange, Group, or Transaction Set)	Not valid.

### Syntax

```
cerror(error_number,$GROUP_NAME[index][index][index].
&#FIELD_NAME);
```

### Example

```
cerror(100,$ILD[0][1][1].#VATC);
//This raises compliance error 100 on the VATC element of the
//specified instance of the ILD group
```

### Compliance error codes

The compliance error codes are as follows

Error Number	Error Generated
100	Mandatory Element
105	Conditional Element
110	Incorrect Element Format
111	Invalid Date

(Continued on next page)



<b>(Contd) Error Number</b>	<b>Error Generated</b>
112	Invalid Time
113	Invalid String
114	Invalid Number
115	Incorrect Element Length
116	Element Too Short
117	Element Too Long
118	Invalid Overpunch Data
119	Invalid Numeric XData
120	Too Many Components
121	Too Many Elements
125	Element Delimiter
130	Conditional Relation
135	Element Code
140	Standard Rule Failure
145	Binary Substitution Failed
200	Mandatory Component
205	Conditional Component
210	Incorrect Component Format
215	Incorrect Component Length
220	Component Delimiter
225	Component Code
300	Mandatory Segment
305	Segment Sequence
310	Invalid Start End
315	Incorrect Block Format

(Continued on next page)

<b>(Contd) Error Number</b>	<b>Error Generated</b>
320	Segment Terminator
330	Unexpected Segment
400	Session Error
405	Unknown Partner
410	Invalid Control Number
415	Invalid Sub Count
420	Unknown Relationship
425	Unknown Standard
430	Unknown Version
435	Unknown Transaction
440	Duplicate Control Number
445	Duplicate Doc Name
500	ActiveX Bad Param Count
505	ActiveX Bad Argument Type
510	ActiveX Unknown Name
515	ActiveX Type Mismatch
520	ActiveX Missing Required Param
525	ActiveX Server Unavailable
530	ActiveX Object Not Created
535	ActiveX Other Error
600	XML Element Type Match
601	XML Element Root Type
602	XML Element Required
620	XML Attribute Undeclared
621	XML Attribute Not Unique

(Continued on next page)

<b>(Contd) Error Number</b>	<b>Error Generated</b>
622	XML Attribute Required
623	XML Attribute Fixed Error
624	XML Attribute ID Not Unique
625	XML Attribute ID REF Match
630	XML Attribute Parse Name Error
631	XML Attribute Parse Equal Error
632	XML Attribute Parse OQuote Error
633	XML Attribute Parse CQuote Error
640	XML Attribute Name Error
641	XML Attribute Names Error
642	XML Attribute NMTOKEN Error
643	XML Attribute NMTOKENS Error
644	XML Attribute Entity Error
645	XML Attribute Entities Error
660	XML Reference Char Invalid
661	XML Reference Entity Recursion
662	XML Reference Entity Undeclared
699	XML Unknown Error
700	ODBC Data Source Open Error
701	ODBC Data Source Rollback
702	ODBC Data Source Commit Error
703	ODBC Data Source Rollback Error
710	ODBC Query Open Error
711	ODBC Command Error
712	ODBC Cursor Error

(Continued on next page)

<b>(Contd) Error Number</b>	<b>Error Generated</b>
720	ODBC Output Table Error
721	ODBC Output Operation Error
730	ODBC Mandatory Element
731	ODBC Data Conversion Error
732	ODBC Incorrect Element Format
733	ODBC Standard Rule Failure
799	ODBC Unknown Error
800	NCPDP BFHS Missing
801	NCPDP BFHS Seg ID Error
802	NCPDP BFHS Transmission Type Error
803	NCPDP BFHS Sender ID Missing
804	NCPDP BFHS Batch ID Error
805	NCPDP BFHS File Type Error
806	NCPDP BFHS Receiver ID Missing
807	NCPDP BFDDR Missing
808	NCPDP BFDDR Seg ID Error
809	NCPDP BFDDR Transaction Ref Missing
810	NCPDP BFTR Missing
811	NCPDP BFTR Seg ID Error
812	NCPDP BFTR Batch ID Error
813	NCPDP BFTR Record Count Error
814	NCPDP BFTR Record Count Mismatch Error
815	NCPDP DR Repeating Segment Error
816	NCPDP DR Transaction Code Error
817	NCPDP DR Transaction Count Error
	(Continued on next page)

<b>(Contd) Error Number</b>	<b>Error Generated</b>
818	NCPDP DR Transaction Count Mismatch Error
819	NCPDP DR Header Response Code Error
820	NCPDP DR Transaction Response Code Error
830	NCPDP Null Char
832	NCPDP Invalid Char
833	NCPDP Unexpected EOF
835	NCPDP Unexpected STX
836	NCPDP Unexpected ETX
837	NCPDP Missing Stream Separator 1
838	NCPDP Missing Stream Separator 2
839	NCPDP Unexpected Separator
840	NCPDP Unexpected FS
841	NCPDP Unexpected GS
842	NCPDP Missing GS
845	NCPDP Positional Segment Tag Off
855	NCPDP Delimited Segment Empty
856	NCPDP Delimited Segment Missing First FS
857	NCPDP Delimited Field ID Invalid
858	NCPDP Delimited Field ID Invalid Char
859	NCPDP Delimited Field ID Null Char
860	NCPDP Delimited Field ID Incomplete
861	NCPDP Delimited Field Data Missing
862	NCPDP Delimited Field Data Null Char
863	NCPDP Delimited Field Data Invalid
864	NCPDP Delimited Field Data Invalid Char

(Continued on next page)

<b>(Contd) Error Number</b>	<b>Error Generated</b>
865	NCPDP Delimited Segment ID Invalid
866	NCPDP Delimited Segment ID Invalid Char
867	NCPDP Delimited Segment ID Null Char
868	NCPDP Delimited Segment ID Incomplete
869	NCPDP Delimited Field ID Size
885	NCPDP Batch Nothing After STX
886	NCPDP Batch Unexpected STX
887	NCPDP Batch Missing ETX
888	NCPDP Batch Data File Open
890	NCPDP Data Field String Separator
891	NCPDP Data Field String No Filename
892	NCPDP Data Field String Start Byte
893	NCPDP Data Field String End Byte
894	NCPDP Data Field String Incomplete
895	NCPDP Data Field String Extra Characters
896	NCPDP Data Field Size Incorrect
897	NCPDP Data Field Start Out of Bounds
898	NCPDP Data Field Start Beyond End
899	NCPDP Unknown Error

## concat

---

**Introduction** The **concat** function concatenates a specified number of characters from one string onto the end of another string.

---

**Syntax** `concat(string,string,num_char);`

where:    `string`            = string variable  
          `num_char`        = number of characters from the second string onto the  
                                  end of the first string

**Note**  
You may not use an ActiveX property as the first parameter of the concat function because the length of the property is unknown prior to compilation.

---

**Example**

```
string[20] s1,s2;
s1 = "Sterling";
s2 = "Commerce";

concat(s1,s2,8);

//Concatenate eight characters from string "s2" onto the end of
//string "s1"
// s1 will now contain the value "Sterling Commerce".
```

---

## continue

### Introduction

The **continue** function continues the execution of the innermost loop without processing the statements in the loop that follow the **continue** statement.

### Example

```
integer i
  i = 0;

While i<10 do
Begin
  #Total = Total + 50;
  i = i + 1;

  If #Total > 100000
    Break;
  Else
    Continue;
End

//While the value contained in the variable "i" is less than ten,
//50 will be added to the field Total.
//If the value in the field Total becomes greater than 100000
//before "i" equals 10, break out of the while loop else continue
//processing until "i" equals 10.
```



## count

---

### Introduction

The **count** function counts and returns the number of iterations of a group. An example is as follows:

```
integer i;  
i = count($N1[*]);  
//The [*] is a wildcard that counts the number of iterations of the //  
N1 group.
```

---

# createobject

---

**Introduction** The **createobject** function enables you to create an instance of an ActiveX Automation Server.

---

**Syntax** object = createobject("ProgID");

where: ProgID = programmatic identifier

---

**Example**

```
object ob;
ob = createobject("InternetExplorer.Application");
//Creates an instance of the default interface of an ActiveX
//Automation Server.
//Note:
//The createobject command is more efficient if you use the IID
//instead of the interface name.
```

---

# date

**Introduction** The **date** function converts a string type into a datetime type using a format that you specify. This function allows you to include static characters such as a slash (/), which gives you access to full date support.

**Syntax** Datetime = date("format",string);

where:        datetime = datetime variable (month specified as 1-12)  
               format     = desired date format  
               string     = string variable

**Example**

```
datetime d;
d = date(1995,4,6);
d = date(1995,4,6,12,0);
d = date("%y/%m/%d","95/4/6");
d = date("%y/%m/%d",#strdate);
```

**Format specifiers** This table lists the format specifiers.

Format Specifier	Description
%8	ISO-8601 date format  <b>Format</b> YYYYMMDDTHHMMSS.mmmZ  Four-digit year, two-digit month, two-digit day, T (time) indicator, two-digit hour, two-digit minutes, two-digit seconds in Universal Time (also called Zulu Time or Greenwich Mean Time), Z (Zulu time) indicator (example: 20031209T123000.000Z)  <b>Note</b> This date format cannot be combined with any other format specifier.
%a	Abbreviated weekday name
%A	Full weekday name
%b	Abbreviated month name
%B	Full month name
%d	Day of the month as a decimal number (01 – 31)

(Continued on next page)

<b>(Contd)</b> <b>Format Specifier</b>	<b>Description</b>
%D	ISO-8601 date format (date component only)  <b>Format</b> YYYYMMDDZ  <b>Note</b> This date format cannot be combined with any other format specifier.
%H	Hour in 24-hour format (00 – 23)
%I	Hour in 12-hour format (01– 12)
%j	Day of the year as a decimal number (001 – 366)
%m	Month as a decimal number (01 – 12)
%M	Minute as a decimal number (00 – 59)
%S	Second as a decimal number (00 – 59)
%U	Week of the year as a decimal number, with Sunday as the first day of the week (00 – 51)
%w	Weekday as a decimal number (0 – 6, with Sunday as “0”)
%W	Week of the year as a decimal number, with Monday as the first day of the week (00 – 51)
%y	Year without the century as a decimal number (00 – 99)
%Y	Year with the century as a decimal number
%%	Percent sign

# delete

---

**Introduction** The **delete** function deletes a specified occurrence of a group.

---

**Syntax** `delete($GROUP_NAME[N]);`  
where [n] is the occurrence of the group that you want to delete.

---

**Example** `delete($ILD[2]);`  
`//Deletes the second occurrence of the ILD group.`

---

# deleteobject

---

**Introduction**

The **deleteobject** function enables you to delete an instance of an ActiveX Automation Server. An object must be deleted before the end of the map that uses it. It is more efficient to delete the object immediately on completion, although the Gentran:Server translator will delete the object automatically at the end of the map. Also, if you assign one object to another one, both copies of the object must be deleted for that object to be properly unloaded.

---

**Syntax**

```
deleteobject(object);
```

---

**Example**

```
object ob;  
ob = createobject("InternetExplorer.Application");  
deleteobject(ob);  
//Deletes the instance of the object.
```

---

# empty

---

**Introduction** The **empty** function sets the value of a field in internal storage to null. This function is not the same as setting the value of a field to a zero length string (" ") or to zero.

---

**Syntax** empty(\$GROUP\_NAME[index][index][index]. #FIELD\_NAME)

---

**Example**

```
empty($ILD.#VATC);  
//Set the value of the specified instance of the VATC element  
//to null
```

---

# end

---

## Introduction

Gentran:Server uses conditional logic to test conditions and then, depending on the results of the test, perform different operations. Conditions can be nested to any level. You can use the **if/then** keywords to execute one or more statements conditionally. If you include more than one statement in the body of an **if/then** loop, you must surround the statements with the **begin/end** keywords. If you only use a single statement, you can omit the **begin** and **end**.

## Note

Do not end conditions with a semicolon (;) – this terminating syntax is necessary for statements *only*.

---

## Syntax

```
if condition then
begin
    statement1;
    statement2;
end
```

---



## exec

---

### Introduction

The **exec** function invokes the execution of a batch file or program. The translator waits until the script finishes before continuing with translation. After the script runs and returns a numeric return code, the **exec** function:

- Retrieves the return code
- Returns to translation
- Uses the return code in translation.

### Note

You must set an integer value equal to the return value of the **exec (...)** call for the rule to compile.

---

### Syntax

```
nReturn = exec(string)
```

where:

nReturn = return value

string = string variable or literal value that represents the shell script

---

### Example

```
integer nReturn;  
nReturn = 0;  
nReturn = exec ("c:\addrunm.sh");
```

---

# exist

---

## Introduction

The **exist** function tests to determine if a field is empty (null). It returns a non-zero (true) value if there is data in a specified field in internal storage. If data is not present in the specified field, this function returns a zero (false) value. This function is typically used as a part of a condition.

There are some situations (for example, if the field or element has a "Use Code" standard rule applied to it) when the IF EXIST returns a non-zero (true) value whether or not the condition is true. You can work around this by only using IF EXIST for date- and number-type fields. Be certain that all references to the field which is interrogated are nested within the IF EXIST begin block.

For string-type fields, use the following format:

```
IF Field1 = ""
```

---

## Syntax

```
if exist($GROUP_NAME[index][index][index]. #FIELD_NAME) then
```

---

## Example

```
if exist($ILD.#VATC) then
    #TEMP_FIELD = #VATC;

//Return a non-zero value if the condition is true (data is present
//in the
//specified instance of the VATC element). A zero value is returned
//if the condition is false (no data is present in the specified
//instance of
//the VATC element).
```

---

# fseek

## Introduction

The **fseek** function moves the file pointer to a new location which is a specified number of bytes (offset) from the designated point of origin in the file (the point of origin may be the beginning of the file or relative to either the end of the file or the current position).

To invoke the fseek function against the input file, the value for `current_file` is "0." To invoke the fseek function against the output file, the value for `current_file` is "1." The fseek function is typically only used in conjunction with the `ftell`, `readblock`, and `writeblock` functions.

## Syntax

```
fseek(current_file,offset,origin);
```

where:

<code>current_file</code>	=	0 indicates the input file; 1 indicates the output file
<code>offset</code>	=	The position to which the file pointer is moved relative to the origin.
<code>origin</code>	=	The keyword depends on the starting location: <code>begin</code> = start at beginning of the file <code>end</code> = start at the end of the file <code>current</code> = start at the current position in the file

## Example

```
string[1024]temp_buffer;
Integer Position;
Position = ftell(0);
while readblock(temp_buffer) do
begin
    if left(temp_buffer,3) = "IEA" then
        begin
            fseek(0,Position,begin);
            break;
        end
    writeblock(temp_buffer);
    Position = ftell(0);
end
//Read a segment from input file and place in temp_buffer. Look for
//"IEA" segment tag. If found, reset file pointer to where it was
//before the "IEA" segment was read. Write contents of temp_buffer
//to output file. Set "Position" = the current file pointer
//position.
```

# ftell

---

**Introduction**

The **ftell** function obtains the current position of the file pointer and returns it as an integer. To invoke the **ftell** function against the input file, the value for `current_file` is "0." To invoke the **ftell** function against the output file, the value for `current_file` is "1." The **fseek** function is typically only used in conjunction with the **fseek**, **readblock**, and **writeblock** functions.

---

**Syntax**

```
numeric_variable = ftell(current_file);
```

---

**Example**

```
string[1024]temp_buffer;
Integer Position;
Position = ftell(0);
while readblock(temp_buffer) do
begin
    if left(temp_buffer,3) = "IEA" then
        begin
            fseek(0,Position,begin);
            break;
        end
    writeblock(temp_buffer);
    Position = ftell(0);
end
//Read a segment from input file and place in temp_buffer. Look for
//"IEA" segment tag. If found, reset file pointer to where it was
//before the "IEA" segment was read. Write contents of temp_buffer
//to output file. Set "Position" = the current file pointer
//position.
```

---

# get

---

**Introduction** The `get` function enables you to access individual components of a datetime variable.

---

**Syntax** `integer_variable = get datetime_component (datetime_variable);`

where:

<code>integer_variable</code>	=	integer variable
<code>datetime_component</code>	=	individual component of the datetime variable
<code>datetime_variable</code>	=	datetime variable of which you want to access a component part

---

**Example**

```
integer temp_days;
integer temp_hours;
datetime d;
temp_days = 0;
temp_hours = 0;
d = '12/25/2001 12:15:30'; //A fields value in the map can be
// assigned to the datetime variable "d"
//or a hard coded value can be assigned.

temp_days = get days (d);
temp_hours = get hours (d);

//Accesses the days from the datetime variable "d" and loads into
//variable " temp_days ". Accesses the hours from the datetime
//variable "d" and loads into variable "temp_hours".
```

---

# getiid

---

**Introduction**

The **getiid** function enables you to obtain the unique identifier for an interface, by using the string-character name of the interface to return the globally unique identifier that is used by software to run the interface.

---

**Syntax**

This creates an instance of the default interface of an ActiveX Automation Server.

```
string_variable = getiid("ProgID");
```

This looks up the InterfaceID (IID) of an interface.

```
string_variable = getiid("ProgID", "Interface_Name or {Interface ID}");
```

---

**Example**

```
object ob;  
string[50] iid;  
iid = getiid("InternetExplorer.Application", "IWebBrowser2");  
ob = createobject("InternetExplorer.Application", iid);  
ob.Visible = 1;  
//Displays the Internet Explorer on the desktop by setting a  
//property value in an ActiveX Automation Server.
```

---

## if then...else

### Introduction

The **if**, **then**, and **else** keywords allow the use of conditional logic. Gentran:Server uses conditional logic to test conditions and then, depending on the results of the test, perform different operations. Conditions can be nested to any level. You can use the **if/then** keywords to execute one or more statements conditionally. The condition is typically a comparison, but it can be any expression that concludes with a numeric value. Gentran:Server interprets the value as either *true* or *false*. The system interprets a zero value as false and a nonzero value as true.

Gentran:Server evaluates the **if/then** condition, and if it is true, the system executes all the statements that follow the **then** keyword. If the condition is false, none of the statements following **then** are executed.

You can use the **else** keyword in conjunction with **if/then** to define several blocks of statements, one of which is executed. Gentran:Server tests the first **if/then** condition. If the condition is false, the system proceeds to test each sequential condition until it finds one that is true. The system executes the corresponding block of statements for the true condition. If none of the **if/then** conditions are true, the system executes the statements following the **else** keyword.

### Note

Do not end conditions with a semicolon (;) – this terminating syntax is necessary for statements *only*.

### Example

```
if condition then
begin
    statement1;
    statement2;
end
else
begin
    statement3;
    statement4;
end
```

# index

---

**Introduction** The **index** function enables you to determine which instance of a particular loop the translator is currently accessing.

---




**Syntax** `index(integer_variable);`

where:      `integer_variable` = integer variable that indicates the hierarchical level for which you want to determine the loop count

---

**Example**

```
index(1)
//This extended rule is located on the ILD group in the diagram
//below. This determines the current loop count for the first level
//(ODD) in the hierarchical structure.
```

```
INVOIC -M- INVOICE DETAILS 
      ODD -M- 999999 
      ILD -M- 999999 
```



# insert

## Introduction

The **insert** function allows information in the database tables to be updated.

## Syntax

```
insert into tablename [ (fieldlist) ] [ (valuelist) ];
```

where:      tablename      DivisionLookup | PartnerLookup | DivisionLocation  
                                  PartnerLocation | DivisionXref | PartnerXref

          fieldlist           ( fieldname [ , fieldname ] )

          fieldname         The name of one of the fields in the table.

          valuelist         ( String [ , String ] )

## Notes

- A list of the tables is available later in this chapter.
- The fieldlist lists one or more fieldnames to which data is to be added.
- The fieldnames can be listed in any order.
- The valuelist must be in the same order as the field list.

## Example

```
updateStatus = update PartnerLookup
set Description = "Lookup Update
Test",Text1="Text1Updated",Text2="Text2Updated",Text3="Text3Updated",T
ext4="Text4Updated"
where TableName = "PartLkp" and Item = "1";

if updateStatus = 1 then
begin
  messagebox("Update PartnerLookup: Record Not Found,Attempting
  Insert...",1);
  insertStatus = insert into PartnerLookup( PartnerKEY, TableName,
  Item,Description,Text1,Text2,Text3,Text4)
  values ("PETZONE","PartLkp","1","Lookup Insert
  Test","Text1","Text2","Text3","Text4");
  if insertStatus = 0 then
  begin
    messagebox("Insert PartnerLookup: Failed",1);
  end
end

if updateStatus = 2 then
begin
  messagebox("Update PartnerLookup Failed with OtherError",1);
end
//Example assumes that a lookup table named "PartLkp" has been
//created.
```

# left

---

**Introduction**

The **left** function extracts a specified number of character from the left side of a string variable or field and returns the result as a string.

The syntax is as follows:

```
string_variable = left(string_variable,num_char)
```

where: num\_char = integer variable

**Note**

You may not use an ActiveX property as the first parameter of the left function because the length of the property is unknown prior to compilation.

---

**Example**

```
string [25]name;  
string [5]temp_variable;  
name = "Acme Shipping Company"  
temp_variable = left(name,4);  
// "temp_variable" would contain "Acme"
```

---

# len

---

**Introduction** The **len** function is a numerical function that counts and returns the number of characters in a string. The numerical functions enable you to convert one data type to another.

---

**Syntax** `number_char = len(string);`

where:      `num_char = integer variable`

---

**Example**

```
integer a;  
a = 0;  
a = len("hello");  
// "a" contains the value 5
```

---

## messagebox

### Introduction

The **messagebox** function enables you to display a message box for which you have designated the format and content. You can specify the number and type of buttons on the message box, the message icon (e.g., hand, question mark, exclamation point, or asterisk), and the message displayed. You can also issue a return value based on the chosen action.

### Note

Only String values can be displayed in a message box.

### Syntax

```
messagebox("message",defined_number)
```

where:    message            =    message string  
          defined\_number =    defined number of the desired buttons plus the  
                                     defined number of the desired icon (if used)

### Example

```
if messagebox("Do you really want to delete this object?",36) = 6
begin
.
.
.
end

//Displays a message box with the given string as the message, Yes
//and No buttons (4) and a question mark icon (32). The number and
//type of buttons (4) plus the icon (32) equals the defined_number
//(36). If the user clicks the Yes button (return value "6"), the
//statements in the begin/end loop are executed.
```

### Defined\_ numbers

This table lists the defined\_numbers for the button and icon types.

Defined_Number	Button or Icon Types
0	<b>OK</b> button only
1	<b>OK</b> and <b>Cancel</b> buttons
4	<b>Yes</b> and <b>No</b> buttons
16	Icon Hand
32	Icon Question Mark
48	Icon Exclamation Point
64	Icon Asterisk

**Message box  
return values**

This table lists the message box return values.

<b>Return Value</b>	<b>Action Selected</b>
1	<b>OK</b> selected
2	<b>Cancel</b> selected
6	<b>Yes</b> selected
7	<b>No</b> selected

# mid

---

**Introduction**

The **mid** function extracts from a specified position in a string, either to the end of the string or for a specified number of characters and returns the resultant string. This function is zero-based.

---

**Syntax**

```
string_variable = mid(string_variable,start_pos,num_char)
```

where:      start\_pos    =   integer variable  
             num\_char    =   integer variable

**Note**

You may not use an ActiveX property as the first parameter of the mid function because the length of the property is unknown prior to compilation.

---

**Example**

```
string [25]name;  
string [10]temp_variable;  
name = "Acme Shipping Company"  
temp_variable = mid(name,6,8);  
// "temp_variable" will contain "Shipping"
```

---

# ntoa

---

**Introduction** The **ntoa** function is a numerical function that converts integers and real numbers into strings. The numerical functions enable you to convert one data type to another.

---

**Syntax** `string = ntoa(integer/real,string);`

where:      `integer/real` = integer or real number variable  
             `string`         = string variable

---

**Example**

```
Integer b;  
string[8] s;  
b = 5.5;  
ntoa(b,s);  
//The variable "s" contains the string "5.5".
```

---

# param

---

## Introduction

The **param** function is used to read the value of the PARAM(n) variable. The extended rule allows you to reference values that have been passed into the translator via the command line. The -u switch can be used during the invocation of the translator to pass values in so they can be referenced by an extended rule. Typically, the param extended rule is not used by maps that are running within the Gentran:Server for Windows environment because Gentran:Server does not use the -u switch when invoking the translator. However, if you are invoking the translator to perform translation outside of Gentran:Server (tx32 -f <inputfile> <templatefile> <outputfile> <reportfile>), you have the option to pass values into the translator via the -u switch and reference those values using the param extended rule.

---

## Syntax

param(integer\_PARAM\_number);  
where: interger\_PARAM\_number = the number of the pre-defined variable

---

## Example

For example, if you invoke TX32.EXE in the following manner:

```
tx32.exe -f input.txt 850.tpl output.txt out.rpt -u  
"PurchaseOrderNumber" -u 12345 -u 500.25
```

Then you could write an extended rule in the \\GENSRVNT\Tutorial\Pet\_850.map that looks like the following:

```
string [32] strDescription;  
string [32] strPONbr;  
string [32] strTotalCost;  
real nTotalCost;  
  
strDescription = param(0);  
strPONbr = param(1);  
strTotalCost = param(2);  
nTotalCost = aton(strTotalCost);  
  
if nTotalCost > 200 then  
begin  
    messagebox("Get approval from boss",0);  
end
```

You can run this rule from any scope in your map (e.g., pre-session, post-session, group onbegin, field, etc.).

---



# queryobject

---

**Introduction** The **queryobject** function is used to request a different interface on an existing object.

---

**Syntax** `object2 = queryobject(object1, "{IID}");`

where:

- `object2` = is the result that contains the requested interface to the object
- `IID` = is the interface identifier of the requested interface
- `object1` = an existing object

---

**Example**

```
object ob, ob2;
ob = createobject("InternetExplorer.Application");
ob2 = queryobject(ob, "{EAB22AC1-30C1-11CF-A7EB-Å0000C05BAE0B}");
//Uses the Interface ID of the desired interface to obtain another
//(different) interface of the existing object (object1).
```

---

# readblock

---

## Introduction

The **readblock** function reads a block of data (segment or record) from the input file and places it into the argument of a string variable. The **readblock** and **writeblock** functions are used in conjunction with each other to pass a block of data from the input file to the output file without compliance checking or testing for proper EDI syntax. Together these functions provide a more efficient alternative of using “wildcard” segments, which are typically implemented in build and break maps.

---

## Syntax

`readblock(string_variable);`

## Note

You may not use an ActiveX property as the first parameter of the **readblock** function because the length of the property is unknown prior to compilation.

---

## Example

```
while readblock(temp_buffer) do
begin
  if left(temp_buffer,3) = "IEA" then
  begin
    fseek(0,Position,begin);
    break;
  end
  writeblock(temp_buffer);
  Position = ftell(0);
end
//Read segment from input file and place in temp_buffer. Look for
//"IEA" segment tag. If found, reset file pointer to where it was
//before the "IEA" segment was read. Write contents of temp_buffer
//to output file. Set "Position" = the current file pointer
//position.
```

---

# readbytes

---

## Introduction

The **readbytes** function reads a number of bytes from the input file. This function is used in conjunction with the writebytes function. Used together, the readbytes and writebytes function provide an efficient method of passing data through a map if the data does not need to be compliance checked or altered in any way.

Readbytes is similar to the readblock function, but readblock only works with entire blocks (i.e., an entire segment or record), and readbytes works with any quantity of data, whether it is smaller or larger than a block.

The readbytes function uses two parameters, first the string variable into which the data being read will be stored, and second the number of bytes to read.

---

## Syntax

```
readbytes(read_from_buffer, num_bytes);
```

where: read\_from\_buffer= string variable into which the data being read will be stored  
num\_bytes = integer value representing the number of bytes to read from the input file.

## Note

Readbytes returns the number of bytes it was actually able to read.

---

## Example

```
string [1024] tempBuffer;  
while readbytes(tempBuffer,1024) do  
begin  
    writebytes(tempBuffer,1024);  
End  
//Read 1024 bytes from input file and place in string variable  
//named tempBuffer.  
  
writebytes("^0D^0A",2);  
//Appends a CRLF to the end of the output file.
```

---

# right

---

**Introduction** The **right** function extracts a specified number of characters from the right side of a string variable or field.

---

**Syntax** `string_variable = right(string_variable,num_char)`

where:      `num_char`    =    integer variable

**Note**

You may not use an ActiveX property as the first parameter of the right function because the length of the property is unknown prior to compilation.

---

**Example**

```
string [25]name;  
string [10]temp_variable;  
name = "Acme Shipping Company"  
temp_variable = right(name,7);  
// "temp_variable" would contain "Company"
```

---

# select

---

**Introduction**

The **select** function allows information to be retrieved from the database tables. Only the tables and fields available in the Select standard rule are available for the select extended rule. In the command syntax “expression” and “receiverlist” can be a string field, string variable, or string literal. It is important to note that the table and field names for the select extended rule are slightly different than those depicted in the standard rule. A listing of these table and field names can be found at the end of this chapter.

---

**Syntax**

select *fieldname* into *receiverlist* from *tablename* where *key* = *expression* [and *key* = *expression*];

---

**Example**

```
string[10]temp_var1,temp_var2,temp_var3;
select Text1, Text2, Text3 into temp_var1, temp_var2, temp_var3 Åfrom
Partnerlookup where Tablename = "Test" and Item ="widgit"
//Example assumes that a Lookup table named "Test" has been
//created.
```

---

# set

---

**Introduction** The **set** function enables you to define individual components of a datetime variable.

---

**Syntax** `set datetime_component (datetime_variable, integer_variable);`

where:

<code>datetime_component</code>	= individual component of the datetime variable
<code>datetime_variable</code>	= datetime variable of which you want to access a component part
<code>integer_variable</code>	= integer variable

---

**Example**

```
integer temp_days;
datetime d;
temp_days = 15;
d = '12/25/2001'; //A fields value in the map can be
// assigned to the datetime variable "d"
//or a hard coded value can be assigned.

set days (d,temp_days);

//Defines the days of the datetime variable "d" from variable
//"temp_days".
```

---

# strdate

**Introduction** The **strdate** function converts a datetime type into a string using a format that you specify. This function allows you to include static characters such as a slash (/), which gives you access to full date support.

**Syntax** `strdate(datetime,"format",string);`

where:

- `datetime` = datetime variable (month specified as 1-12)
- `format` = desired date format
- `string` = string variable

**Example**

```
datetime d;
string[8] s;

d = #DateField;

strdate(d, "%y/%m/%d", s);

//Converts a datetime variable into an eight character string in
//the format "year/month/day".
```

**Format specifiers**

This table lists the format specifiers.

Format Specifier	Description
%8	ISO-8601 date format  <b>Format</b> YYYYMMDDTHHMMSS.mmmZ  Four-digit year, two-digit month, two-digit day, T (time) indicator, two-digit hour, two-digit minutes, two-digit seconds in Universal Time (also called Zulu Time or Greenwich Mean Time), Z (Zulu time) indicator (example: 20031209T123000.000Z)  <b>Note</b> This date format cannot be combined with any other format specifier.
%a	Abbreviated weekday name
%A	Full weekday name
%b	Abbreviated month name
%B	Full month name

(Continued on next page)

<b>(Contd) Format Specifier</b>	<b>Description</b>
%d	Day of the month as a decimal number (01 – 31)
%H	Hour in 24-hour format (00 – 23)
%I	Hour in 12-hour format (01– 12)
%j	Day of the year as a decimal number (001 – 366)
%m	Month as a decimal number (01 – 12)
%M	Minute as a decimal number (00 – 59)
%S	Second as a decimal number (00 – 59)
%U	Week of the year as a decimal number, with Sunday as the first day of the week (00 – 51)
%w	Weekday as a decimal number (0 – 6, with Sunday as “0”)
%W	Week of the year as a decimal number, with Monday as the first day of the week (00 – 51)
%y	Year without the century as a decimal number (00 – 99)
%Y	Year with the century as a decimal number
%%	Percent sign



## strstr

---

### Introduction

The **strstr** function finds a substring inside a string. This function returns the position of the first instance of the designated substring within the specified string. If this function does not find the specified substring in the string, it returns a value of -1. This function is zero-based.

---

### Syntax

```
integer = strstr("string","substring");
```

where:   integer   =   integer variable  
         string     =   string  
         substring =   part of the string

---

### Example

```
integer d;  
d = 0;  
  
d = strstr("mississippi","is");  
  
//Finds the first instance of the substring "is" inside the string  
//"mississippi" and returns the position of that first substring.
```

---

# unreadblock

---

## Introduction

The **unreadblock** function provides a method of moving the input file-pointer back one block (a block of data is equivalent to one EDI segment or one positional record). This function unread the block of data that was just processed by the readblock function.

### Note

Unreadblock should only be used in conjunction with the readblock function. The unreadblock function will only unread the most recent block of data processed.

Unreadblock is provided as an alternative to using the fseek and ftell functions, and is the preferred method of moving the file pointer back one block of data. Unreadblock allows the translator to correctly track the number of bytes read and number of segments read during the translation process by moving the file-pointer back and decrementing the segment and byte counts accordingly.

---

## Syntax

unreadblock();

---

## Example

```
string[1024] tempBuffer;

while readblock(tempBuffer) do
begin
    if left(tempBuffer,3) = "IEA" then
    begin
        unreadblock();
        break;
    end
    writeblock(tempBuffer);
end

//Read segment from input file and place in tempBuffer variable.
//Look for the "IEA" segment tag and, if found, reset the file-
//pointer to where it was located before the "IEA" segment was
//read. Write contents of tempBuffer to output file.
```

---

# update

## Introduction

The **update** function allows information in the database tables to be updated. This function is similar to the Update standard rule, except that it provides more flexibility. Only the tables and fields available in the Update standard rule are available for the update extended rule. In the command syntax “expression” can be a string field, string variable, or string literal. It is important to note that the table and field names for the update extended rule are slightly different than those depicted in the standard rule. A listing of these table and field names can be found at the end of this chapter.

This table describes how the update function is supported in Gentran:Server.

IF the map/form is of type...	THEN the update function is...
Export	Only valid on the input side of the map.
Import	Valid on the input or output side of the map.
Break (Interchange, Group, or Transaction Set)	Only valid on the input side of the map.
Build (Interchange, Group, or Transaction Set)	Only valid on the input side of the map.
Print	Not supported.
Screen entry	Not supported.

## Syntax

update *tablename* set *fieldname* =expression [ *fieldname* =expression ] where *key* = *expression* [ and *key* = *expression* ];

## Example

```
string[15]temp_variable;
temp_variable = "5 Pound Hammer";
update PartnerXref set Text1 = temp_variable where MyItem = " and
^Tablename = "Tools";
//Example assumes that a cross-reference table named "Tools" has
//been created.
```

## while...do

### Introduction

The **while...do** function executes a statement repeatedly until the specified termination condition evaluates to zero. The system tests the terminating condition *before* each iteration of the loop, so a **while** loop executes zero or more times depending on the value of the termination expression.

### Note

Do not end conditions with a semicolon (;) – this terminating syntax is necessary for statements *only*.

### Example

```
integer i
  i = 0;

While i<10 do
Begin
  #Total = Total + 50;
  i = i + 1;

  If #Total > 100000
    Break;
  Else
    Continue;
End

//While "i" is less than 10 execute the loop.
```

# winexec

## Introduction

The **winexec** function enables you to execute another program while running the translator. This program is executed asynchronously. You specify the program and determine how you want the program window displayed. You can also return an error code, if desired. If the error code is greater than 32, the program ran without errors. If the error code is less than 32, the program did not run because of an error. If the error code is "0," the system is out of memory. If the error code is "2," you didn't specify a file name. The error code is *not* the return value from the program you executed.

### Note

- If you specify a program on another machine or in another domain, you must have the appropriate permission to access the specified folder.
- If translation is executing from an unattended process control command, the user ID under which that service is running must have the appropriate permission to access the specified file.

## Syntax

```
winexec("program",window_display)
```

where:    program            =    executable program name string (if necessary, including UNC or direct file path)  
           window\_display =    number that indicates how you want the program window displayed

## Example

```
winexec("program.exe", 3)

//Exits Gentran:Server and executes the "program.exe"
//program asynchronously. The system displays the program window
//maximized (3).
```

## Window display numbers

The window\_display numbers that control the appearance of the program window are as follows (you must use the *number* to indicate how you want the program window displayed, not the window\_display value):

Number	Window_Display	Definition
0	SW_HIDE	Hides the window and activates another window.
1	SW_SHOWNORMAL	Activates and displays a window. If the window is minimized or maximized, it is restored to the original size and position. This flag should be specified when displaying a window for the first time.  (Continued on next page)

<b>(Contd) Number</b>	<b>Window_Display</b>	<b>Definition</b>
1	SW_NORMAL	Activates and displays a window in the original size and position.
2	SW_SHOWMINIMIZED	Activates the window and displays it as a minimized windows.
3	SW_SHOWMAXIMIZED	Activates the window and displays it as a maximized window.
3	SW_MAXIMIZE	Maximizes the window.
4	SW_SHOWNOACTIVATE	Displays the window in its most recent size and position, but does not activate it (the current active window remains active).
5	SW_SHOW	Activates the window and displays it in its current size and position.
6	SW_MINIMIZE	Minimizes the window.
7	SW_SHOWMINNOACTIVE	Displays the window as a minimized window without activating it (the current active window remains active).
8	SW_SHOWNA	Displays the window in its most recent size and position without activating it (the current active window remains active).
9	SW_RESTORE	Activates and displays the window. If the window was minimized or maximized, it is restored to its original size and position.
10	SW_SHOWDEFAULT	Activates the window and allows Windows to determine the size and position.

# writeblock

---

**Introduction**

The **writeblock** function writes the data contained in the argument of a string variable to the output file. The **readblock** and **writeblock** functions are used in conjunction with each other to pass a block of data from the input file to the output file without compliance checking or testing for proper EDI syntax. Together these functions provide a more efficient alternative of using “wildcard” segments, which are typically implemented in build and break maps.

---

**Syntax**

```
writeblock(string_variable);
```

---

**Example**

```
while readblock(temp_buffer) do
begin
  if left(temp_buffer,3) = "IEA" then
  begin
    fseek(0,Position,begin);
    break;
  end
  writeblock(temp_buffer);
  Position = ftell(0);
end
//Read segment from input file and place in temp_buffer. Look for
//"IEA" segment tag. If found, reset file pointer to where it was
//before the "IEA" segment was read. Write contents of temp_buffer
//to output file. Set "Position" = the current file pointer
//position.
```

---

# writebytes

## Introduction

The **writebytes** function writes a specified number of bytes to the output file. This function is used in conjunction with the readbytes function. Used together, the readbytes and writebytes function provide an efficient method of passing data through a map if the data does not need to be compliance checked or altered in any way.

### Note

The system does not append a segment terminator to the end of the string value written to the output file.

Writebytes is similar to the writeblock function, but writeblock only works with entire blocks (i.e., an entire segment or record), and writebytes works with any quantity of data, whether it is smaller or larger than a block.

The writebytes function uses two parameters, first the string variable containing the data to be written to the output file, and second the number of bytes to write.

## Syntax

```
writebytes(write_to_buffer, num_bytes);
```

where: write\_to\_buffer = string variable containing the data to be written to the output file  
num\_bytes = integer value representing the number of bytes to read from the write\_to\_buffer variable and write to the output file

### Note

If hex values are used in constants, use \0x instead of ^.

## Example

```
string [1024] temp_buffer;
while readbytes(temp_buffer,1024) do
begin
    writebytes(temp_buffer,1024);
End
writebytes("^0D^0A",2);

//Read 1024 bytes from input file and place in string variable Snamed
temp_buffer.
//Appends a CRLF after the while loop finishes executing.
```



## Select and Update Available Options

---

### Introduction

The following section contains the database table names and the associated field names that are available when using the **select** and **update** extended rules. Additional keys are also available for certain tables. Where applicable, a description of the key follows the field name.

#### Note

When you use extended rules to reference a Gentran:Server database table, the syntax used is different from the actual table name. The appropriate referencing syntax is outlined for each table in this section.

#### Example

To refer to the Document\_tb in an extended rule, you would reference Document.

---

### Document

These are the field names that are available when using the select or update extended rules on the Document Table (Document\_tb).

#### Note

Refer to this table as **Document**.

- ▶ TESTMODE
  - ▶ AGENCY
  - ▶ VERSION
  - ▶ TRANSACTIONSETID
  - ▶ RELEASE
  - ▶ DOCUMENTTYPE
  - ▶ REFERENCEDATA
  - ▶ DOCUMENTNAME
  - ▶ APPFIELD1
  - ▶ APPFIELD2
  - ▶ APPFIELD3
  - ▶ APPFIELD4
  - ▶ APPFIELD5
  - ▶ APPFIELD6
  - ▶ DOCUMENTPARTNERKEY
  - ▶ CONTROLNUMBER
  - ▶ PARTNERKEY
-

---

**Partner** These are the field names that are available when using the select or update extended rules for a non-division partner on the Partner Table (Partner\_tb).

**Note**

Refer to this table as **Partner**.

- ▶ PARTNERNAME
- ▶ EDICODE
- ▶ APPLICATIONPARTNERKEY

These are the additional keys that are available for the Partner table.

- ▶ PARTNERKEY
- ▶ ALTERNATEKEY (refers to Application Key)
- ▶ APPLICATIONPARTNERKEY (refers to Application Key)

---

**Division** These are the field names that are available when using the select or update extended rules to reference the division partner on the Partner Table (Partner\_tb).

**Note**

Refer to this table as **Division**.

- ▶ PARTNERNAME
- ▶ EDICODE
- ▶ APPLICATIONPARTNERKEY

---

**PartnerLocation** These are the field names that are available when using the select or update extended rules for a non-division location on the Location Table (Location\_tb).

**Note**

Refer to this table as **PartnerLocation**.

- ▶ CONTACTNAME
  - ▶ NAME
  - ▶ ADDRESS1
  - ▶ ADDRESS2
  - ▶ ADDRESS3
  - ▶ CITY
  - ▶ STATE
  - ▶ ZIP
  - ▶ COUNTRY
  - ▶ TELEPHONE
  - ▶ PRIMARYREFCODE
  - ▶ SECONDARYREFCODE
  - ▶ FAX
-

---

**DivisionLocation**

These are the field names that are available when using the select or update extended rules to reference a division location on the Location Table (Location\_tb).

**Note**

Refer to this table as **DivisionLocation**.

- ▶ CONTACTNAME
- ▶ NAME
- ▶ ADDRESS1
- ▶ ADDRESS2
- ▶ ADDRESS3
- ▶ CITY
- ▶ STATE
- ▶ ZIP
- ▶ COUNTRY
- ▶ TELEPHONE
- ▶ PRIMARYREFCODE
- ▶ SECONDARYREFCODE
- ▶ FAX

---

**PartnerLookup**

These are the field names that are available when using the select or update extended rules for a non-division lookup on the Lookup Table (Lookup\_tb).

**Note**

Refer to this table as **PartnerLookup**.

- ▶ ITEM
- ▶ DESCRIPTION
- ▶ TEXT1
- ▶ TEXT2
- ▶ TEXT3
- ▶ TEXT4

This is an additional key that is available for the PartnerLookup table.

- ▶ TABLENAME
-

---

**DivisionLookup**

These are the field names that are available when using the select or update extended rules to reference a division lookup on the Lookup Table (Lookup\_tb).

**Note**

Refer to this table as **DivisionLookup**.

- ▶ ITEM
- ▶ DESCRIPTION
- ▶ TEXT1
- ▶ TEXT2
- ▶ TEXT3
- ▶ TEXT4

This is an additional key that is available for the DivisionLookup table.

- ▶ TABLENAME
- 

**PartnerXref**

These are the field names that are available when using the select or update extended rules for a non-division cross-reference on the Cross-reference Table (CrossReference\_tb).

**Note**

Refer to this table as **PartnerXref**.

- ▶ MYITEM
- ▶ PARTNERITEM
- ▶ DESCRIPTION
- ▶ TEXT1
- ▶ TEXT2
- ▶ TEXT3
- ▶ TEXT4

This is an additional key that is available for the PartnerXref table.

- ▶ TABLENAME
-

---

**DivisionXref**

These are the field names that are available when using the select or update extended rules to reference a division cross-reference on the Cross-reference Table (CrossReference\_tb).

**Note**

Refer to this table as **DivisionXref**.

- ▶ MYITEM
- ▶ PARTNERITEM
- ▶ DESCRIPTION
- ▶ TEXT1
- ▶ TEXT2
- ▶ TEXT3
- ▶ TEXT4

This is an additional key that is available for the DivisionXref table.

- ▶ TABLENAME
- 

**Interchange**

These are the field names that are available when using the select or update extended rules on the Interchange Table (Interchange\_tb).

**Note**

Refer to this table as **Interchange**.

- ▶ TESTMODE
  - ▶ CONTROLNUMBER
  - ▶ VERSION
  - ▶ APPFIELD1
  - ▶ APPFIELD2
  - ▶ APPFIELD3
  - ▶ APPFIELD4
  - ▶ APPFIELD5
  - ▶ APPFIELD6
  - ▶ AGENCY
  - ▶ PARTNERKEY
-

---

**Group** These are the field names that are available when using the select or update extended rules on the Group Table (Group\_tb).

**Note**

Refer to this table as **Group**.

- ▶ TESTMODE
- ▶ CONTROLNUMBER
- ▶ FUNCTIONALGROUPID
- ▶ VERSION
- ▶ APPFIELD1
- ▶ APPFIELD2
- ▶ APPFIELD3
- ▶ APPFIELD4
- ▶ APPFIELD5
- ▶ APPFIELD6
- ▶ AGENCY

---

**GenericEnvelope  
Segment**

These are the field names that are available when using the select extended rule on the Generic Envelope Segment Table (GenericEnvelopeSegment\_tb).

**Note**

Refer to this table as **GenericEnvelopeSegment**.

- ▶ CONTROLNUMBER
- ▶ SUBCOUNT
- ▶ FIELD1
- ▶ FIELD2
- ▶ FIELD3
- ▶ FIELD4
- ▶ FIELD5
- ▶ FIELD6
- ▶ FIELD7
- ▶ FIELD8
- ▶ FIELD9
- ▶ FIELD10
- ▶ FIELD11
- ▶ FIELD12
- ▶ FIELD13
- ▶ FIELD14

---

(Continued on next page)

---

**GenericEnvelope  
Segment  
(contd)**

- ▶ FIELD15
  - ▶ FIELD16
  - ▶ FIELD17
  - ▶ FIELD18
  - ▶ FIELD19
  - ▶ FIELD20
  - ▶ FIELD21
  - ▶ FIELD22
  - ▶ FIELD23
  - ▶ FIELD24
  - ▶ FIELD25
  - ▶ FIELD26
  - ▶ FIELD27
  - ▶ FIELD28
  - ▶ FIELD29
  - ▶ FIELD30
-





---

# Customizing Screen Entry and Print Forms

---

<b>Contents</b>	<ul style="list-style-type: none"> <li>▶ Introduction . . . . . 7 - 1</li> </ul>
<b>Screen Entry and Print Forms</b>	<b>7 - 2</b>
<ul style="list-style-type: none"> <li>▶ Customizing Field Labels . . . . . 7 - 2</li> <li>▶ Hiding Fields . . . . . 7 - 3</li> </ul>	
<b>Screen Entry Forms</b>	<b>7 - 4</b>
<ul style="list-style-type: none"> <li>▶ Setting an Initial Value for a Field . . . . . 7 - 4</li> <li>▶ Using a Constant Value for a Field . . . . . 7 - 7</li> <li>▶ Creating Definitions for a List Box . . . . . 7 - 9</li> <li>▶ Creating Help Text for Fields and List Boxes . . . . . 7 - 11</li> <li>▶ Creating Help Text for Frames . . . . . 7 - 13</li> <li>▶ Formatting Entry Fields . . . . . 7 - 14</li> <li>▶ Formatting Display-Only (Non-Editable) Fields . . . . . 7 - 15</li> </ul>	

---

## Introduction

---

**In this chapter** This chapter describes how to customize screen entry and print forms.

---

**Customizing Screen Entry Forms** On a screen entry form, you can customize the way data is arranged in fields, or the format of numbers, dates, or times. You can even prevent fields in the screen entry form from being displayed when the translation object (compiled form) is used.

---

**Customizing Print Forms** After you select the segments, groups, and elements for the print form, you can customize the format of the EDI data as it is generated by Gentran:Server.

---

# Screen Entry and Print Forms

## Customizing Field Labels

### Introduction

Field labels identify the contents of the fields included in a screen entry or print form. The system uses the element name from the EDI standard as the default to label a field. You can customize the default field label to be any name you choose.

### Recommendation

We recommend that you label every field. If you decide *not* to label a field, follow the steps below, but delete the existing field label. For example, you might decide not to label the fields of a company address to be entered in a standard address format, because it is obviously an address. In this instance, the name of the company is on the first line; the street address is on the second line; and the city, state, and zip code are on the third line and do not need to be labeled.

### Procedure

Use this procedure to customize a field label.

Step	Action	
1	Double-click the element for which you want to customize the field label.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).	
2	Use this table to determine your next step.	
	<b>IF you want to customize a field label on a...</b>	<b>THEN...</b>
	Screen Entry form,	Click the <b>Display</b> tab to access Display options. Proceed to the next step.
	Print form,	Click the <b>Print</b> tab to access Print options.
3	In the Please enter a label for this field, modify the text to customize the label.	
4	Click <b>OK</b> to save the new label and exit the Element Properties dialog box.	
5	Click <b>Generate Layout</b> on the Main Toolbar to refresh the form display.	

# Hiding Fields

**Introduction** Usually, you want all EDI data on a screen entry or print form to be displayed. However, there may be times when you want to prevent the data in a field from being displayed.

**Example scenarios** You might want to hide a field when it contains a qualifier that is explained by the label of the associated data field. In this instance, you might set a constant for the field (so that the field always contains the appropriate qualifier without it having to be entered each time), then hide the field. You also might want to hide a field if it is a mandatory segment that must be kept activated, but does not need to be displayed or printed.

**Procedure** Use this procedure to hide a field in a screen entry or print form.

Step	Action	
1	Double-click the element that you want to hide on the form.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).	
2	Use this table to determine your next step.	
3	<b>IF you want to hide a field on a...</b>	<b>THEN...</b>
	Screen Entry form,	<ul style="list-style-type: none"> <li>▶ Click the <b>Display</b> tab to access display options.</li> <li>▶ Select the field should not appear on the form option to indicate that this field should be hidden.</li> <li>▶ Proceed to the next Step.</li> </ul>
	Print form,	<ul style="list-style-type: none"> <li>▶ Click the <b>Print</b> tab to access print options.</li> <li>▶ Clear the <b>Check here to include this field in print</b> check box to turn the option off.</li> <li>▶ Proceed to the next Step.</li> </ul>
4	Click <b>OK</b> to exit the Element Properties dialog box.	
5	Click the <b>Generate Layout</b> icon to refresh the form display.	

## Screen Entry Forms

### Setting an Initial Value for a Field

---

**Introduction**

If you typically need the same value in a field each time that field is used, you might want to set an initial value for the field. After you set an initial value, that value displays in the field every time that field is used in the form.

For a screen entry form, the user can override the initial value for a field if you want to enter data that is different from the initial value. If you do *not* want the user to be able to override the field value, then you should use a constant value for the field, instead of an initial value.

**Example**

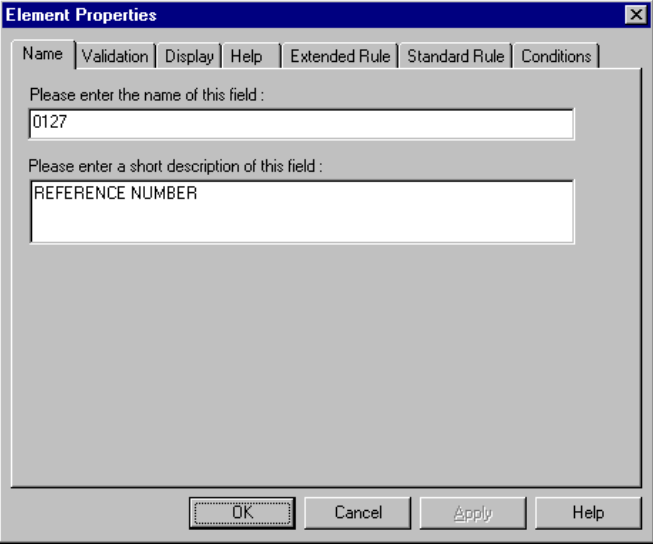
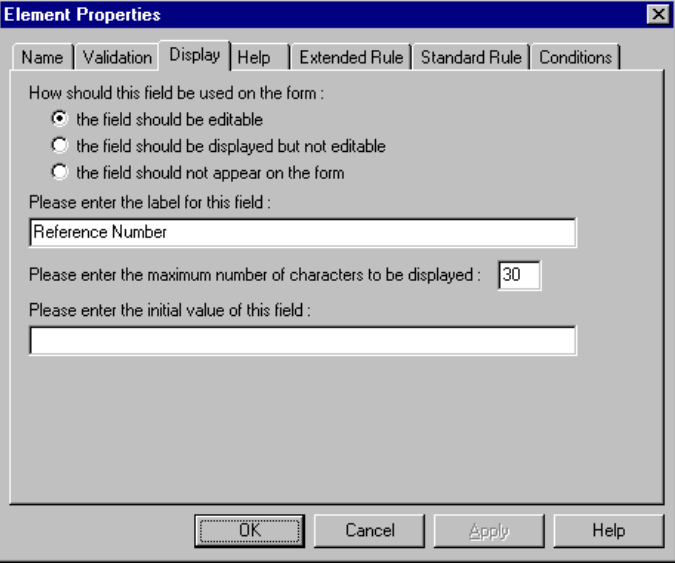
If you know that one person at your company is nearly always the person who should be contacted by telephone, you might set an initial value for a Telephone Contact field on an invoice, by setting the telephone contact's name as the initial value. There are occasions when someone else at your company is the contact person, however, so you want to be able to override the initial value in the Telephone Contact field when those exceptions occur.

**Reference**

See *Using a Constant Value for a Field* on page 7 - 7 for more information.

---

**Procedure** Use this procedure to set an initial value for a field.

Step	Action
1	<p>Double-click the element for which you want to set an initial value.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Display</b> tab.</p> <p><b>System Response</b> The system displays display options.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Step</b>	<b>Action</b>
3	In the initial value box, type the value that you want to be the default displayed for this field.
4	Click <b>OK</b> to save the initial value for this field and exit the Element Properties dialog box.

---

## Using a Constant Value for a Field

---

### Introduction

If you always use the same value in a field each time that field is used, you might want to use a constant value for the field so the user does not have to enter the same information each time. After you set the constant value, that value displays in the field every time the field is used.

### CAUTION

**YOU CANNOT OVERRIDE THE CONSTANT VALUE FOR A FIELD. IF YOU WANT TO BE ABLE TO OVERRIDE THE FIELD VALUE, THEN YOU SHOULD SET AN INITIAL VALUE FOR THE FIELD, INSTEAD OF A CONSTANT VALUE.**

### Example

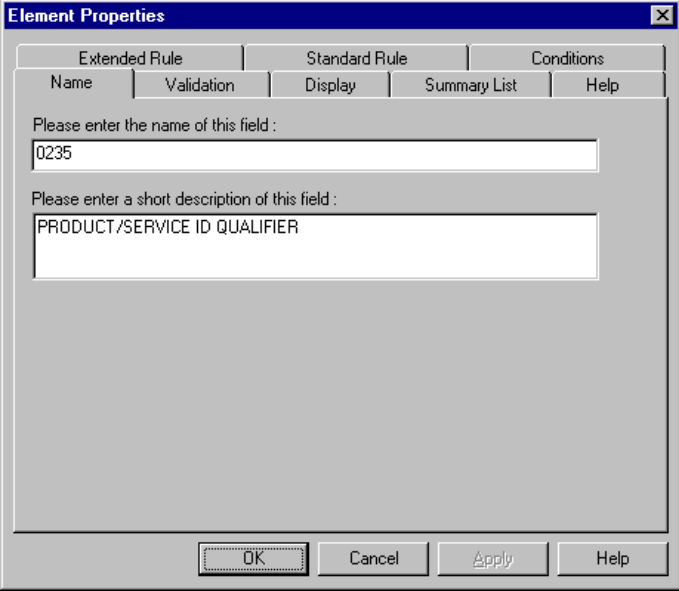
You might set a constant value for a qualifier field on an invoice if you know that you never need to override the value in the field. In this instance, you would probably hide the field, as well.

### Reference

See *Setting an Initial Value for a Field* on page 7 - 4 and *Hiding Fields* on page 7 - 3 for more information.

---

**Procedure** Use this procedure to use a constant value for an element.

Step	Action
1	<p>Double-click the element for which you want to set a constant value.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Standard Rule</b> tab to access standard rule options.
3	From the standard rule list, select <b>Use Constant</b>
4	<p>From the constant list, select the constant that you want to use. If the constant that you want to use for this field is not in the list, you need to create or edit a constant.</p> <p><b>Reference</b> Please see <i>Editing Literal Constants</i> on page 5 - 43 and <i>Editing Literal Constants</i> on page 5 - 43 for more information.</p>
5	Click <b>OK</b> and the data that was stored in the selected constant is loaded in the field.



# Creating Definitions for a List Box

## Introduction

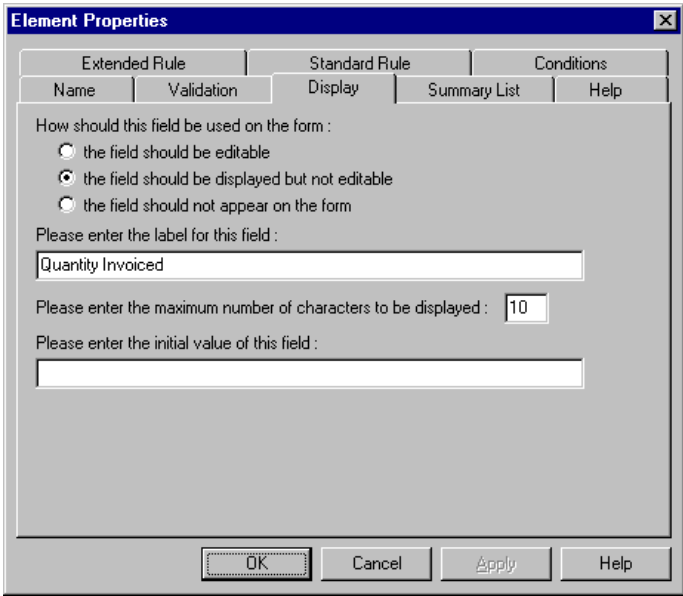
When Gentran:Server identifies a group in a transaction set, it creates a separate frame on the Screen Entry form to contain the data within the group. It also creates a list box on the parent frame to allow the user access to the items in the group. To identify the item in the group, at least one field needs to be displayed in the list box. Each set of data can be displayed as a column in the list box. For example, you might have a list box of items on a purchase order. Within the Items list box, you can set up three columns of data:

- Quantity
- Description
- Unit Price

To define the list box columns, you would perform the steps below for the Quantity field, then repeat the steps for the Description and Unit Price fields.

## Procedure

Use this procedure to create a definition for a list box.

Step	Action
1	Double-click the element for which you want to define a list box.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).
2	Click the <b>Display</b> tab to access display options.  

(Continued on next page)



<b>(Contd) Step</b>	<b>Action</b>
3	Select the display option to indicate that this field should be displayed as not-editable on the form.
4	Click the <b>Summary List</b> tab to access the summary list options.
5	Select <b>Check here if this field should be displayed</b> on summary lists check box.
6	From the column list, select the sequence of this column in the list box. For example, if this is to be the first column in the list box, select <b>1</b> .
7	<p>In the width box, type the width of this column in number of characters allowed.</p> <p><b>Note</b> Ensure that the column width is long enough to accommodate the average length (in characters) of the values expected for this column of data.</p>
8	Click <b>OK</b> to save the list box definition and exit the Element Properties dialog box.
9	Repeat <b>Steps 1</b> through <b>8</b> for each element in the group that contains data you want to define in the list box.
10	To label the columns of data set up in the list box, you must add column headings (static text) to the form.

## Creating Help Text for Fields and List Boxes

### Introduction

You can create context-sensitive Help text for a field or list box to aid screen entry translation object users in entering the information required (in the Document Editor). The Help text displays during data entry when the a user selects a field or an entry in the list box and presses **F1**.

### How to create Help text for fields

Use this procedure to create Help text for a field.

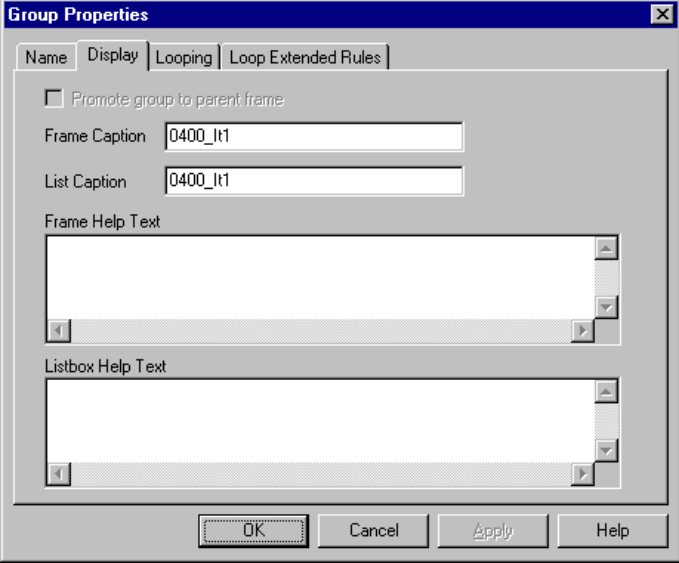
Step	Action
1	Double-click the element for which you want to create Help text.  <b>System Response</b> The system displays the Element Properties dialog box (Name tab).
2	Select the <b>Help</b> tab to access the help options.
3	In the help text list, type the Help text.  <b>Note</b> Gentran:Server does not automatically wrap Help text when users access context-sensitive help in Document Editor. Therefore, you may want to insert a hard return each time the Help text you are typing reaches the right side of the help text list, rather than allowing the list to scroll to the right.
4	Click <b>OK</b> to save the Help text and exit the Element Properties dialog box.

### How to create Help text for list boxes

List boxes contain information from selected elements in a repeating group. The information from each element displays as a column in the list box.

Use this procedure to create Help text for a list box.

Step	Action
1	Right-click the group for which you want to create Help text to access the shortcut menu. Select <b>Properties...</b> from the shortcut menu.  <b>System Response</b> The system displays the Group Properties dialog box (Name tab).  (Continued on next page)

<b>(Contd) Step</b>	<b>Action</b>
2	<p>Click the <b>Display</b> tab.</p> <p><b>System Response</b> The system displays the options.</p> 
3	In the Help Text list, type the Help text.
4	Click <b>OK</b> to save the Help text and exit the Group Properties dialog box.

## Creating Help Text for Frames

---

**Introduction** The Help text displays during data entry when the a user clicks FRAME? or selects Frame Help from the Document Editor Help menu, when the frame is currently in focus.

---

**Procedure** Use this procedure to create Help text for a frame.

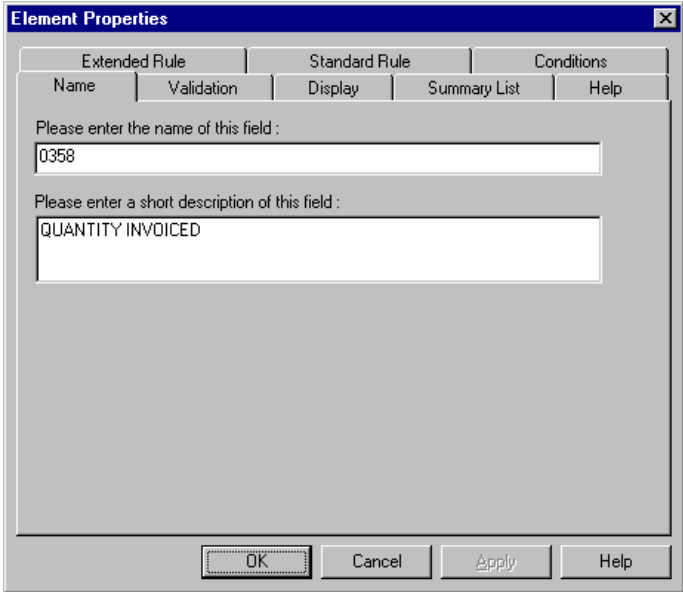
Step	Action
1	Right-click the group for which you want to create Help text to access the shortcut menu.
2	Select <b>Properties</b> from the shortcut menu to access either the File Format Properties dialog box or the Group Properties dialog box (depending on the level of the group).
3	Click the <b>Display</b> tab to access display options.
4	In the Frame Help Text list, type the Help text. For groups, the Frame Help Text list is only be active if the Max Usage (found on the Group Properties dialog box, Looping tab) is greater than "1."
5	Click <b>OK</b> to save the Help text and exit the Properties dialog box.

---

## Formatting Entry Fields

**Introduction** An entry field is a field that can be edited. When you create a form, Gentran:Server makes every field an entry field by default. For entry fields, you can change the field type to Display (the field is display-only) or Hidden (the field is not displayed at all).

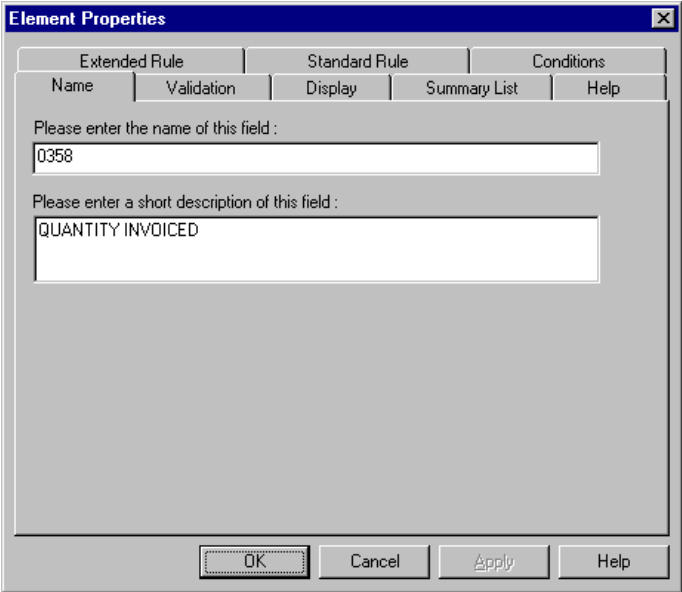
**Procedure** Use this procedure to change a display-only or hidden field to an entry field.

Step	Action
1	<p>Double-click the element for which you want to define a list box.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	Click the <b>Display</b> tab to access display options.
3	Select the editable option to indicate that this field can be edited.
4	Click <b>OK</b> to exit the Element Properties dialog box.
5	Click the <b>Generate Layout</b> button on the Main Toolbar to refresh the display of the screen entry form.

## Formatting Display-Only (Non-Editable) Fields

**Introduction** When you create a form, Gentran:Server makes every field an entry field by default. You can change the default to Display if you want a field to be display-only (non-editable). Data can be viewed but not edited in display-only fields.

**Procedure** Use this procedure to format a display-only field.

Step	Action
1	<p>Double-click the element for which you want to define a list box.</p> <p><b>System Response</b> The system displays the Element Properties dialog box (Name tab).</p> 
2	<p>Click the <b>Display</b> tab.</p> <p><b>System Response</b> The system displays the display options.</p>
3	<p>Select the display option to indicate that this field should be displayed as not-editable on the form.</p>
4	<p>Click <b>OK</b> to exit the Element Properties dialog box.</p>
5	<p>Click the <b>Generate Layout</b> button on the Main Toolbar to refresh the display of the screen entry form.</p>





---

# Formatting Screen Entry and Print Forms

---

<b>Contents</b>	<ul style="list-style-type: none"> <li>▶ Introduction . . . . . 8 - 2</li> </ul>	
	<b>Selecting and Grouping Fields . . . . .</b>	<b>8 - 3</b>
	<ul style="list-style-type: none"> <li>▶ Selecting One or More Fields or Field Labels . . . . . 8 - 3</li> <li>▶ Grouping Fields and Field Labels . . . . . 8 - 5</li> </ul>	
	<b>Resizing Fields . . . . .</b>	<b>8 - 6</b>
	<ul style="list-style-type: none"> <li>▶ Overview . . . . . 8 - 6</li> <li>▶ Using Size to Length . . . . . 8 - 7</li> <li>▶ Resizing a Field Manually . . . . . 8 - 8</li> <li>▶ Resizing Groups of Fields Manually . . . . . 8 - 9</li> </ul>	
	<b>Arranging Fields . . . . .</b>	<b>8 - 10</b>
	<ul style="list-style-type: none"> <li>▶ Overview . . . . . 8 - 10</li> <li>▶ Changing Grid Settings . . . . . 8 - 11</li> <li>▶ Using Alignment Lines . . . . . 8 - 13</li> <li>▶ Moving Fields and Field Labels . . . . . 8 - 14</li> <li>▶ Aligning Fields and Field Labels . . . . . 8 - 15</li> <li>▶ Spacing Fields and Field Labels . . . . . 8 - 17</li> <li>▶ Resizing and Positioning a Frame . . . . . 8 - 18</li> <li>▶ Resizing Drop-Down Combo Boxes . . . . . 8 - 19</li> <li>▶ Setting Tab Sequences . . . . . 8 - 21</li> <li>▶ Adding and Positioning Static Text . . . . . 8 - 22</li> <li>▶ Modifying Frame Titles . . . . . 8 - 23</li> <li>▶ Modifying List Box Names . . . . . 8 - 24</li> <li>▶ Adding Column Headings to List Boxes . . . . . 8 - 25</li> </ul>	

---

## Introduction

**In this chapter**

---

After you customize screen entry and print forms, you can format the organization and size of the fields in the form. This chapter explains how to format screen entry and print forms.

---

# Selecting and Grouping Fields

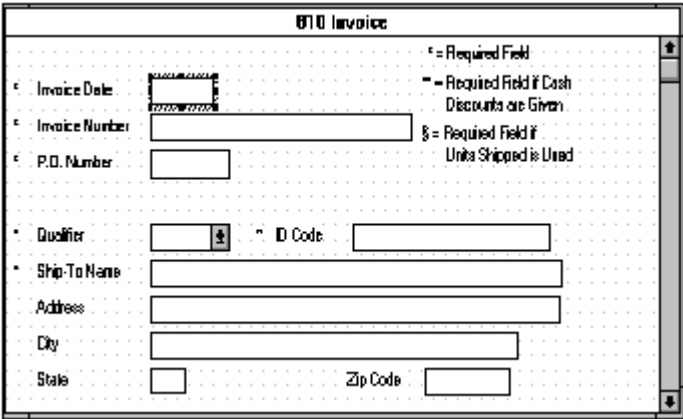
## Selecting One or More Fields or Field Labels

**Introduction** You must know how to select one or more items (fields and field labels) in a form to be able to move, resize, align, or change the spacing between them.

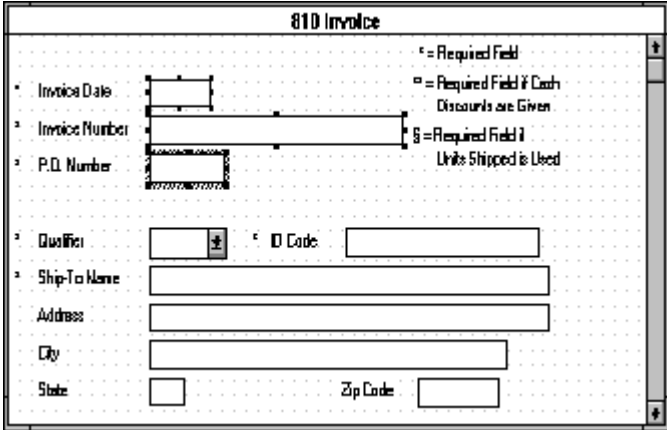
**Reference items** The last item you select is outlined by diagonal lines. This item is called the *reference*. All other items you selected are now outlined by a solid line.

When you arrange a group of items, the reference item is significant because the alignment, size, and spacing between all other items in the group is relative to the alignment, size, and spacing of the reference item.

**Procedure** Use this procedure to select a field or a field label.

Step	Action
1	<p>In the Layout Window, click anywhere on the field or field label.</p> <p><b>Note</b> The Lock Labels icon on the Main toolbar enables you to specify that when you click a field (in the Layout Window), its corresponding label is selected at the same time.</p> <p><b>System Response</b> The field or field label is outlined by diagonal lines. The selected field also displays <i>handles</i>, which mark the corners and sides.</p>  <p style="text-align: right; color: green;">(Continued on next page)</p>

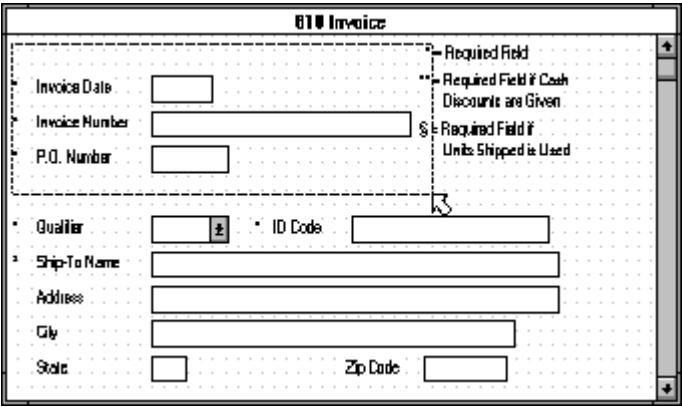


<b>(Contd) Step</b>	<b>Action</b>
2	Do you want to select more fields or field labels? <ul style="list-style-type: none"> <li>▶ If yes, proceed to the next Step.</li> <li>▶ If no, continue with another formatting task.</li> </ul>
3	Press <b>CTRL</b> and click another field or field label.
4	Repeat these steps for all other fields and field labels in the form you want to select. <p style="color: red; font-weight: bold; margin-top: 10px;">CAUTION</p> <p style="color: red; font-weight: bold; margin-top: 5px;">WHEN YOU SELECT TWO OR MORE ITEMS, MAKE SURE THE ITEM YOU WANT AS THE REFERENCE IS THE LAST ITEM YOU SELECT.</p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;">  </div>
5	Did you select any fields by mistake? <ul style="list-style-type: none"> <li>▶ If yes, press <b>CTRL</b>, and click the item you want to deselect.</li> </ul> <p style="font-weight: bold; margin-top: 10px;">Note</p> <p>If the item you deselect is the reference, the item that was selected immediately prior to the last item (the reference) becomes the new reference.</p> <ul style="list-style-type: none"> <li>▶ If no, continue with another formatting task.</li> </ul>

# Grouping Fields and Field Labels

**Introduction** You can group fields and field labels that are in the same area of a form.

**Procedure** Use this procedure to group fields or field labels.

Step	Action
<p>1</p>	<p>In the Layout Window, click and drag the mouse across the form items that you want to group.</p> <p><b>Note</b> As you drag the mouse, the grouped area is designated by dotted lines similar to the following illustration.</p> 
<p>2</p>	<p>Release the mouse button.</p> <p><b>System Response</b> All items <i>completely</i> contained within the selection area are selected. The last item in the group is the reference. All the selected items are now a group, and can be moved together, as well as resized or spaced relative to the reference.</p>

# Resizing Fields

## Overview

---

**Introduction**

When you generate a form, each field is automatically sized to the length specified in the Display Length field on the Display Properties dialog box. If you change the display length of a field after the form is generated for the first time, you must resize that field in the Layout Window.

---

**Resizing fields**

You can resize a field in one of these two ways:

- Use the Size to Length function.
- Use the mouse to reshape the field manually.

In addition, you can resize groups of fields or field labels so that they are all the same size.

**Note**

Resizing the field graphically maintains the integrity of the form.

---

# Using Size to Length

**Introduction** You can resize one or more fields according to the display length specified on the Display Properties dialog box.

**Procedure** Use this procedure to resize a field or group of fields with the Size to Length function.

Step	Action
1	In the Layout Window, select the form fields that you want to resize.  <b>Reference</b> See <i>Grouping Fields and Field Labels</i> on page 8 - 5 for more information.
2	Select <b>Size To Length</b> from the <u>D</u> isplay menu to resize according to the display length specified on the Display Properties dialog box for each selected item.



## Resizing a Field Manually

**Procedure** Use this procedure to manually resize one field.

Step	Action
1	In the Layout Window, select a form field.
2	Click and drag one of the handles. <div data-bbox="641 588 1279 1039" style="border: 1px solid black; padding: 5px; margin-top: 10px;"> </div>



# Resizing Groups of Fields Manually

**Introduction** You can resize several fields at the same time if you want them all to be the same size.

**Example**

You might want to resize a group of fields that are similar in size to give the form a more consistent appearance.

**Reference**

See *Grouping Fields and Field Labels* on page 8 - 5 for more information.

**Procedure** Use this procedure to resize a group of fields manually.

Step	Action
1	In the Layout Window, select the form fields you want to resize.  <b>Note</b> Ensure that the field that is to be used as the reference for resizing all other selected fields is selected last.
2	Select <b>Size Equally</b> from the <u>D</u> isplay menu.  <b>System Response</b> The system equally resizes all selected items. The reference is used as the basis for determining the size of all other selected items.



# Arranging Fields

## Overview

---

**Introduction**

To arrange the fields and field labels in a screen entry or print form, you must move, align, and space the fields and field labels in the Layout Window.

---

# Changing Grid Settings

## Introduction

Gentran:Server provides a grid, indicated by the dots in the background of the form, that you can use to assist you in formatting the layout of a screen entry or print form. The dots on the grid form horizontal and vertical lines that guide you in aligning and spacing fields and field labels. The dots on the grid are evenly spaced, with the spacing between them measured in pixels. The grid is not printed (for print forms) – it is only displayed in the background of the template to aid you in arranging fields and field labels.

## Snap to Grid Controls

Sometimes it is useful to have items snap to the grid when you move them. With the “Snap controls to the alignment grid” feature turned on, as you move, align, or space items, they snap to the lines of the grid. This feature can help you align items with one another or evenly space them.

## Procedure

Use this procedure to change the grid settings.

Step	Action
1	Select <b>P</b> references from the <b>O</b> ptions menu.  <b>System Response</b> The system displays the Preferences dialog box (the Tree tab displays by default). The Preferences dialog box is a property sheet that enables you to set global defaults for Gentran:Server.
2	Click the <b>L</b> ayout tab.  <b>System Response</b> The system displays the layout options.
3	Do you want the grid to display? • If yes, click <b>S</b> how alignment grid. • If no, continue with step 6.
4	Do you want the fields and field labels to snap to the grid? • If yes, click the <b>S</b> nap controls to the alignment grid check box to turn the option on. Proceed to the next Step.  <b>CAUTION</b> <b>IF YOU ARE USING ALIGNMENT LINES, YOU SHOULD NOT USE THE “SNAP CONTROLS TO THE ALIGNMENT GRID” FUNCTION, AS THIS HINDERS THE EFFECTIVENESS OF THE ALIGNMENT LINES.</b>  (Continued on next page)



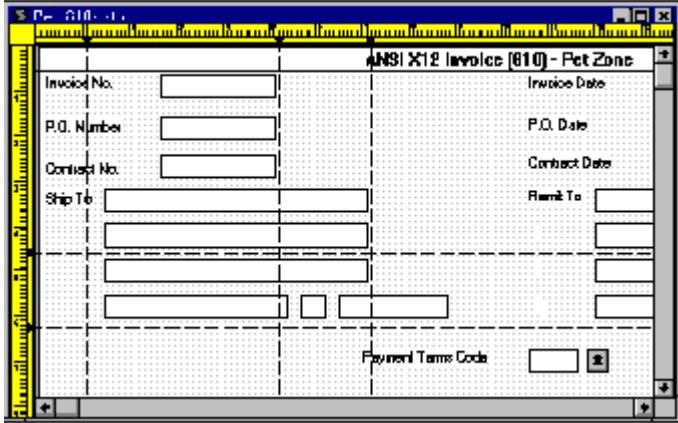
<b>(Contd) Step</b>	<b>Action</b>
5	Do you want to change the spacing between the dots on the grid? <ul style="list-style-type: none"><li>▶ If yes, type the number of pixels in the Grid spacing box. The default setting is five pixels. Increasing the number of pixels increases the spacing between the dots; decreasing the number of pixels decreases the spacing between the dots.</li><li>▶ If no, proceed to the next Step.</li></ul>
6	Click <b>OK</b> to exit the Preferences dialog box.

---

# Using Alignment Lines

**Introduction** Gentran:Server enables you to create alignment lines (lines that guide you in aligning the components of the Layout Window).

**Procedure** Use the procedure to display, move, or remove alignment lines.

Step	Action
1	<p>In the Layout Window, double-click the vertical or horizontal ruler just below the graduation where you want the alignment line to be located.</p> <p><b>WARNING</b>  <b>IF YOU ARE USING ALIGNMENT LINES, YOU SHOULD NOT USE THE SNAP CONTROLS TO GRID FUNCTION (ON THE GRID SETTINGS DIALOG BOX), AS THIS HINDERS THE EFFECTIVENESS OF THE ALIGNMENT LINES.</b></p> <p><b>System Response</b>                      The the system displays the alignment line.</p> 
2	<p>Do you want to move the alignment line to another position?</p> <ul style="list-style-type: none"> <li>▶ If yes, click and drag the black triangle that connects the alignment line to the ruler.</li> <li>▶ If no, proceed to the next Step.</li> </ul>
3	<p>Do you want to remove an alignment line?</p> <ul style="list-style-type: none"> <li>▶ If yes, double-click the black triangle that connects the alignment line to the ruler. Proceed to another formatting task.</li> <li>▶ If no, proceed to another formatting task.</li> </ul>



## Moving Fields and Field Labels

---

### Introduction

You can move fields or field labels from their default positions to other locations in a form. You can move either individual items or groups of items.

### Recommendation

If you intend to resize fields, we recommend that you do so before moving them, since the arrangement of the fields might depend on the size of the fields.

### Example

You might be able to arrange more fields in a row if the fields are smaller. Therefore, you should resize the fields before you arrange them, so you can accurately determine how many fields can fit in a row.

### Reference

See *Using Size to Length* on page 8 - 7 or *Resizing a Field Manually* on page 8 - 8 for more information about resizing fields.

---

### Procedure

Use this procedure to move a field or field label.

Step	Action
1	<p data-bbox="630 989 1416 1045">In the Layout Window, click the field that you want to move and drag it to the new location in the form.</p> <p data-bbox="630 1073 1416 1161"><b>Note</b> You can use Lock Label icon on the Main toolbar to move a label with its associated field.</p> <p data-bbox="630 1199 1416 1287">If you are moving a group of selected items, all items in the group stay in position relative to each other as they are moved from one location to the next.</p>

---

# Aligning Fields and Field Labels

## Introduction

When fields and field labels are created, they are automatically left-aligned in the screen entry or print form, with one field per line. However, you can align particular fields and field labels with one another in different ways to change the way they are arranged in the form.

If you intend to resize or move fields or field labels, we recommend that you do so before you align them. If you align fields first, then move or resize them, the fields might move out of alignment.

## Reference

See *Using Size to Length* on page 8 - 7 or *Resizing a Field Manually* on page 8 - 8 for more information about resizing fields. See *Moving Fields and Field Labels* on page 8 - 14 for more information about moving fields or field labels.

## Align Controls Options

You use Align Controls to specify how you want items on your form to appear. This table describes valid Align Control options.

Align Control	Description
Top	Aligns all selected items with the top of the reference item.
Bottom	Aligns all selected items with the bottom of the reference item.
Left	Aligns all selected items with the left side of the reference item.
Right	Aligns all selected items with the right side of the reference item.
Horizontal Centre	Aligns all selected items horizontally in the frame.
Vertical Centre	Aligns all selected items vertically in the frame.

## Procedure

Use this procedure to align fields and field labels.

Step	Action
1	<p>Select two or more fields that you want to align in a form.</p> <p><b>Note</b> Ensure that the field or field label to be used as the reference for aligning all other selected items is selected last.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Step</b>	<b>Action</b>
2	Select <b>Align Controls</b> from the Display menu.
3	From the Align Controls cascading menu, select how you want the selected items to be aligned.  <b>CAUTION</b> <b>USE CAUTION IN SELECTING THE ALIGNMENT METHOD TO ENSURE THAT ITEMS DO NOT OVERLAY EACH OTHER. USE TOP, BOTTOM, OR HORIZONTAL CENTRE ALIGNMENT FOR ALIGNING A HORIZONTAL ROW OF ITEMS; USE LEFT, RIGHT, OR VERTICAL CENTRE ALIGNMENT FOR ALIGNING A VERTICAL COLUMN OF ITEMS.</b>

---



# Spacing Fields and Field Labels

## Introduction

When fields and field labels are resized and moved, the spacing between them can become uneven. You can change the spacing between items to give the form a more consistent appearance.

### Recommendation

If you intend to resize or move fields or field labels, we recommend that you do so before you space them. If you space fields first, then move or resize them, the spacing between the fields might change.

### Reference

See *Using Size to Length* on page 8 - 7 or *Resizing a Field Manually* on page 8 - 8 for more information about resizing fields. See *Moving Fields and Field Labels* on page 8 - 14 for more information about moving fields or field labels.

## Spacing Options

You use the Space Evenly cascading menu to specify how far apart fields display vertically or horizontally on the form. This table describes valid spacing options.

Method	Description
Across	Evenly spaces all selected items with the next selected item to the left or right (horizontal spacing).
Down	Evenly spaces all selected items with the next selected item above or below (vertical spacing).

## Procedure

Use this procedure to evenly space several fields and field labels.

Step	Action
1	Select three or more items that you want to be evenly spaced in a form. <b>Note</b> Ensure that the field or field label to be used as the reference for evenly spacing all other selected items is selected last.
2	Select <b>Space Evenly</b> from the Display menu.
3	From the Space Evenly cascading menu, select the type of spacing you want for the selected items.

## Resizing and Positioning a Frame

---

### Introduction

You can define how a frame is sized and positioned in the Document Editor by resizing and positioning that frame in the Forms Integration subsystem. When a form is created, the default location for frames is the upper left corner of the Layout Window. When a screen entry translation object is opened in Document Editor, its frame is sized and positioned the same as it was when the source form was last compiled.

### Example

If you want the frame to be centered on the user's desktop, then you must center the frame in the Layout Window in the Forms Integration subsystem before compiling it.

---


### Warning

When you resize the frames of a screen entry form, you must resize based on the resolution of the target monitor (i.e., the monitor on which the end user views the translation object). For example, if you size the frames of a screen entry form on a Super VGA monitor, the compiled translation object may not fit in the display area of a VGA monitor.

---

### Procedure

Use this procedure to resize and position a frame.

Step	Action
1	In the Layout Window, point to the frame border, when the pointer becomes a  drag the border to resize.
2	To position the frame in the Layout Window, click and drag the frame title to the new position in the Layout Window.

---

# Resizing Drop-Down Combo Boxes

## Introduction

A drop-down combo box consists of a text box and a drop-down list. When the combo box is closed, the text box and the drop-down arrow are displayed. When the combo box is opened, the available field values (and a scroll bar, if there are more values than can fit in the drop-down list) are displayed. You can resize a drop-down combo box in a screen entry form to make the size appropriate for the field values contained in the list.

## Resizing Drop-Down Combo Box Tips

Please remember the following tips about drop-down combo boxes:

- When you change the width of the combo box, the width of the text box changes. (The width is the same whether the drop-down combo box is opened or closed.)
- When you change the height of the combo box, the height of the drop-down list is changed. When closed, the drop-down combo box is long enough to contain one item. When opened, the drop-down combo box is long enough to contain several items.
- The width of the drop-down combo box should be appropriate to the width of the values available in the list. You should make the width a few spaces wider than the longest value available. For example, if the values available are two-character codes, the combo box does not need to be twenty characters wide. However, if the values in the list consist of several words or a long string of characters, then the combo box should be wide enough for users to read the longest field value.
- The height of the drop-down combo box should be appropriate to the number of values available in the list. For example, if there are just four values available, you might make the drop-down combo box long enough to contain all of the values, so that users do not have to scroll through a short list. However, if there are one hundred values available, you might make the drop-down combo box long enough to view up to eight values at once.

### Note

The scroll bar is automatically added to the drop-down combo box when the number of available values exceeds the vertical space provided in the combo box.

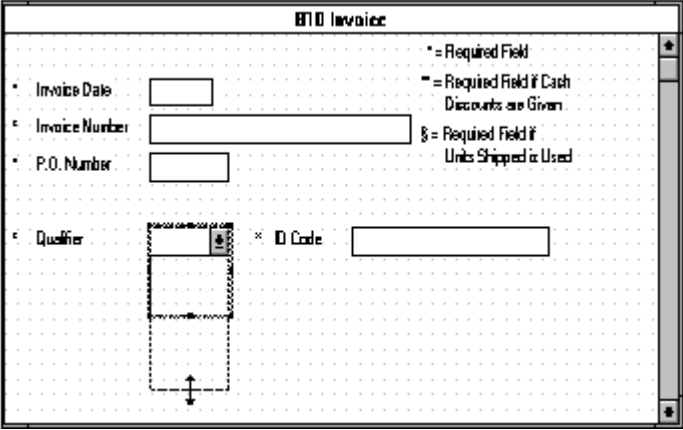
## Procedure

Use this procedure to resize a drop-down combo box.

Step	Action
1	In the Layout Window, click the drop-down combo box you want to resize in the form.

(Continued on next page)



(Contd) Step	Action
2	Drag a handle of the combo box.  A screenshot of a software window titled "BTD Invoice". The window contains several input fields: "Invoice Date", "Invoice Number", "P.O. Number", "Qualifier", and "ID Code". A legend on the right side explains symbols: an asterisk (*) for "Required Field", a double asterisk (**) for "Required Field if Cash Discounts are Given", and a dollar sign (\$) for "Required Field if Units Shipped is Used". The "Qualifier" field is a drop-down menu, and a dashed box with a vertical double-headed arrow indicates it is being resized.

# Setting Tab Sequences

## Introduction

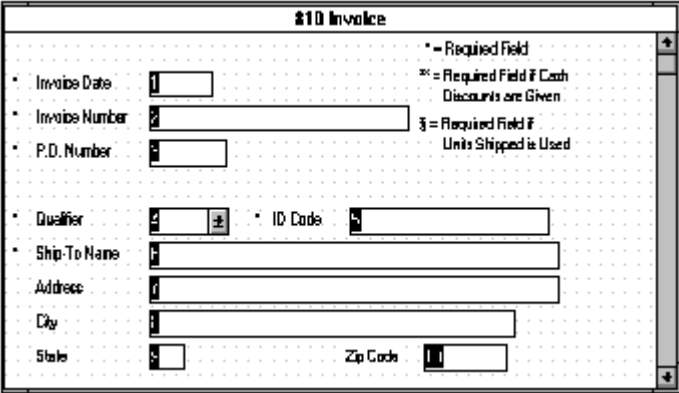
The tab sequence is the order in which the cursor moves from field to field in a screen entry form when a user presses **TAB**. When you create a screen entry form, the system automatically positions the fields and sets the tab sequence for those fields. However, you usually want to modify the tab sequence of the fields to meet your screen entry requirements.

### Note

You can set tab sequences for screen entry forms *only*.

## Procedure

Use this procedure to set the tab sequence of fields in a screen entry form.

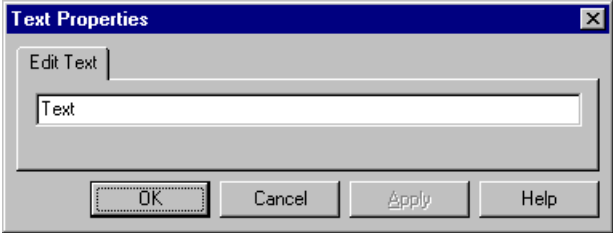
Step	Action
1	<p>Select <b>Set Tab Order</b> from the <b>D</b>isplay menu. Numbers are displayed in the fields in the form to indicate the current tab sequence.</p> 
2	<p>Click the fields sequentially, in the order in which you want the tab sequence to be set. For example, if you want a field that is currently set as third in the tab sequence to become first in the tab sequence, then click that field first. The field that was first in the tab sequence is now the second, the second is now the third, etc. After you have clicked all the fields, the tab sequence numbers are no longer displayed in the fields.</p> <p><b>Note</b> If you want to change the tab sequence for some of the fields without having to click them all, then select Set Tab Order from the <b>D</b>isplay menu when you have finished setting the tab sequence for the fields you want to change. The tab sequence numbers are no longer be displayed.</p>



## Adding and Positioning Static Text

**Introduction** Static text is text that is always included in the screen entry or print form. Examples of static text include legends and column headings. Field labels, which usually accompany fields to identify the contents of the fields, are not considered static text.

**Procedure** Use this procedure to add and position static text to a screen entry or print form.

Step	Action
1	Click <b>Add Text</b> from the <u>D</u> isplay menu. Text is added to the current frame, displayed in the upper left corner of the frame. If the translation object is larger than the frame (there is a scroll bar on the right side of the frame), the static text displays in the upper left corner of the part of the translation object that displays, not at the top of the translation object.
2	Type the static text in the new selected text field.  <b>System Response</b> The system displays the Text Properties dialog box. 
3	Click <b>OK</b> to add the static text.
4	To move the static text, click and drag it to a new location in the frame.

# Modifying Frame Titles

## Introduction

The system automatically titles the frames in a screen entry form when the form is created, based on the name of the file, segments, or groups. You can modify the frame title of any repeating group (EDI file, segment, or group) in a screen entry form. For example, if you use several purchase order forms, each for a different trading partner, you might give each purchase order form a slightly different title to distinguish one form from the others.

## Procedure

Use this procedure to modify a frame title.

Step	Action
1	Highlight the form component (EDI file, segment, or group) of the frame title you want to modify.
2	Select <b>Properties</b> from the Edit menu to display the appropriate dialog box for the form component you selected. <ul style="list-style-type: none"> <li>▶ If you select the EDI file, the system displays the File Format Properties dialog box.</li> <li>▶ If you select a segment, the system displays the Segment Properties dialog box.</li> <li>▶ If you select a group, the system displays the Group Properties dialog box.</li> </ul>
3	Click the <b>Display</b> tab to access display options.
4	In the Frame Caption box, type the new frame title.
5	Click <b>OK</b> to display the new frame title in the appropriate frame.

## Modifying List Box Names

---

### Introduction

When Gentran:Server identifies a group in a transaction set, it creates a separate frame to contain the data within the group. It also creates a list box in the parent frame to allow the screen entry translation object user access to the items in the group. The system automatically sets the names for list boxes in a form when the form is created, based on the name of the segment or group. You can modify the names of the list boxes in screen entry forms. For example, you might want to modify a list box name to give it a name that is more meaningful to you than the default name that the system gives it.

---

### Procedure

Use this procedure to modify a list box name.

Step	Action
1	Highlight the form component (segment or group) containing the list box name you want to modify.
2	Select Properties from the Edit menu to display the appropriate dialog box for the form component you selected. <ul style="list-style-type: none"><li>• If you select a segment, the system displays the Segment Properties dialog box.</li><li>• If you select a group, the system displays the Group Properties dialog box.</li></ul>
3	Click the <b>Display</b> tab to access display options.
4	In the List Caption box, type the list box name.
5	Click <b>OK</b> to display the new list box name in the form.

---



# Adding Column Headings to List Boxes

**Introduction**

A single list box in a form may contain two or more columns of data, depending on which fields have been designated list box fields. Although the system automatically generates field labels for list boxes, it does not automatically generate column headings for them. If you want the columns in a list box to be labeled, you must add the column headings by using static text.

**Procedure**

Use this procedure to add column headings to a list box.

Step	Action
1	<p>In the Layout Window, click <b>Add Text</b> from the Display menu.</p> <p><b>System Response</b> Text is added to the current frame, displayed in the upper left corner of the frame. If the translation object is larger than the frame (there is a scroll bar on the right side of the frame), the static text displays in the upper left corner of the part of the translation object displays, not at the top of the translation object.</p>
2	<p>Type the static text in the new selected text field.</p> <p><b>System Response</b> The system displays the Text Properties dialog box.</p>
3	<p>Click <b>OK</b> to add the static text.</p>
4	<p>To move the column heading, click and drag the column heading to a new location in the frame.</p>





# Error Messages

---

**Contents**

- ▶ Overview . . . . . A - 2
  - ▶ Compile Error Messages . . . . . A - 3
  - ▶ Gentran:Server Error Messages . . . . . A - 16
-

## Overview

---

### Introduction

The Gentran:Server error messages and other informational messages are noted in the Compile Errors dialog box when you compile the translation object or in the Error section of the Extended Rules dialog boxes when you compile an extended rule containing errors prior to compiling the translation object, and when you commit an erroneous action in Gentran:Server.

The informational messages are dependent on the context of the program, and are intended to be self-explanatory. The error messages are described in the topics accessed below, along with the actions you can take to correct the problem.

---

### In this appendix

This appendix explains how to determine the action you should take when you receive an error message.

---

## Compile Error Messages

**Introduction** The Compile Error Messages are displayed in the Compile Errors dialog if you compile a translation object with errors. Or, error messages are displayed in the Error section of an Extended Rule dialog if you compile the extended rule containing errors prior to compiling the translation object. After you correct the cause of the errors, click the Compile button again to ensure that the rule is error-free.

**Messages** The compile error messages are listed by the four- or five-digit message number and the error message text. The error definitions contain the actions that you can take to correct the problem (if appropriate) and a description that includes possible causes of the error.

Msg ID	Message Text	Explanation/Your Action
1000	expected '!'	<p><b>Explanation</b> The rule does not have the required “.” between a group name and a field name.</p> <p><b>Your action</b> Insert a “.” between the group name and field name.</p>
1001	no statement to compile	<p><b>Explanation</b> The rule is does not contain any statements.</p> <p><b>Your action</b> Add a statement or statements to the rule.</p>
1002	unexpected end of program	<p><b>Explanation</b> The rule was not complete.</p> <p><b>Your action</b> Finish the rule.</p>
1003	expected ','	<p><b>Explanation</b> The rule does not have the required “,” between parameters.</p> <p><b>Your action</b> Insert a “,” between parameters.</p>
1004	expected ';'.	<p><b>Explanation</b> A statement was not terminated properly.</p> <p><b>Your action</b> Terminate the statement in an appropriate manner.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
1006	no statements to compile	<b>Explanation</b> The body of an IF/ELSE or WHILE condition was empty. <b>Your action</b> Complete the body of the unfinished condition.
1007	syntax error ...	<b>Explanation</b> The map component contains a syntax error. <b>Your action</b> Correct the syntax error.
1008	expected '#'	<b>Explanation</b> The rule does not have the required “#” before a field name. <b>Your action</b> Insert a “#” before the field name.
2000	group ... undefined	<b>Explanation</b> The rule references a group that does not exist. <b>Your action</b> Change the reference to an existing group or delete the reference.
2001	... is not a member of ...	<b>Explanation</b> The rule references a field that does not belong to the specified group. <b>Your action</b> Change the reference to an existing field in the specified group.
2002	insufficient indices to access group ...	<b>Explanation</b> The rule does not give the full addressing for a group. <b>Your action</b> Complete the addressing for the group.
2003	too many indices to access group...	<b>Explanation</b> The rule uses too many addresses for the group. <b>Your action</b> Address the group correctly.
2004	... has not been defined	<b>Explanation</b> The rule references an undefined variable. <b>Your action</b> Define the variable in the declarations section.  (Continued on next page)

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
2005	out of temporary variables	<p><b>Explanation</b> The rule could not be compiled because some expressions are too complex.</p> <p><b>Your action</b> Simplify the expressions and compile the rule again.</p>
2008	field type unknown	<p><b>Explanation</b> The compiler was unable to determine the type of a field.</p> <p><b>Your action</b> Verify that a Data Type is selected for this field.</p>
2009	cannot reference a local field with no current group	<p><b>Explanation</b> The rule (probably pre- or post-session) references a local field, but is not associated with any group.</p> <p><b>Your action</b> Reference the field using the proper addressing.</p>
2010	instances of '%1' are transient and cannot be accessed	<p><b>Explanation</b> The rule references an output group using full addressing. This form of addressing is only appropriate for input groups.</p> <p><b>Your action</b> Reference the output group with the proper address.</p>
2011	no field specified	<p><b>Explanation</b> The rule omits a field reference.</p> <p><b>Your action</b> Add the field reference to the rule.</p>
2012	group ... is promoted and cannot be accessed in this manner	<p><b>Explanation</b> You attempted to access a promoted group in a COUNT or DELETE rule.</p> <p><b>Your action</b> Do not COUNT or DELETE a promoted group.</p>
2100	... is not an array variable	<p><b>Explanation</b> The rule uses array indexing for a variable that is not an array.</p> <p><b>Your action</b> Use the proper indexing for the variable.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
2101	...: array index required	<b>Explanation</b> The rule uses an array variable without using the necessary array indexing. <b>Your action</b> Add the necessary indexing to the array variable.
2102	...: array overflow	<b>Explanation</b> The rule uses an invalid array index. <b>Your action</b> Use the proper array index for the array variable.
2103	only one wildcard index is permitted	<b>Explanation</b> The rule uses more than one wildcard index. <b>Your action</b> Only one wildcard index is permitted per rule.
2104	a wildcard index must be specified	<b>Explanation</b> The rule does not specify a wildcard index when required. <b>Your action</b> Add a wildcard index where necessary.
2200	expected a string	<b>Explanation</b> A required string or string variable is not supplied. <b>Your action</b> Add the required string or string variable.
2201	string overflow	<b>Explanation</b> A string overflow occurred. <b>Your action</b> Verify that the size of a target string is equal to or greater than the source string.
3000	array size expected in declaration of ...	<b>Explanation</b> Invalid array declaration.
3001	declaration of ... missing ']'	<b>Explanation</b> Invalid array declaration.
3002	string size expected	<b>Explanation</b> Invalid string declaration.
3003	string size missing ']'	<b>Explanation</b> Invalid string declaration.  (Continued on next page)



<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
3004	variable name expected	<b>Explanation</b> Invalid variable declaration.
3005	... already defined	<b>Explanation</b> Two variables with the same name are defined at the same scope. <b>Your action</b> Rename one of the two variables.
3006	expected an accumulator number, found ...	<b>Explanation</b> Invalid accumulator reference.
3007	... is not a valid accumulator number	<b>Explanation</b> Invalid accumulator reference.
4000	expected a numeric expression, found ...	<b>Explanation</b> You specified something other than the expected numeric expression. <b>Your action</b> Specify the correct numeric expression.
4001	expected a term, found ...	<b>Explanation</b> You specified something other than the expected term. <b>Your action</b> Specify the correct term.
4002	expected '+' or '-'	<b>Explanation</b> You specified something other than the expected "+" or "-". <b>Your action</b> Specify "+" or "-".
4003	expected '*' or '/'	<b>Explanation</b> You specified something other than the expected "*" or "/". <b>Your action</b> Specify "*" or "/".
4004	expected ')'	<b>Explanation</b> You specified something other than the expected ")". <b>Your action</b> Specify ")".

(Continued on next page)

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
4005	expected a factor, found ...	<b>Explanation</b> Invalid numeric expression.
4006	expected '('	<b>Explanation</b> You specified something other than the expected “(”. <b>Your action</b> Specify “(”.
4007	... is of incorrect type	<b>Explanation</b> The specified expression is of an incorrect type. <b>Your action</b> Specify the correct type for the expression.
4008	expected a relational operator	<b>Explanation</b> You specified something other than the expected relational operator. <b>Your action</b> Specify the correct relational operator.
4009	missing argument	<b>Explanation</b> A required parameter was omitted. <b>Your action</b> Add the required parameter.
4010	assignment expected	<b>Explanation</b> The assignment operator was omitted from an assignment statement. <b>Your action</b> Add the correct assignment operator to the statement.
4011	operator ... requires two arguments	<b>Explanation</b> Only one parameter was supplied for a binary operator. <b>Your action</b> Supply a second parameter for the binary operator.
4012	converting real to integer may lose significant digits	<b>Explanation</b> You are converting a real number to an integer. <b>Your action</b> Verify that losing decimal places is acceptable in this instance.

(Continued on next page)

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
4013	expression too complex, use sub-expressions	<b>Explanation</b> A mathematical expression contains too many terms. <b>Your action</b> Group some of the terms of the mathematical expression in parentheses.
4014	expected an integer, found ...	<b>Explanation</b> The compiler expected to find an integer instead of [...]. <b>Your action</b> Supply the correct value for [...].
4015	no valid condition specified	<b>Explanation</b> You did not specify a valid IF or WHILE condition. <b>Your action</b> Supply a valid condition in the IF or WHILE statement.
4100	expected a date, found ...	<b>Explanation</b> A date is required but not supplied. <b>Your action</b> Specify a date.
4101	expected a date modifier, found ...	<b>Explanation</b> A date modifier was required but not supplied. <b>Your action</b> Specify a date modifier.
4102	... is not a date	<b>Explanation</b> Invalid date expression.
5000	THEN expected	<b>Explanation</b> The compiler expected a THEN condition.
5001	DO expected	<b>Explanation</b> The compiler expected a DO condition.
5002	END expected	<b>Explanation</b> The compiler expected a END condition.
5004	FROM expected	<b>Explanation</b> The compiler expected a FROM condition.
5005	INTO expected	<b>Explanation</b> The compiler expected a INTO condition.  (Continued on next page)

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
5006	END without BEGIN	<p><b>Explanation</b> An END statement was found without a corresponding BEGIN statement.</p> <p><b>Your action</b> Insert a BEGIN statement in the correct location.</p>
5016	misplaced 'BREAK'	<p><b>Explanation</b> The compiler found BREAK outside of a loop.</p> <p><b>Your action</b> Remove the BREAK or place it inside a loop.</p>
5017	misplaced 'CONTINUE'	<p><b>Explanation</b> The compiler found CONTINUE outside of a loop.</p> <p><b>Your action</b> Remove the CONTINUE or place it inside a loop.</p>
5018	too many parameters	<p><b>Explanation</b> Too many parameters were supplied for a function.</p> <p><b>Your action</b> Remove the unnecessary parameters.</p>
5019	too few parameters	<p><b>Explanation</b> Too few parameters were supplied for a function.</p> <p><b>Your action</b> Add the necessary parameters.</p>
6000	expected a database table name	<p><b>Explanation</b> The compiler expected a database table name.</p> <p><b>Your action</b> Insert a database table name in the necessary location.</p>
6001	expected a database column name	<p><b>Explanation</b> The compiler expected a database column name.</p> <p><b>Your action</b> Insert a database column name in the necessary location.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
6002	too many column receivers specified	<p><b>Explanation</b> The number of columns specified did not match the number of receivers in a SELECT statement.</p> <p><b>Your action</b> Correlate the number of columns specified in the SELECT statement to the number of receivers.</p>
6003	too few column receivers specified	<p><b>Explanation</b> The number of columns specified did not match the number of receivers in a SELECT statement.</p> <p><b>Your action</b> Correlate the number of columns specified in the SELECT statement to the number of receivers.</p>
6004	WHERE expected	<p><b>Explanation</b> The compiler expected a WHERE statement.</p>
6005	expected a database key	<p><b>Explanation</b> An invalid database key was specified in a WHERE statement.</p> <p><b>Your action</b> Specify the correct database key in the WHERE statement.</p>
6006	WHERE not allowed with this database table	<p><b>Explanation</b> A WHERE statement is not necessary to access the specified database table.</p> <p><b>Your action</b> Remove the WHERE statement.</p>
6007	invalid key combination	<p><b>Explanation</b> The combination of keys specified in the WHERE statement is not a valid unique compound key to the table.</p> <p><b>Your action</b> Specify a valid combination of keys in the WHERE statement.</p>
7000	... is not a valid seek type	<p><b>Explanation</b> You used an invalid seek type in FSEEK.</p> <p><b>Your action</b> Use BEGIN, END, or CURRENT.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
20001	Record ..., Field ... : date field missing date format	<p><b>Explanation</b> The specified field does not have a date format.</p> <p><b>Your action</b> Edit the field and choose a date format.</p>
20003	Field ... : constant used in standard rule does not exist	<p><b>Explanation</b> The standard rule for the specified field uses an invalid constant.</p> <p><b>Your action</b> Correct the standard rule or create the constant.</p>
20004	Field ... : code list used in standard rule does not exist	<p><b>Explanation</b> The standard rule for the specified field uses an invalid code list.</p> <p><b>Your action</b> Correct the standard rule or create the code list.</p>
20005	Field ... : the qualifier field specified in a use constant standard rule is invalid	<p><b>Explanation</b> The standard rule for the specified field uses an invalid qualifier.</p> <p><b>Your action</b> Correct the standard rule.</p>
20006	Field ... : the field specified to store the code description in a use code standard rule is invalid	<p><b>Explanation</b> The standard rule for the specified field designates an invalid field for the description.</p> <p><b>Your action</b> Correct the standard rule.</p>
20007	Record ... : the specified key field ... : uses an undefined constant	<p><b>Explanation</b> The key field for the specified record uses an invalid constant.</p> <p><b>Your action</b> Correct the key field.</p>
20008	Record ... : the specified key field ... : uses an undefined code list	<p><b>Explanation</b> The key field for the specified record uses an invalid code list.</p> <p><b>Your action</b> Correct the key field.</p>
20010	Record ... : the specified key field ... : is inactive	<p><b>Explanation</b> The key field for the specified record is inactive.</p> <p><b>Your action</b> Activate the key field.</p> <p style="text-align: right;">(Continued on next page)</p>

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
20700	only one binary data and one binary length field are permitted	<p><b>Explanation</b> You have more than one binary data and/or more than one binary length element in one segment.</p> <p><b>Your action</b> Remove the additional binary data and/or binary length elements from the segment.</p>
20701	binary length must precede binary data	<p><b>Explanation</b> The binary length element must be sequenced <i>before</i> the binary data element in the segment so the translator knows how much data to expect.</p> <p><b>Your action</b> Move the binary length element to before the binary data element.</p>
20702	incomplete binary data	<p><b>Explanation</b> You marked a segment as binary, but did not include either a binary length element or a binary data element (or both).</p> <p><b>Your action</b> Add a binary length element and/or a binary data element to the segment.</p>
20703	group ... has no active child objects	<p><b>Explanation</b> You tried to compile the translation object, but the specified group is empty.</p> <p><b>Your action</b> Activate at least one child object in the group.</p>
20704	Element ..., Attribute ...: enumerated attribute declared without accompanying standard rule	<p><b>Explanation</b> The specified XML attribute is configured to use an enumeration but there is no code list standard rule that defines the allowed values.</p> <p><b>Your action</b> Define a use code list standard rule for the specified attribute.</p>
20705	Element ..., Attribute ...: code list used in enumerated attribute does not exist.	<p><b>Explanation</b> The code list for the specified enumerated XML attribute does not exist.</p> <p><b>Your action</b> Define the code list.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>

<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
20706	Element ..., Attribute ...: default value is not valid ...	<p><b>Explanation</b> The specified default value is not in the code list for this XML attribute.</p> <p><b>Your action</b> Ensure that the specified default value is in the code list.</p>
20707	Codelist ..., Attribute ...: value used in enumerated attribute code list does not match XML NMTOKEN production	<p><b>Explanation</b> A value in the enumeration code list is not valid for XML.</p> <p><b>Your action</b> Verify that all the value specified in the code list are valid (legal) for an XML attribute.</p>
20708	Entity ... : Entity value is invalid	<p><b>Explanation</b> The value of the specified entity is not valid (legal) in XML.</p> <p><b>Your action</b> Correct the entity value.</p>
20709	... : Illegal use of reference character	<p><b>Explanation</b> An entity reference was not terminated properly.</p> <p><b>Your action</b> Correct the entity reference.</p>
20710	... : Malformed character reference encountered	<p><b>Explanation</b> An XML character reference was not terminated properly.</p> <p><b>Your action</b> Correct the character reference.</p>
20711	... : Invalid character referenced	<p><b>Explanation</b> An XML character reference is invalid.</p> <p><b>Your action</b> Correct the character reference.</p>
20712	... : Reference to undefined entity encountered	<p><b>Explanation</b> The referenced entity is not defined.</p> <p><b>Your action</b> Define the correct entity.</p>
20713	... : Circular entity references encountered	<p><b>Explanation</b> An entity references an entity that in turn references the original entity.</p> <p><b>Your action</b> Do not reference the original entity in a circular manner.</p> <p style="text-align: right; color: green;">(Continued on next page)</p>



<b>(Contd) Msg ID</b>	<b>Message Text</b>	<b>Explanation/Your Action</b>
20714	...: default does not match the attribute type	<b>Explanation</b> The default value is the wrong type for the attribute. <b>Your action</b> Either correct the type of the attribute or the default value.
20715	...: Contains illegal character ('<')	<b>Explanation</b> The default value contains the illegal character '<'. <b>Your action</b> Correct the default value.

---

## Gentran:Server Error Messages

**Introduction** The Gentran:Server error messages are displayed when you commit an erroneous action. When the system displays an error message, you must first acknowledge the message by either clicking **OK**, and then take the appropriate action.

**Messages** The error messages are listed alphabetically below by the first letter of the error message text. The error definitions contain the actions that you can take to correct the problem (if appropriate) and a description that includes possible causes of the error.

Message Text	Explanation/Your Action
A code list entry must have a code value	<b>Explanation</b> You attempted to add a code without an associated code value.
A code list must have an element id	<b>Explanation</b> You attempted to create a code list without an element ID.
A code list for this element already exists. You must use another element id or delete the original code list first	<b>Explanation</b> You attempted to create a code list with the same element ID as an already existing code list.  <b>Your action</b> Either change the element ID or delete the original code list.
A condition field is required	<b>Explanation</b> You established a conditional relationship that requires a condition field, but did not specify a condition field.
A data type is required.	<b>Explanation</b> You did not specify the Data Type of a field.
A field must have a name that is unique within its parent group	<b>Explanation</b> You tried to give a field a name that is already in use by another field within the same group (not necessarily the same record).
A group must have a unique name	<b>Explanation</b> You tried to give a group a name that is already used by another group or record.
A group or record with a maximum usage of 1 cannot be split or promoted	<b>Explanation</b> You attempted to split or promote a single-occurrence group or record. This is not a valid action.  (Continued on next page)

Message Text	Explanation/Your Action
A key field has been selected but no key value has been specified.	<p><b>Explanation</b> You established a key field without specifying a key value.</p>
A Memory Exception has occurred.	<p><b>Explanation</b> A system error occurred.</p> <p><b>Your action</b> Exit Gentran:Server and restart Windows.</p>
A name must be entered	<p><b>Explanation</b> You did not specify a name for an object.</p>
A problem occurred while attempting to close loop .... This is probably due to incorrect standards.	<p><b>Explanation</b> An error occurred when reading from the standards.</p>
A record must have a unique name	<p><b>Explanation</b> You tried to give a record a name that is already used by another group or record.</p>
A Resource Exception has occurred.	<p><b>Explanation</b> A system error occurred.</p> <p><b>Your action</b> Exit Gentran:Server and restart Windows.</p>
A serious error was encountered whilst accessing the clipboard and the action was abandoned	<p><b>Explanation</b> The system aborted a cut, copy, or paste operation because a serious error occurred.</p> <p><b>Your action</b> Perform the cut, copy, or paste operation again.</p>
A system error was encountered while compiling the translation object	<p><b>Explanation</b> While compiling a translation object, a system-related problem, such as lack of memory, prevented the compile from completing.</p> <p><b>Your action</b> Close unnecessary applications to free memory, and reboot your computer (if necessary), and then recompile the translation object.</p>
A TDF could not be generated due to a lack of available memory. Either restart Windows or close down other applications, then try again	<p><b>Explanation</b> While generating a TDF, the system ran out of memory.</p> <p><b>Your action</b> Exit Gentran:Server, restart Windows and regenerate the TDF.</p> <p style="text-align: right;">(Continued on next page)</p>

Message Text	Explanation/Your Action
An accumulator must be chosen for this entry	<b>Explanation</b> You did not specify the accumulator to be used in a Use Accum standard rule.
An invalid usage count has been entered	<b>Explanation</b> You entered an invalid usage count for a record or group.
No actions have been chosen for this accumulator entry	<b>Explanation</b> You selected an accumulator but did not specify any actions for it in a Use Accum standard rule.
No alternate accumulator has been selected	<b>Explanation</b> In a Use Accum standard rule, you attempted to create an accumulator entry that used an alternate accumulator, but did not specify which alternate accumulator should be used.
No character ranges have been specified for this token.	<b>Explanation</b> You did not specify the characters allowed for a syntax token.
No fields have been used in this relationship	<b>Explanation</b> You established a conditional relationship but did not specify the fields involved.
No more accumulator entries can be created	<b>Explanation</b> You attempted to create more than six accumulator entries in a Use Accum standard rule.
No more field mappings can be created	<b>Explanation</b> You attempted to create more than eight field mappings for a Select standard rule.
The constant ID must be unique.	<b>Explanation</b> You entered a literal constant identifier in the ID field that is already used in an existing constant.  <b>Your action</b> Type a unique literal constant identifier in the ID field.
The contents of the clipboard cannot be pasted into the translation object at this point	<b>Explanation</b> Either the Clipboard does not contain copied or cut data, or you are trying to paste XML information into a positional file format or paste position information into an XML file format.  (Continued on next page)

Message Text	Explanation/Your Action
The file format cannot be deleted	<b>Explanation</b> You tried to delete a file format object.
The Maximum Usage must be greater than zero and not less than the Minimum Usage.	<b>Explanation</b> You entered an invalid maximum usage count.
The number of entries to be split must be greater than zero and less than the maximum number of entries in the original	<b>Explanation</b> You entered an invalid number in the Split dialog box. The number to be split must be greater than zero and less than the maximum number of entries in the original.
The translation object could not be compiled	<b>Explanation</b> The translation object is not ready to be compiled.
The token code must be unique.	<b>Explanation</b> You attempted to create a syntax token with the same token identifier as an existing syntax token.  <b>Your action</b> Type a unique token identifier in the Token field.
These fields cannot be linked because the input field is deeper than the output field. The translator would be unable to address the input field.	<b>Explanation</b> You are attempting to create an invalid link. A valid link involves an Input field and an Output field that are at the same level.
This code already exists. You must use another code or delete the original code first	<b>Explanation</b> You attempted to add a code to a code list that already contained that code.
This field cannot be linked to another field	<b>Explanation</b> Due to an unknown error, Gentran:Server was unable to begin creating the link.
This file is not a valid Gentran translation object	<b>Explanation</b> You attempted to open (load) a file that is not a translation object.
You have entered an invalid character or character code in Record Delimiter 1	<b>Explanation</b> You entered an invalid Record Delimiter 1 on the Positional File Format Properties dialog box.
You have entered an invalid character or character code in Record Delimiter 2	<b>Explanation</b> You entered an invalid Record Delimiter 2 on the Positional File Format Properties dialog box.  (Continued on next page)

Message Text	Explanation/Your Action
You have entered an invalid decimal separator or character code.	<b>Explanation</b> You entered an invalid decimal separator.
You have entered an invalid element delimiter character or character code	<b>Explanation</b> You entered an invalid element delimiter.
You have entered an invalid Pad character or character code	<b>Explanation</b> You entered an invalid pad character for a positional field.
You have entered an invalid release character or character code	<b>Explanation</b> You entered an invalid release character.
You have entered an invalid sub-element delimiter character or character code	<b>Explanation</b> You entered an invalid sub-element delimiter.
You have entered an invalid tag delimiter character or character code	<b>Explanation</b> You entered an invalid tag delimiter.
You must enter a description of the translation object.	<b>Explanation</b> You attempted to save the translation object without entering the translation object description on the Translation Object Details dialog box.  <b>Your action</b> Type the translation object description in the Description field on the Translation Object Details dialog box. This field must be unique because it is used by the system to identify the map.
You must enter a subtable name.	<b>Explanation</b> For a Select or Update standard rule, the selected table requires you to specify a table name in the Sub Table field.
You must enter a valid character range.	<b>Explanation</b> You entered an invalid character range for a syntax token.
You must enter a valid syntax token.	<b>Explanation</b> You did not enter a valid token identifier in the Token field when creating or editing a syntax token.
You must enter a value for this constant.	<b>Explanation</b> You did not specify a value for the constant in the Value field when creating or editing a syntax token.  (Continued on next page)

Message Text	Explanation/Your Action
You must enter an ID.	<b>Explanation</b> You did not specify a literal constant identifier for the constant in the ID field when creating or editing a syntax token.
You must enter the name of the author of this translation object	<b>Explanation</b> You attempted to save the translation object without entering the name of the translation object author on the Translation Object Details dialog box.
You must select a constant type.	<b>Explanation</b> You did not select a constant type from the Type list when creating or editing a constant.
You need to specify Input and Output File Types for the new translation object	<b>Explanation</b> You clicked <b>OK</b> on the Create New Translation Object dialog box without selecting both the Input Format Type and Output Format Type.

---





---

# User Exits

---

## Contents

- Overview . . . . . B - 2
- Definition of ActiveX Terminology . . . . . B - 3
- Gentran:Server User Exit Overview . . . . . B - 4
- Examples of User Exits . . . . . B - 7
- Examples of Automation Servers . . . . . B - 12
- How to Create a User Exit. . . . . B - 15

---



# Overview

---

## Assumptions

Use of this appendix assumes that you:

- Are familiar with the use of ActiveX Automation Servers and languages such as Visual Basic.
- Know the Translator Programming Language (TPL) constructs for creating, manipulating, and deleting ActiveX objects.
- Know the ProgID of the ActiveX object and all its exposed interfaces.
- Understand how and when extended rules are invoked and their scope.

### Advanced feature

User exits are an *advanced feature* of Gentran:Server that should only be used with the above prerequisites.

---

## Restrictions

Gentran:Server extended rules selectively support ActiveX technology.

### Supported

Gentran:Server extended rules support:

- ActiveX Automation Servers.
- Some *ActiveX Controls*, but only those that work as Automation Servers.

### Not supported

Gentran:Server does not support:

- ActiveX controls that are not ActiveX Automation Servers. Such ActiveX controls must be hosted in a graphical user interface (GUI) display, and therefore cannot be used from extended rules.
  - ActiveX Arrays (e.g., the VT\_ARRAY data type modifier)
  - References (e.g., the VT\_BYREF data type modifier) except for output parameters in method calls. In this instance, references are only valid for the duration of the method call.
  - Gentran:Server does not read registry entries or type libraries when compiling extended rules to verify the accuracy of programmatic identifiers (ProgIDs) or interfaces.
  - Extended rules that compare two ActiveX properties or method results, or any combination of properties or method results. This type of comparison is invalid because property and method types are unknown prior to compilation and thus it is not possible to generate the correct comparison code.
-

## Definition of ActiveX Terminology

---

**What is ActiveX?**

ActiveX is a term that encompasses a set of rules that define how applications should share information. ActiveX grew out of technologies developed by Microsoft, specifically Object Linking and Embedding (OLE) and Component Object Model (COM).

---

**What is an ActiveX automation server?**

An ActiveX automation server is an ActiveX component (a .DLL or .EXE program) that can expose part of its functionality, specifically properties and methods, via the IDispatch interface to another program on the system.

Some ActiveX controls function as automation servers.

---

**What is the IDispatch interface?**

The IDispatch interface is a standard COM interface. Automation Servers typically expose their methods and properties through this interface.

---

**What is a method?**

A method is an action or function that is performed by an object (e.g., a calculation or a search).

---

**What is a property?**

A property is a characteristic or parameter of an object (e.g., type, size, or creation date).

---

**What are ActiveX controls?**

ActiveX controls are a specifically defined method of implementing ActiveX technologies. Basically an ActiveX control is a software component that executes common tasks and can integrate into the user interface of an application that provides the necessary ActiveX control host functionality.

The ActiveX control specification enables you to build component software that interacts with your application and Gentran:Server. ActiveX controls require a user interface, so they are not appropriate for translation user exits. These controls can be developed in a variety of programming languages, including Visual Basic.

---

## Gentran:Server User Exit Overview

---

### Why use user exits?

User exits allow you to enhance your functionality or fulfill specific requirements that Gentran:Server does not perform during normal translation.

---

### User exits with Gentran

Gentran:Server supports user exits in *all types of translation objects* through extended rules that enable the use of Microsoft's ActiveX technology. This feature allows you to use extended rules to invoke custom functionality that was created as an *ActiveX Automation Server*.

Specifically, you can:

- create objects,
- delete objects,
- query objects,
- access properties, and
- invoke methods.

#### Note

You can create your own custom functionality as an ActiveX Automation Server, or you can use third party Automation Servers.

Gentran:Server supports ActiveX indexed properties, specifically these types of index:

- variant
- numeric
- string

The index support allows you to use these syntaxes (if they are supported by your Automation server):

```
n = ob.property[1];  
n = ob.property["Count"];  
n = ob.property[ob.method()];
```

Gentran:Server also supports chaining of ActiveX method calls and properties, which simplifies extended rules for Automation servers with moderately complex object models (e.g., ADO).

#### Example

The following is an example of chaining statements:

```
recordset.fields.item["MessageId"].value
```

---

**Examples of user exits**

User exits may be used in the following manner.

- To access your database table to perform cross-reference or lookups instead of using the Gentran:Server tables.
- To perform complex pricing calculations (e.g., involving multiple customers, where they are located, and where the product is sold).

**Data types supported**

The following table lists the extended rule data types and the corresponding ActiveX data types that are supported for use with user exits:

Extended Rule Data Type	ActiveX Data Type
INTEGER	VT_14
REAL	VT_R8
DATETIME	VT_DATE
STRING	VT_BSTR
OBJECT	VT_DISPATCH

**Data type conversion**

The Gentran:Server translator automatically converts the extended rule data type to the ActiveX data type, whenever possible. If the conversion cannot be performed, a type mismatch error is written to the Audit Log and the extended rule is immediately terminated. An error is also written to the translator report, if one is used by the operation. Since an export operation does not generate a translator report, any conversion errors during export translation are only written to the Audit Log.

**Objects**

ActiveX automation servers use the datatype OBJECT, which consists of two elements: properties (a defined set of data) and methods. A method is a function that is defined by the interface of the object.

All objects that are automation servers provide the default interface IDispatch, which exposes the internal functions of an application to Gentran:Server. You can also expose other interfaces.

**Example**

Internet Explorer exposes the interface IWebBrowser2.

---

**Syntax** The object must be declared and then created via the CREATEOBJECT command, by pointing to the object's IDispatch interface. Then, the object's methods may be invoked using the syntax *object.method(parameters)*.

The user may specify the ProgID and an optional InterfaceID (IID). If you just specify the ProgID, Gentran:Server uses the default IDispatch interface. The IID is a machine-readable identifier that uniquely identifies the interface of an object (e.g., {00020300-00B000-C00004005300}). Interfaces can also be identified by their human-readable name (e.g., IDispatch).

**Note**

Each IID must be enclosed in braces {}.

---

**Location** The location of the user exit in the map depends on what the user exit needs to do and what you want the scope of the rule to be. Typically, you might:

- declare an object in the On Begin rule of the Input side of the map (Loop Level Extended Rules dialog box of Input positional file),
- create and use the object in the On Begin rule of the Input side of the map (Loop Level Extended Rules dialog box of Input positional file), and
- delete the object in the On End rule of the Output side (Loop Level Extended Rules dialog box of Output EDI file).

**Note**

The sample above is intended as an example only. You can declare, create, use, and delete an object in an extended rule for a map component at any hierarchical level, including field level extended rules, if appropriate.

**Reference**

See *Defining a Form Component Extended Rule* on page 6 - 11 for more information about creating extended rules for various form components.

---

# Examples of User Exits

---

## Introduction

This section contains examples of the construction of user exits. In this section the following conventions apply:

- Bold typeface is used to express what you need to type.
- Lines that are too long to fit on one line in this section may be continued on the next line using a line-continuation character (Â).
- A double slash (//) introduces comments.

---

## Defining object variable

This example defines a variable that represents an ActiveX Automation Server:

```
object ob;
```

---

## Creating a default interface

This example creates an instance of the default interface of the Internet Explorer ActiveX Automation Server:

```
object ob;  
ob = createobject("InternetExplorer.Application");  
//Creates an instance of the default interface of an ActiveX  
//Automation Server.
```

---

## Creating a specific interface

This example creates an instance of the Internet Explorer ActiveX Automation Server and requests a specific interface:

```
object ob;  
ob = createobject("InternetExplorer.Application", "{EAB22AC1-30C1  
Â-11CR-A7EB-000C05BAE0B}");  
//Creates an instance of a specific interface of an ActiveX Automation  
//Server. In this case, the IID is known (specified in braces).  
//Note:  
//The createobject command is more efficient if you use the IID  
//instead of the interface name.
```

This example creates an instance of the Internet Explorer ActiveX Automation Server, where the interface identifier of the desired interface is unknown, but the name of the interface is IWebBrowser:

```
object ob;  
string[50] iid;  
iid = getiid("InternetExplorer.Application", "IWebBrowser2");  
ob = createobject("InternetExplorer.Application", iid);  
//Creates an instance of a specific interface of an ActiveX  
//Automation Server. In this case, the IID is not known, so the  
//ProgID and name of the interface (IWebBrowser) are specified and  
//Gentran:Server looks up the IID. The IID is loaded into the  
//string variable "iid" and then that value is used to create the  
//object.
```

**Deleting an object**

An object must be deleted before the end of the map that uses it. It is more efficient to delete the object immediately when you no longer need it, although the Gentran:Server translator deletes the object automatically at the end of the map. Also, if you assign one object to another one, both copies of the object must be deleted for that object to be properly unloaded.

This example deletes the object “ob” that was used in the previous examples:

```
deleteobject (ob);
```

---

**Getting a property value**

This example obtains a property value from an ActiveX Automation Server:

```
object ob;
string[50] iid;
iid = getiid("InternetExplorer.Application", "IWebBrowser2");
ob = createobject("InternetExplorer.Application", iid);
IF ob.Visible = 1 THEN
BEGIN
//The property value is accessed.
END
//Test to see if the system displays the Internet Explorer.
```

---

**Setting a property value**

This example sets a property value in an ActiveX Automation Server:

```
object ob;
string[50] iid;
iid = getiid("InternetExplorer.Application", "IWebBrowser2");
ob = createobject("InternetExplorer.Application", iid);
ob.Visible = 1;
//The property value is set.
//Display Internet Explorer on the desktop.
```

---



### Passing parameters to a method

To pass parameters to a method, use the syntax *objectname.methodname(parameters)*. The Gentran:Server translator automatically converts the extended rule data type of the property to the ActiveX data type, whenever possible. If there is no directly corresponding type in the extended rules, the conversion is performed as well as possible by the operating system.

#### Note

Gentran:Server relies on default data types. For example, Gentran:Server converts VT\_CURRENCY (which is an unknown data type to the translator programming language) to a real or numeric data type. If the conversion cannot be performed, a type mismatch error is written to the Translator Report and the extended rule is immediately terminated.

#### Example

This example navigates to a web page by passing the URL to the Navigate method of Internet Explorer:

```
object ob;
string[50] iid;
iid = getiid("InternetExplorer.Application", "IWebBrowser2");
ob = createobject("InternetExplorer.Application", iid);
ob.Navigate("www.sterlingcommerce.com");
//Navigates to the Sterling Commerce web page by passing the URL to
//the Navigate method of Internet Explorer. Note: URL must be
//enclosed in quotation marks.
```

### Returning values in output parameters

To return values in output parameters, use the syntax *objectname.methodname(InputParameter, OUT OutputParameter)*.

#### Note

- Input and output parameters may occur anywhere in the parameter list.
- Output parameters must be preceded by the OUT keyword.
- When an extended rule variable is used as an output parameter, the ActiveX datatype must match exactly.

#### Reference

See *Data types supported* on page B - 5 for a list of ActiveX data types and the analogous extended rule data type for each.

#### Example

This example decrypts data in a field:

```
object ob;
ob = createobject("YourCompany.DecryptionUserExit");
ob.Decrypt("EncryptionKey", OUT #Field_Name);
```

### Testing successful object creation

This example tests the value of an object against zero to determine if it was correctly created:

```
object ob;
ob = createobject("YourCompany.DecryptionUserExit");
IF ob = 0 THEN
BEGIN
//Object not created properly; perform task X.
END
//Test to see if the object was created successful. If it was not
//created, perform the specified task.
```

### Obtaining another interface of an existing object

This example uses the IID of an object to obtain a different interface of an existing object (object1):

```
object ob, ob2;
ob = createobject("InternetExplorer.Application");
ob2 = queryobject(ob, "{EAB22AC1-30C1-11CF-A7EB-0000C05BAE0B}");
If the IID of the desired interface is unknown, use the getiid function
to determine the correct IID and then use the queryobject function, as
this example demonstrates:
object ob, ob2;
string[50] iid;
ob = createobject("InternetExplorer.Application");
iid = getiid("InternetExplorer.Application", "IWebBrowser2");
ob2 = queryobject(ob, iid);
```

### Accessing a database

This example uses a user exit to cross-reference a UPC code with a value in a database to return an SKU number. This example adds the user exit to an invoice (import) map (refer to the Gentran:Server TUTORIAL folder, PET\_810.MAP for an example). The user exit code is implemented by increments in the map, with each section added to the logical map component.

#### On Begin Extended Rule

Type the following code into the On Begin extended rule of the INPUT positional file:

```
//Declare the object
object ob;
//Create an instance of the Visual Basic Active X Automation Server
ob = CreateObject("SCUPCSKU.SKUResolve");
//To create an instance of the C++ ActiveX Automation Server
//use the following command instead:
//ob = CreateObject("UpcSku.Application");

if ob = 0 then
begin
MessageBox("Create Object failed",0);
end
```

(Continued on next page)

**Accessing a  
database  
(contd)****INVDETAIL.UPCCODE Field Extended Rule**

Type the following code into the UPCCODE field extended rule on the Input side of the map (INVDETAIL record):

```
string [100] msg;
msg = "UPC Code = ";
concat(msg, #UPCCODE, 12);

//Call the automation server
#UPCCODE = ob.ResolveSKU(#UPCCODE);
concat(msg, ", SKU Code = ", 13);
concat(msg, #UPCCODE, len(#UPCCODE));

//Display message box to user listing UPC code and SKU matches
MessageBox(msg, 0);
```

**On End Extended Rule**

Type the following code into the On End extended rule of the INPUT positional file:

```
//Delete the object
deleteobject(ob);
```

**Note**

See *Examples of Automation Servers* on page B - 12 for sample automation servers (written in Visual Basic and C++) that access a Microsoft Access Database to cross-reference a UPC code with an SKU code.

## Examples of Automation Servers

### Introduction

The following pages contain sample automation servers that access a Microsoft Access Database to cross-reference a UPC code with an SKU code. The first example is written in the Visual Basic programming language and the second one in C++.

### Note

Both of the following automation servers could be used with the “Accessing a database” user exit extended rules described in *Accessing a database* on page B - 10.

### Visual Basic Automation Server

This sample automation server was written in the Visual Basic programming language. It accesses a Microsoft Access Database to cross-reference a UPC code with an SKU code. The syntax is: *ResolveSKU(UPCCode as String)*. Note that after compiling the Automation Server (SCUPCSKU.EXE), it was registered using REGSRV32.EXE because it was not self-registering.

```
//This is a Visual Basic Active X EXE project. All code shown here
//resides
//in the SKUResolve class. (Class 1 was renamed to SKUResolve).
//To access the database, the following reference was included in
//the project: Microsoft DAO 2.5/3.5 Compatibility Library.
//The following line forces the compiler to make sure all variables
//are declared prior to use within the program.
Option Explicit
//This function receives a UPC code as it's only parameter and
//cross-references it with a value in the database to get the
//corresponding SKU code.
Public Function ResolveSKU(UPCCode As String) As String
    Dim DB As Database, RS As Recordset
    Set DB = DBEngine.Workspaces(0).OpenDatabase
    A("c:\GENSRVNT\tutorial\upcsku.mdb")
    Set RS = DB.OpenRecordset("UPCSKU", dbOpenDynaset)
//Make sure there are no trailing spaces on the parameter passed
//to this function.
    UPCCode = Trim(UPCCode)
//Since the database is case-sensitive make sure that the parameter
//has all capitalized letters in it.
    UPCCode = UCase(UPCCode)
    RS.MoveFirst
//Loop until we find the UPC code or until the end of the recordset
//is found.
    While Not RS.EOF
//If we find the UPC code, return the 'SKU code associated with it.
        If RS("UPCCode") = UPCCode Then
            ResolveSKU = RS("SKUCode")
            Exit Function
        End If
        RS.MoveNext
    Wend
//No matching UPC code was found for the parameter passed to the
//function. Return a value stating that there was no SKU found.
    ResolveSKU = "No Record Found"
End Function
```

**C++ Automation Server**

This sample automation server was written in the C++ programming language. It performs exactly the same function as the previous Visual Basic sample user exit—accessing a Microsoft Access Database to cross-reference a UPC code with an SKU code. Note that after compiling the Automation Server (UPCSKU.DLL), it was registered using REGSRV32.EXE because it was not self-registering.

```
// This is a MFC AppWizard (dll) project with the Automation option
//selected. All code shown here resides in the CUpcSkuMain class
//which is derived from CCmdTarget. When the class was created in
//ClassWizard, the ProgID
//"UpcSku.Application" was specified in the "Createable by type
//ID:" field.
// This ProgID is different than the one used for the VB example so
//that both Automation servers can co-exist in the registry.

// To access the database, a class (CUpcSkuDaoRecordset) derived
//from CDaoRecordset is needed. DAO and Snapshot options were
//selected and the location of the database (Upcsku.mdb) was
//specified during the creation of the class in ClassWizard.
// Include the header file for the derived recordset class
#include "UpcSkuDaoRecordset.h"
// This function is called when the last reference for this
//automation object is released

void CUpcSkuMain::OnFinalRelease()
{
    CCmdTarget::OnFinalRelease();

    // Since we used DAO to access the database,
    // terminate DAO now so the DLL can unload.

    if (AfxOleCanExitApp())
        AfxDaoTerm();
}
// This function receives a UPC code as it's only parameter and
//cross-references it with a value in the database to get the
//corresponding SKU code.

BSTR CUpcSkuMain::ResolveSKU(LPCTSTR UPCCode)
{
    CString strResult;
    CString sUPCCode = UPCCode;

    CDaoDatabase DB;
    DB.Open("C:\\GENSRVNT\\Tutorial\\Upcsku.mdb", FALSE, TRUE);

    CUpcSkuDaoRecordset RS(&DB);
    RS.Open();
```

(Continued on next page)

**C++ Automation  
Server (contd)**

```
// Make sure there are no trailing spaces on the parameter passed
//to this function.

    sUPCCode.TrimRight();

// Since the database is case-sensitive make sure that the
//parameter has all capitalized letters in it.

    sUPCCode.MakeUpper();

    RS.MoveFirst();

// Loop until we find the UPC code or until the end of the recordset
//is found.

    while ( !RS.IsEOF() )
    {
// If we find the UPC code, return the SKU code
// associated with it.

        if ( RS.m_UPCCode == sUPCCode )
        {
            strResult = RS.m_SKUCode;

            RS.Close();
            DB.Close();

            return strResult.AllocSysString();
        }

        RS.MoveNext();
    }

// No matching UPC code was found for the parameter passed to the
// function. Return a value stating that there was no SKU found.

    strResult = "No Record Found";

    RS.Close();
    DB.Close();

    return strResult.AllocSysString();
}
```

---

## How to Create a User Exit

### Before you begin

The ActiveX Automation Server must be installed on the same machine as the Gentran:Server translator (where you are using the translation object that contains the user exit).

### Procedure

Use this procedure to create a user exit.

Step	Action
1	Start the Gentran:Server Application Integration subsystem.
2	Open the map in which you want to apply a user exit.
3	<p>Determine where the user exit will be located in the map (e.g., session rule or extended rule).</p> <p><b>Note</b> When you apply a user exit to a map component, subordinate map components may also be able to execute the same user exit.</p>
4	<p>Construct the user exit.</p> <p><b>Reference</b></p> <ul style="list-style-type: none"> <li>▶ See <i>Defining a Form Component Extended Rule</i> on page 6 - 11 for more information about creating extended rules for various form components.</li> <li>▶ See <i>Alphabetic Language Reference</i> on page 6 - 41 for more information about the <i>createobject</i>, <i>deleteobject</i>, and <i>getiid</i> commands.</li> </ul>





---

# Using the Translator Command Line Interface

---

<b>Contents</b>	▶ Introduction . . . . .	C - 2
	▶ Command Line Syntax . . . . .	C - 3

---



## Introduction

---

**In this appendix**

This appendix describes how to use translator command line interface to invoke the services of TXDE.EXE (print and screen entry translator).

**Note**

This interface is currently used by all applications that invoke the services of the TXDE.EXE translator.

---

# Command Line Syntax

**Introduction** The syntax of the command line is:

```
txde[-r] [-n]
    [-a server]
    [-e eventid]
    [-p filename]
    [-d [dockey] ]
    [-u param -u param ...]
```

**Syntax parameters** This table describes the command line syntax parameters.

Parameter	Description	
-r	Read-only mode	
-n	Do not move document to InDrawer after print or export	
-a	Server (name of audit server to which TXDE.EXE connects)	
-e	EventID (identifier of an audit event to which this translator belongs)	
-P	Print mode	
	Subparameter	Description
	filename	Path to file containing document keys (one per line)
-d	Screen entry mode	
	Subparameter	Description
	dockey	Key of document to edit, if modifying an existing document
-u	User-defined parameters accessible from extended rules	
	Subparameter	Description
	param	Parameter value



# Glossary

---

<b>?In Documents</b>	This Gentran:Server browser contains a list of documents that were received by the system but failed compliance checking or that do not have an identifiable partner or transaction set.
<b>?Out Documents</b>	This Gentran:Server browser contains a list of documents that were imported into the system via unattended processing but are invalid.
<b>acknowledge-ment</b>	This term is used to indicate the ANSI 997 functional acknowledgement, the EDIA 999 acceptance/rejection advice, and the EDIFACT CNTRL document.
<b>activation</b>	This function enables you to make map components available for use. The system activates all the groups, segments, composites, and elements that are defined as “mandatory” (must be present) by the standard. The system does not enable you to deactivate the mandatory groups, segments, composites, and elements. When translating data, the system does not process groups, segments, composites, and elements (or records and fields) that are not activated. Therefore, <i>you</i> must activate the groups, segments, composites, and elements that are not defined as mandatory by the standard, but that you have determined that you need to use in mapping.
<b>AIAG</b>	The Automotive Industry Action Group (AIAG) is the standards-setting group for the automotive industry. The standards form a subset of the ANSI X12 standard.
<b>ANA</b>	Article Numbering Association.
<b>ANSI</b>	American National Standards Institute. ANSI sets standards for many products and services, such as safety glasses and battery capacities. The ANSI X12 committee is the chief EDI standards-setting organization for the United States.
<b>application file</b>	If you are creating an import or export map, you must define your application to the Application Integration subsystem. In Gentran:Server terminology, your application file is also referred to as a <i>fixed-format file</i> or a <i>positional file</i> . Your application file must contain all the information that you either need to extract from your partner’s document (if the map is inbound) or need to send to your partner (if the map is outbound), so that your system can accurately process the data.

---

---

<b>application system</b>	Computer systems, outside of EDI, designed to fulfill specific business functions. These include accounting, purchasing, materials control, human resources, shipping, and other systems.
<b>code list tables</b>	These are used by EDI standards as repositories for lists of codes. Each EDI standard provides a code list for each element that can be further defined with a code.
<b>colours</b>	This feature enables you to select foreground and background colours to visually define the various map components. The use of colour is optional.
<b>communications session</b>	Everything sent and received to/from one telephone number in one continuous period of connection. This could include sending two or three interchange envelopes to a network, each for a different trading partner.
<b>Communicator</b>	This is the Gentran:Server communications software. The Communicator enables you to send, receive, resend, establish communication port definitions, establish communication profiles, establish communication scripts, and view and delete communication sessions.
<b>compile</b>	This function compiles the form and creates a translation object. The form that you create using Gentran:Server is a <i>source form</i> . When that source form is compiled, the result is a <i>compiled translation object</i> . This translation object must be registered with the Gentran:Server system before you can use it.
<b>compliant</b>	This means that the document conforms to the EDI standards as defined by the translation object.
<b>component data element</b>	A simple data element that belongs to a composite data element. Component data elements are also called sub-elements.
<b>composite data element</b>	A contains two or more component data elements or sub-elements. Composites are defined by the EDI standards that use them (EDIFACT, TRADACOMS, and certain ANSI X12 standards).
<b>conditions</b>	See relational conditions.
<b>constant</b>	This standard rule enables you to move a literal constant value to the specified element or field, indicate a qualifying relationship with another element or field, and map the current date or time to the specified element or field.

---

---

<b>cross-reference table</b>	This table is created in Partner Editor and is used to convert your values to your trading partner's values during outbound processing, or your partner's values to your values during inbound processing.
<b>date/time</b>	This type of field or element contains a date or time. If you specify that a field or element is a date/time type, you must specify exactly how the date or time must be formatted.
<b>deactivation</b>	This function enables you to make map components unavailable for use by the system. The system does not enable you to deactivate the mandatory groups, segments, composites, and elements.
<b>default</b>	A predefined value – programs use these built-in values <i>unless</i> you specifically override them.
<b>document</b>	<p>One transaction set containing actual data and treated as a single entity. The amount of data does not affect whether something is a document or not, only the fact that it is treated as a single entity. For example, regardless of whether a purchase order contains one item or ten thousand, if it is one purchase order, it is one document.</p> <p>If a trading partner sends you a communication containing 10 purchase orders, you have received 10 documents. If the communication contained 15 invoices, you have received 15 documents.</p>
<b>EANA</b>	International Article Numbering Association, Brussels, Belgium.
<b>EDI</b>	Electronic Data Interchange (EDI) is the process by which companies can exchange business documents directly from application to application by computer without paper documents being produced.
<b>EDI File Format Window</b>	The EDI File Format Window contains the EDI file format. When you click a segment or group in the EDI file format, the corresponding frame is displayed in the Layout Window. When you click an element in the EDI file format, the corresponding layout component is highlighted in the Layout Window.
<b>EDI standard</b>	This is the rules for turning a business document into an EDI document.
<b>EDIA</b>	The Electronic Data Interchange Association, formerly known as the Transportation Data Coordinating Committee (TDCC).

---

---

**EDIFACT** This is the standards organization of the United Nations Economic and Social Council. The acronym is short for the Electronic Data Interchange for Administration, Commerce, and Transport.

---

**element** The smallest piece of usable information defined by the standards. Examples might include a quantity, unit price, or description. An individual element can have somewhat different meanings depending on context. Therefore, elements are normally not considered to have useful meaning until they are combined into segments. An element is the EDI map component that is mapped (linked) to a corresponding application field to move data to and from the EDI file.

There are three types of data elements, as illustrated in the table below:

Data Element	Definition
Simple data element	A single piece of information defined by the standards.
Composite data element	A data element that is made up of multiple component elements.
Component data element	A simple data element that belongs to a composite data element. Component data elements are also called sub-elements.

---

**envelope** A way of separating information in transmissions for ease of processing. Each envelope contains a header segment and a trailer segment, which separate the envelope from other envelopes and provide information about the contents of the envelope.

There are three levels of envelopes, as follows:

- **Transaction Set**  
Each transaction set (business document) is contained within a transaction set envelope.
- **Functional Group**  
An envelope containing related business documents. The standards define which transaction sets should be placed together into a functional group envelope.
- **Interchange Envelope**  
All material being sent to one trading partner in one communication. The term Interchange Envelope is the term used by ANSI. EDIA uses the term Transmission Envelope to refer to this level of envelope. Since we use the term transmission for other uses, we refer to Interchange Envelope only.

**Note**

A communications session could easily include a number of interchange envelopes. For this reason, the standards-setting bodies are considering a fourth level of envelope to cover an entire communications session.

---



---

<b>equalize</b>	This function enables you to reinstate the two sides of the map with focus in equal dimensions. The use of the Equalize function is optional.
<b>export map</b>	This map defines how to move data from the EDI standard-formatted documents that your partners send you to your application file (flat file definition). An export map is necessary for inbound processing.
<b>export translation object</b>	See export map.
<b>extended rules</b>	These rules enable you to use a Gentran:Server proprietary programming language to perform virtually any mapping operation you require.
<b>field</b>	The smallest piece of information defined in the application file. A field is the application map component that is mapped (linked) to a corresponding EDI element to move data to and from your application file.
<b>fixed-format file</b>	See application file.
<b>form</b>	A form is a set of instructions you define in the Forms Integration subsystem that indicates how the system should format data
<b>frame</b>	A frame contains the groups, repeating segments, and elements at that level (single segments are not represented on the frame). Gentran:Server uses frames to format the EDI file for the translation object.
<b>functional group</b>	A group of transaction sets that the standards-setting body (such as ANSI) has defined as fitting together with other related transaction sets. An example might be a Functional Group defined as containing all purchasing transaction sets.
<b>group</b>	A looping structure that contains related records/segments and/or groups that repeat in sequence until either the group data ends or the maximum number of times that the loop is allowed to repeat is exhausted. If you create a group that is subordinate to another group (a sub-group), this corresponds to a nested looping structure (a loop within a loop). The application (positional) file and the EDI file are both groups and therefore, they are visually represented the same way as other groups and sub-groups in Gentran:Server.

---

---

<b>import</b>	This command enables you to import data from an external application file. Depending on the content of the file, the system may prompt you for partner, transaction, or translation object information.
<b>import map</b>	This map defines how to move data from your application file (flat file definition), which may contain multiple documents, to the EDI standard-formatted documents that your partners expect to receive from you. An import map is necessary for outbound processing.
<b>import translation object</b>	See import map.
<b>In Documents</b>	This Gentran:Server browser contains a list of documents that the system received but that have not yet been processed by the user. Once the documents are processed (printed or exported) the documents are transferred to the In Drawer.
<b>inbound mapping</b>	The system translates your trading partner's EDI standard formatted business document to your application file format, so you can receive documents from you partners. To translate inbound data, you need to create an export map.
<b>In Drawer</b>	This Gentran:Server browser contains a list of documents that were received and processed.
<b>integer</b>	An integer is a number that has an implied decimal point (e.g., "2.01" is formatted as "201").
<b>interchange</b>	The interchange contains all the functional groups of documents (transaction sets) sent from one sender to one receiver in the same transmission.
<b>Interchanges browser</b>	This Gentran:Server browser enables you to view a hierarchical list of all the interchanges that were sent or received. It contains status information about the interchanges, (e.g., whether or not a functional acknowledgement was received and the status of that acknowledgement). This browser also enables you to access the documents within the interchanges. The Interchanges browser can always be accessed.
<b>Layout Window</b>	The Layout Window contains the actual format of the translation object. Each group (with a maximum usage greater than one) and repeating segment has a corresponding frame in the Layout Window. When you click a segment or group in the EDI file format, the corresponding frame is displayed in the Layout Window. When you click an element in the EDI file format, the corresponding layout component is highlighted in the Layout Window.

---

---

<b>list</b>	A list displays a list of choices. Scroll bars are provided if there are more choices than can fit in the box.
<b>literal constant</b>	These are used by the system as a repository to store information that is used at a later point in the map. Typically, constants are used in an outbound map to generate a qualifier.
<b>location table</b>	This is a table created in Partner Editor that is used to contain address-related information about the partner.
<b>lookup table</b>	This is a table created in Partner Editor that is used to select information related to a value in inbound or outbound data.
<b>loop count</b>	This standard rule enables you to count the number of times a loop is repeated, if the element or field is part of a loop. If the loop is a nested loop, you can track the current loop <i>or</i> the outer loop. For example, if the Y loop is nested within the X loop, and the Y loop has cycled through 15 iterations and the X loop has cycled through 3 iterations, you can choose to count either the “15” (Y loop) or the “3” (X loop).
<b>loop start/loop end</b>	Certain EDI standards use Loop Start (LS) and Loop End (LE) segments. LS and LE segments differentiate between two or more loops of the same type. If the transaction contains LS and LE segments and depending on whether your map is inbound or outbound, you need to define the LS and LE segments for the loops you are using in the map in one of two different ways.
<b>map</b>	A set of instructions you define in the Application Integration subsystem that indicates the corresponding relationship between your application file and the EDI standards, and defines how the system should translate data.
<b>network</b>	Also known as a Third Party Network or Value-Added Network (VAN) that accepts and holds transmissions from companies until it is convenient for a trading partner to accept them.
<b>number</b>	This type of field or element contains either an integer or real number. If you specify that a field or element is a number type, you must designate the format by specifying a format of either “N” (integer) or “R” (real) and the number of decimal places.
<b>ODETTE</b>	The Organization for Data Exchange by Tele-Transmission in Europe.

---

---

<b>Out Documents</b>	This Gentran:Server browser contains a list of documents that are ready to be sent. After the documents are successfully sent, they are automatically transferred to the Out Drawer.
<b>outbound mapping</b>	The system translates your application file format to EDI standard formats, so you can send documents to your partners. To translate outbound data, you need to create an import map and a system import map in the Application Integration subsystem.
<b>Out Drawer</b>	This Gentran:Server browser contains a list of documents that were successfully sent out by the system.
<b>partner</b>	Another firm with which your company trades documents. Also referred to as a trading partner.
<b>Partner Editor</b>	This Gentran:Server function enables you to define, edit, and delete all partner information for your company and all of your trading partners.
<b>positional file</b>	See application file.
<b>post</b>	This Gentran:Server function enables you to move compliant documents from the Workspace to the Out Documents.
<b>predefined</b>	On a data entry translation object, a default value for a particular element. You can change the value as necessary.
<b>printout</b>	This is produced when data received from a trading partner passes through a print translation object. You do not have to actually print the data. The printout could be to a file on your hard disk.
<b>print translation object</b>	A print translation object organizes and formats the printout of EDI documents that are received from or sent to the trading partners for which you have established a trading relationship (inbound or outbound) that utilizes that print translation object. The print translation object enables you to view the EDI document in an easily readable format.
<b>promote</b>	This function extracts one iteration (instance) of a group or repeating segment. This enables you to map unique data from your application file, and/or enter a specialized definition. Gentran:Server specifies that only one-to-one (no loops) or many-to-many (loop) mapping relationships are valid.

---

---

<b>qualifier</b>	This is an element that has a value expressed as a code that gives a specific meaning to the function of another element. A qualifying relationship is the interaction between an element and its qualifier. The function of the element changes depending on which code the qualifier contains.
<b>real number</b>	A real number has an explicit decimal point (e.g., “2.01” is formatted as “2.01”), and truncates trailing zeroes.
<b>receive</b>	This Gentran:Server function enables you to manually initiate a communications session to receive data from a network through an invocation of the Connection Manager.
<b>record</b>	Contains a group of related fields. A record can occur once or can repeat multiple times.
<b>relational condition</b>	This function enables to connect fields together for syntax or compliance reasons. For example, Field A is invalid unless Field B is present. Therefore, if you set up a condition that pairs Fields A and B, the system generates a compliance error if one of those fields is not present.
<b>respond</b>	This Gentran:Server function enables you to create a turnaround document (using the Turn Around translation object in the partner relationship) in response to one or more selected documents in the In Drawer.
<b>responsible agency</b>	An organization that develops and updates standards for EDI communications. These organizations include ANSI, EDIA, AIAG, UCS, EDIFACT, ODETTE, and VICS.
<b>screen entry translation object</b>	A screen entry translation object provides a standardized format for keying an EDI document into the Document Editor, for translation and transmission to your trading partners, for which you have established an outbound trading relationship that utilizes that screen entry translation object. The screen entry translation object ensures that your users key all the data necessary to create the required EDI document.
<b>segment</b>	A group of related elements or composite data elements that combine to communicate useful data. Segments are defined by the EDI standards. A segment can occur once or can repeat multiple times. For example, a catalog price segment might consist of elements for item description, volume, price, lead time, etc. By themselves, none of these elements would communicate useful information. Together, they provide the information necessary for someone to tell if the item is what is desired, whether the price is reasonable, etc. A number of segments together form a transaction set.

---

---

<b>select</b>	This standard rule enables you to select entries from a location table, cross-reference table, partner table, or lookup table (all tables created in the Gentran:Server Partner Editor). You can then map the fields in those tables to one or more fields in the data. The Select function uses the value of the current field to perform the selection.
<b>send</b>	This Gentran:Server function enables you to manually start a communications session to send data to a network through an invocation of Connection Manager. Only selected documents are enveloped and sent. If no documents are selected, ALL documents are sent. Successfully sent documents are moved to the Out Drawer.
<b>set</b>	See Transaction Set.
<b>simple mapping</b>	See link.
<b>split</b>	This function enables you to split (break) a group or repeating segment into two loops. You typically use this function when you need more than one instance of the same map component that still occurs multiple times.
<b>standard rules</b>	These rules give you access to functions that are necessary for mapping operations that are more complex than simple linking, but less involved than extended rules. Each of the standard rules are mutually exclusive (you can use only one on a particular field).
<b>static text</b>	Static text is text that is always included in the screen entry or print translation object. Examples of static text include legends and column headings. Field labels, which usually accompany fields to identify the contents of the fields, are not considered static text.
<b>status bar</b>	The status bar of an application window defines information about a selection, command, or process, defines Menu Bar items as the user highlights each item, and indicates any current keyboard-initiated modes for typing (e.g., CAP for the “Caps Lock” key or NUM for the “Num Lock” key, etc.).
<b>string</b>	This type of field or element contains one or more printable characters. If you specify that a field or element is a string type, you must designate the format by specifying a syntax token.

---

---

<b>syntax token</b>	This function enables you to designate a “token” that defines ranges of characters and/or numbers that are allowed to be used for a string-type element or field. You can then use the syntax tokens in the Format lists located on the Validation tab of the Element Properties dialog. This enables you to define what type of characters should be used while compliance checking each element/field (i.e., alphanumeric within a certain range, numeric within a certain range).
<b>system import map</b>	The translator uses this map to determine which trading relationship (established in Partner Editor) corresponds to each document in the application file, so the system knows which import map to use to process the document. You need a system import map that is defined in EDIMGR.INI to translate outbound data.
<b>system import translation object</b>	See system import map.
<b>system translation object</b>	These translation objects control the creation and separation of interchanges, functional groups, and transaction sets. They also used to generate and reconcile functional acknowledgements. All the required system translation objects are automatically installed with the Gentran:Server system.
<b>TDCC</b>	See EDIA.
<b>TDF</b>	TDF is the Transaction Data File. This file serves as a filter between your document files and the Gentran:Server translator. For outbound processing, data is imported from a TDF file and translated to EDI format using a TDF Import translation object. The data is then ready to be posted and sent to your trading partner. For inbound processing, EDI documents that are received by Gentran:Server can be exported to a TDF-formatted file using a TDF Export translation object. This data file is then ready to be processed or converted to your internal application files.
<b>third-party network</b>	See Network.
<b>TRADACOMS</b>	The U.K. standards for EDI that are published by the Article Numbering Association (UK) LTD.
<b>trading partner</b>	Another firm with which you company trades documents. Also referred to as a partner.

---

---

<b>transaction set (document)</b>	A business form as defined by the standards. Examples include an ANSI 850 purchase order or an UCS 880 invoice. The standards define each transaction set in terms of the segments and elements that make up the form, the order in which they appear, and the relationships among them. This is also known as “message” in Europe.
<b>translation object</b>	<p>A predesigned layout set up to ensure that input or output for a particular transaction set exists and is presented in a usable fashion. You must specify which translation objects are used by each partner relationship.</p> <p><b>Inbound Translation Objects:</b></p> <p><b>Turn Around:</b> This translation object is used when a document is received, to create the natural response document that contains as many elements from the received document as possible.</p> <p><b>Export File:</b> This translation object indicates that when a document is received, it is exported to a specified file format.</p> <p><b>Print:</b> This translation object is used to print documents.</p> <p><b>Outbound Translation Objects:</b></p> <p><b>Import:</b> This translation object is used to import data from an application file.</p> <p><b>Print:</b> This translation object is used to print documents.</p> <p><b>Data Entry:</b> This translation object is used to enter data into the Document Editor facility.</p>
<b>translator</b>	This is the engine that processes data for the Gentran:Server, Application Integration subsystem and Forms Integration subsystem.
<b>transmission</b>	See Communications Session.
<b>transmission chain</b>	A path an EDI communication could follow, including one company, one trading partner, and one or more network services.
<b>turnaround document</b>	A document into which data elements from the source document have been automatically transferred using a turnaround map.
<b>turnaround map</b>	A series of instructions that the system uses to create a turnaround document (a logical response document to the source) from an inbound (source) document, by transferring data from the source document to elements in the target document (translation object).
<b>UCS</b>	The Uniform Communications Standard is the standard used by the grocery industry.

---



- 
- update** This standard rule enables you to update a specific field in a document record, envelope segment, interchange, group, current partner, or document (if the map side format type is EDI), with the contents of the element or field.
- Note**
- Please note that this function updates the internal Gentran:Server database tables. We recommend that you use this function only if you are sure you really want to update the internal database tables.
- Typically, you *only* use this function if you want to update the document name and reference in the document table. Any other use of this function could have disastrous consequences!
- 
- use accum** This standard rule gives you access to a set of numeric variables that you can manipulate via numeric operations, and then transfer to and from fields. This function enables you to add, change, or delete calculations for the element/field, including hash totals (used to accumulate numeric field values, i.e., quantity, price, etc.). This function also enables you to map the accumulated total into a control total field, and use accumulators. Accumulators are used generally for counting the occurrences of a specific element or generating increasing or sequential record or line item numbers.
- 
- use code** This standard rule enables you to match an element or field against a predefined code table, specify whether or not a compliance error is generated if the element or field does not contain one of the values in the code table, and store a code's description in another element or field.
- 
- user translation object** These translation objects control data entry, importing, exporting, document turnaround, and creating printed reports. These translation objects are created using the Application Integration or Forms Integration subsystems. The Application Integration subsystem enables you to generate import, export, and document turnaround translation objects. The Forms Integration subsystem enables you to generate data entry and print translation objects.
- 
- version** Each standards-setting body updates its standards on a regular basis. Each formal update is referred to as a version.
- 
- VICS** The Voluntary Inter-industry Communication Standards is the standards-setting body for the retail industry, a subset of ANSI X12.
- 
- Workspace** This Gentran:Server browser contains a list of outbound "Work in Progress" documents. It also contains recently imported or data entry documents.
- 
- X12** The ANSI committee that sets and publishes standards for EDI.
-

