# Gentran:Server® for UNIX® and Workstation

## Technical Reference Guide

Version 6.1

Sterling Commerce
An IBM Company

# Copyright Notice

**March 2011**

## Sterling Commerce Software

### Trade Secret Notice

## Third Party Software

## 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

## About This Guide

## Processes

### General Processes

### Application Integration Processes

### Visual Mapper Processes

## Environment Variables

## Command Reference

### Overview

## Alphabetical Listing of Commands

# File Record Layouts

## Overview

## Data Type Formats

### Visual Mapper Formats

## Example Record Layout Files

## Index

# About This Guide

**Contents**

# Welcome

Welcome to the *Gentran:Server® for UNIX® and Workstation Technical Reference Guide*, a technical manual about Sterling Commerce's Gentran:Server® electronic commerce software.

## Purpose

The purpose of this *Technical Reference Guide* is to provide additional information about Gentran:Server processes, programs, scripts, files, and data types that is beyond the scope of the *Gentran:Server for UNIX and Workstation Application Integration Guide* or the *Gentran:Server for UNIX Data Flow Administration Guide*.

## Scope

The scope of this *Technical Reference Guide* includes the Gentran:Server Workstation product and all levels of the Gentran:Server for UNIX products. Where features differ by product level, the guide informs you of those differences.

## UNIX knowledge required

Sections of this guide that pertain to Gentran:Server for UNIX assume that you are familiar with basic UNIX concepts and commands, including:

- How UNIX identifies users and associates them into groups
- UNIX user profiles
- File ownership
- Environment variables

# Chapter Contents

This table describes the content of the chapters within this Technical Reference Guide.

| Chapter Title | Description |
|---|---|
| About This Guide | Explains the content, organization, and conventions in this guide, and how to get help with the manual or the product. |
| Processes | Describes the processes performed by the mapping and translation components of Gentran:Server. These processes are listed in alphabetical order. |
| Environment Variables | Describes how to set the environment variables that Gentran:Server uses. The environment variables are listed in alphabetical order. |
| Command Reference | Describes how to use the Gentran:Server command line programs and scripts. The chapter lists the programs and scripts in alphabetical order. |
| File Record Layouts | This chapter describes the file layouts for the files used in Gentran:Server's mapping and translation feature. |
| Data Type Formats | This chapter describes the formats for the Gentran:Server data types. |
| Example Record Layout Files | This chapter describes the application descriptions you can create by reading in a record layout file. |

# Related Publications

## Gentran:Server documentation

This table describes additional documentation for the Gentran:Server software.

| Document | Description |
|---|---|
| Upgrade and Data Conversion Guide | Instructions for upgrading from previous versions of Gentran:Server Workstation and Gentran:Server for UNIX. Also includes instructions for converting the files that are part of the upgrade. |
| Installation Checklist | Description of the recommended sequence in which you should install and configure system components. |
| Gentran:Server for UNIX Getting Started Guide | Instructions for installing the Gentran:Server software and performing setup tasks, such as setting up security.<br>Instructions for starting and exiting Gentran:Server and for setting preferences and default values. Also includes instructions for checking files in and out and saving files. |
| Gentran:Server Workstation Getting Started Guide | Instructions for installing the Gentran:Server Workstation software and performing setup tasks.<br>Instructions for starting and exiting Gentran:Server and for setting preferences and default values. Also includes instructions for checking files in and out and saving files. |
| Application Integration User's Guide | Instructions for performing mapping and translation tasks using the Gentran:Server Application Integration Mapper. |
| Mapping and Translation Guide | Instructions for performing mapping and translation tasks using the Gentran:Server Visual Mapper.<br>**Note**<br>This guide is provided only if you maintain maps created with Gentran:Server version 5.3 or prior. |
| NCPCP User's Guide | Instructions for mapping and translating NCPDP files with the Application Integration system.<br>(Continued on next page) |

| (Contd)<br>Document | Description |
|---|---|
| XML User's Guide | Instructions for mapping and translating XML files with the Application Integration system.<br>**Note**<br>This guide is provided only if your organization has the Gentran:Server XML translation option. |
| ODBC User's Guide | Instructions for mapping and translating ODBC files with the Application Integration system.<br>**Note**<br>This guide is provided only if your organization has the Gentran:Server ODBC translation option. |
| GENCOD User's Guide | Instructions for mapping and translating GENCOD files with the Application Integration system and the Visual Mapper. |
| VDA User's Guide | Instructions for mapping and translating VDA files with the Application Integration system and the Visual Mapper. |
| Data Flow Administration Guide | User instructions for configuring data flows using the Gentran:Server for UNIX software.<br>**Note**<br>This guide is provided only if you have the Gentran:Server EC Workbench or higher product level. |
| Maintenance and Troubleshooting Guide | Instructions for maintaining your Gentran:Server installation. Also provides troubleshooting information to help determine the cause and solution of problems that may occur. |
| Advanced Data Distribution Guide | Instructions for configuring and using the Gentran:Server Advanced Data Distribution product.<br>**Note**<br>This guide is provided only if you have Gentran:Server with Advanced Data Distribution.<br>(Continued on next page) |

| (Contd)<br>Document | Description |
|---|---|
| FTP Daemon User's Guide | Instructions for configuring and using the FTP Daemon tool with the Advanced Data Distribution product. |
| Online Help | Context-sensitive help screens describing the Gentran:Server dialog boxes for the mapping and translation features. Also includes procedures for using the mapping and translation and the data flow administration software. |

## Other documentation

This table lists other documentation you may need to reference when using Gentran:Server.

| Description | Source |
|---|---|
| Instructions for installing and using the operating system on your UNIX computer. | Your hardware vendor<br>The computer manufacturer |
| Instructions for installing and using the communications software required by our Communications Toolkit. | CLEO Communications |

# Documentation Conventions

## Typographic conventions

This table describes the typographic conventions used in this guide.

| Convention | Use |
|---|---|
| Italics | This typeface is used for titles of other manuals and documents; names of directories, files, and file extensions; and to emphasize important information.<br>**Examples**<br>*Gentran:Server for UNIX and Workstation Application Integration User's Guide*<br>*810inbd.map*<br>*Do not continue with this procedure until your backup is complete*. |
| Bold | Bold type is used for program and script names, key terms the first time they are used within a chapter, and entries you are to make on-screen.<br>**Examples**<br>Run the **stoprpcs.sh** script to stop the **mhs_server** and **mhp_server** processes within a specific environment.<br>A **password** is a set of characters a user must enter to gain access to a system.<br>Type **stoprpcs.sh** and press ENTER. |

(Continued on next page)

## Symbols used within syntax statements

This table describes symbols used within syntax statements.

| Symbol | Use |
|---|---|
| < > | Substitute a value for any term that appears within angle brackets. Do not enter angle brackets unless specifically told to do so.<br>**Example**<br>**rm** <filename> means that you should type the name of the file you want to delete.<br>> <filename><br>Be sure to include the redirect symbol (>) preceding the file name. |
| { } | Braces indicate a required part of a statement. Do not enter the braces.<br>**Example**<br>{-f <filename>} means you must enter the f parameter followed by a filename. |
| [ ] | Brackets indicate an optional part of a statement. Do not enter the brackets.<br>**Example**<br>[-f <filename>] means you could type the f parameter followed by a filename, but you are not required to do so. |
| … | An ellipsis indicates that the immediately preceding item can be repeated indefinitely. Do not enter the ellipsis as part of a command.<br>**Example**<br>-e… means that you can repeat -e with other values. |
| ( ) | Within a command, parentheses should be entered as shown. They are part of the syntax of a statement and are not special symbols.<br>**Example**<br>(n) means that you should type a number enclosed by parentheses. |

CHAPTER
1

# Processes

Contents

General Processes

Application Integration Processes

Visual Mapper Processes

# Overview

## Introduction

This chapter explains the processes that relate to the mapping components within Gentran:Server for UNIX and Gentran:Server Workstation.

Each topic contains:

- A description of the purpose of the program(s)
- An illustration of the inputs and outputs to the program(s)
- A table describing the results of possible actions taken by the user or performed by the software.

## How to use this chapter

This chapter is divided into three sections:

- **General Processes**, which are processes that apply regardless of the mapper you use.
- **Application Integration Processes**, which are processes that apply only to the Application Integration subsystem.
- **Visual Mapper Processes**, which are processes that apply only to the Visual Mapper. Use this section if you have the Visual Mapper.

You can use this chapter to increase your understanding of the Gentran:Server product and its mapping systems and to determine whether you set up Gentran:Server properly.

# General Processes

## Archive Processing

### Purpose

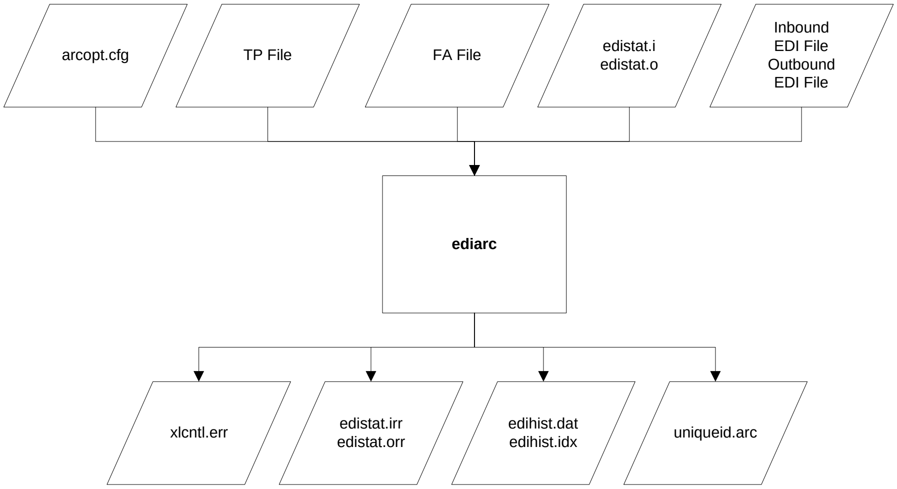
Use this process to copy your EDI data to an archive area for future use. This process is optional.

### XML data

The translator that Gentran:Server for UNIX V6.1 and Gentran:Server Workstation V6.1 use does not archive XML data. This translator archives EDI data in both products.

### Inputs and outputs

This illustration shows the inputs and outputs for the archive process.



(Continued on next page)

## What happens during the archive process

This table describes what happens during the process of archiving data.

| WHEN… | THEN ediarc… |
|---|---|
| **ediarc** starts | Opens the following files:<br>◗ The arcopt.cfg file<br>◗ The EDI data (inbound or outbound) file<br>◗ The FA File<br>◗ The *uniqid.arc* file<br>◗ The *xlcntl.err* file<br>◗ The TP file. |
| **ediarc** reads *edistat.i* or *edistat.o*, one record at a time | Follows the path to the EDI data and reads it. |
| **ediarc** reads the TP record associated with the *edistat.i/o* record from the TP file | Retrieves history creation and archive storage values. |
| **ediarc** reads the history creation value | Appends the *edistat.i/o* record to the permanent Audit files (*edihist.dat* and *edihist.idx*). |
| **ediarc** reads the archive storage value | Places the EDI data sets into the file *<uniqid>.arc.* |
| The data set is the last one in the interchange | Envelopes the *<uniqid>.arc* file with both the interchange and group envelopes. (Group envelopes are not used if none exist in the EDI data.)<br>The system closes the *<uniqid>.arc* file. |
| The user has specified file compression | Executes the appropriate compression program. |
| Any errors occur during archiving | Saves *edistat.i* or *edistat.o* and renames the record to *edistat.irr* or *edistat.orr*. |
| The archive process is complete | Produces or appends to file *xlcntl.err*, which contains the status messages resulting from program execution. |

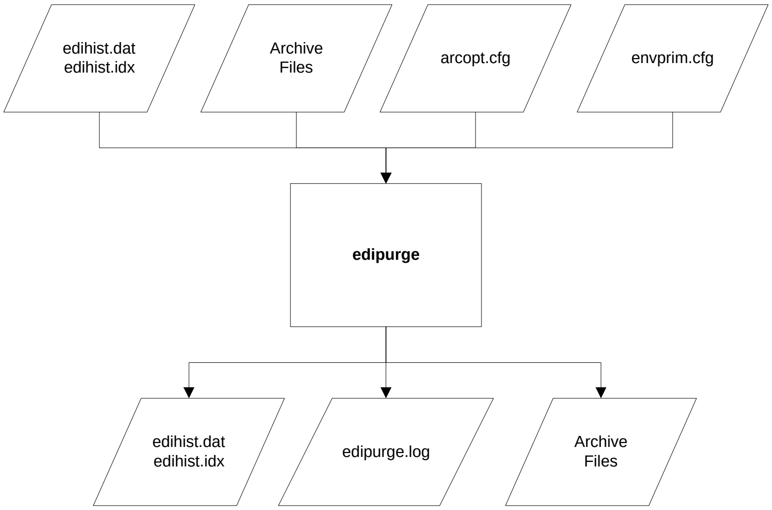
# Archive Purge Processing

## Purpose

Use this process to physically remove Audit records and associated EDI data files.

## Inputs and outputs

This illustration shows the inputs and outputs for the archive purge process.i



## What happens during the archive purge process

This table describes what happens during the process of purging the archive.

| WHEN… | THEN edipurge… |
|---|---|
| **edipurge** reads *envprim.cfg* | Locates the directory paths required for input/output and opens the file *edipurge.log*. |
| **edipurge** reads the Audit files *edihist.dat* and *edihist.idx*, one record at a time | Deletes unneeded records. **edipurge** deletes the associated EDI data file if no other record references the file. |
| The purge process is complete | Produces the file *edipurge.log* in the Gentran:Server Workstation *rpt* directory or the Gentran:Server for UNIX User’s (*temp)* directory. This file contains the status messages from the program execution. |

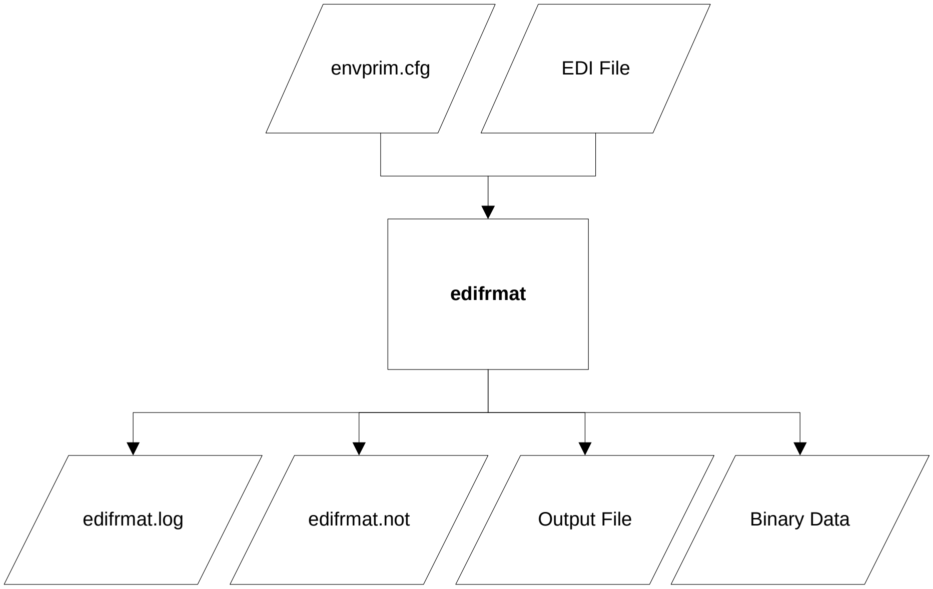
# edifrmat Processing

## Purpose

The **edifrmat** program replaces segment terminators with newline characters and performs other edit functions on the inbound data.

## Inputs and outputs

This illustration shows the inputs and outputs for the **edifrmat** process.



(Continued on next page)

## What happens during edifrmat processing

This table describes what happens during the **edifrmat** process.

| **WHEN edifrmat…** | **THEN edifrmat…** |
|---|---|
| Reads the *envprim.cfg* file | ◗ Locates the directory paths required for input/output and opens the file *edifrmat.log*.<br>◗ Creates the log file in the *rpt* directory for Gentran:Server Workstation or the User's *(tmp)* directory for Gentran:Server for UNIX. |
| Reads the EDI data into memory one block (1024 bytes) at a time | Searches each block of data read before an interchange header is found for possible headers.<br>**Note**<br>For each possible header **edifrmat** finds, it checks the values of separators, sub-element separators, and terminators. If any of these values are invalid, **edifrmat** reports the errors in the *edifrmat.log* file. |
| Finds a valid interchange header | ◗ Processes each block of data until it finds the interchange trailer. The process replaces terminators with newline characters.<br>◗ Reports in the *edifrmat.log* file when the end of the file is reached before an interchange trailer is found.<br>**Note**<br>DOS newline has a hex value of 0d 0a and UNIX newline has a hex value of 0a.<br>(Continued on next page) |

| **(Contd)**<br>**WHEN edifrmat…** | **THEN edifrmat…** |
|---|---|
| Encounters any non-EDI characters before a valid interchange header and after the interchange trailer | Writes the non-EDI data to a file named *edifrmat.not*.<br>**CAUTION**<br>**If the -I parameter is used, edifrmat writes all newline characters to this file. If your input file contains sets received with newline characters as terminators and you use the -I parameter, the newline characters are stripped from the original data. If this happens, edifrmat cannot replace the terminator, and the entire set is written to *edifrmat.not*.**<br>**Note**<br>The edifrmat process overwrites the *edifrmat.not* file each time it runs unless the EDIFRMATNOT environment variable precedes the edifrmat command or the -a parameter is used. If the EDIFRMATNOT environment variable is used, **edifrmat** writes a new *edifrmat.not* file each time. If the -a parameter is used, **edifrmat** appends the existing *edifrmat.not* file. |
| Encounters binary data within a segment | Writes the data to the output file that you specified for binary data.<br>(Continued on next page) |

| (Contd)<br>**WHEN edifrmat…** | **THEN edifrmat…** |
|---|---|
| Generates the output file | Names the file using the convention <*input file name*>*.FRM* if run from the menu or a user-defined name if run from the command line. You define the destination. |
| Completes processing | Generates the file *edifrmat.log* in the *rpt* directory for Gentran:Server Workstation or the User's *(tmp)* directory for Gentran:Server for UNIX. This file contains the status messages generated during the execution of the program.<br>**Gentran:Server Workstation**<br>**edifrmat** places binary data found in the input file into the file *TMP#.$$$*. This file is located in the current working directory.<br>**Gentran:Server for UNIX**<br>On UNIX, **edifrmat** creates the binary data file in the UNIX temporary directory as set in the *.profile*. The UNIX operating system assigns it a unique name which replaces the binary data in the input file as well as the output (record file layout) file.TMP#.$$$ - # = up to 65,535. |

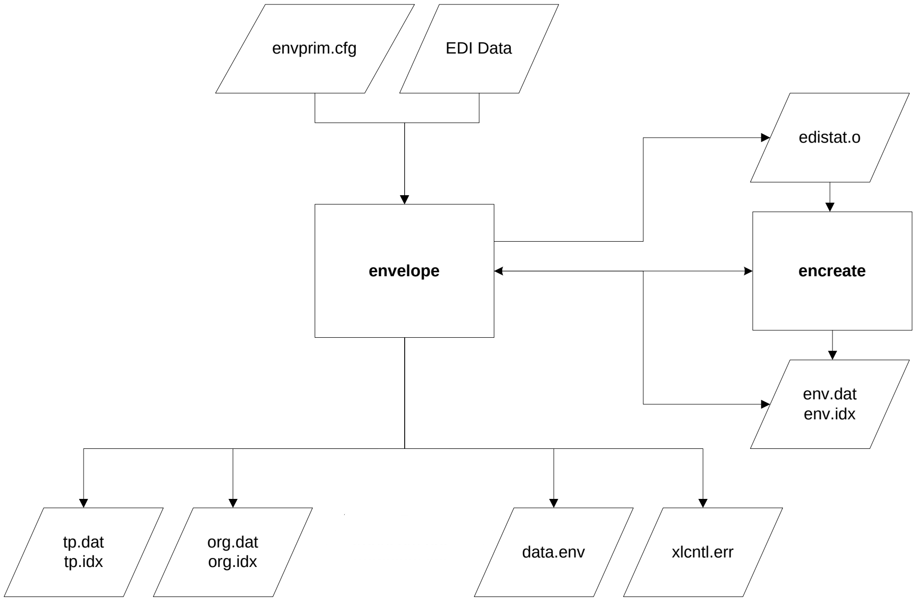
# envelope Processing

## Purpose

Use this outbound process to combine like interchanges in the outbound file.

## Inputs and outputs

This illustration shows the **envelope** process.



## What happens during envelope processing

This table describes the results of actions taken during envelope processing.

| WHEN… | THEN… |
|---|---|
| The translation process creates file *edistat.o* | *edistat.o* contains a flag indicating whether or not “One Interchange Per File” is selected in the Trading Partnership record. |
| **envelope** calls **encreate** | **encreate** creates an empty <*data*>.*env* file. |
| **envelope** reads the *edistat.o* file to determine which sets are flagged for enveloping | **envelope** writes this information to the files *env.dat* and *env.idx*. |

(Continued on next page)

| (Contd)<br>WHEN… | THEN… |
|---|---|
| **envelope** reads the EDI data files one record at a time | **envelope** reads *env.dat* and *env.idx* and places the data into the file <*data*>*.env*.<br>Whenever possible, **envelope** keeps the sets in order by combining the interchange headers.<br>The program updates the Trading Partnership records, organization records (if globally maintaining control numbers), and *edistat.o* file with the interchange control numbers.<br>The program then writes the EDI data in <*data*>*.env* to the output file. |
| The system produces or appends to the file *xlcntl.err* | The program loads *xlcntl.err* with the status and error messages that resulted from program execution. Messages include information about how many envelopes were removed, sender and receiver IDs, and the output file name. |

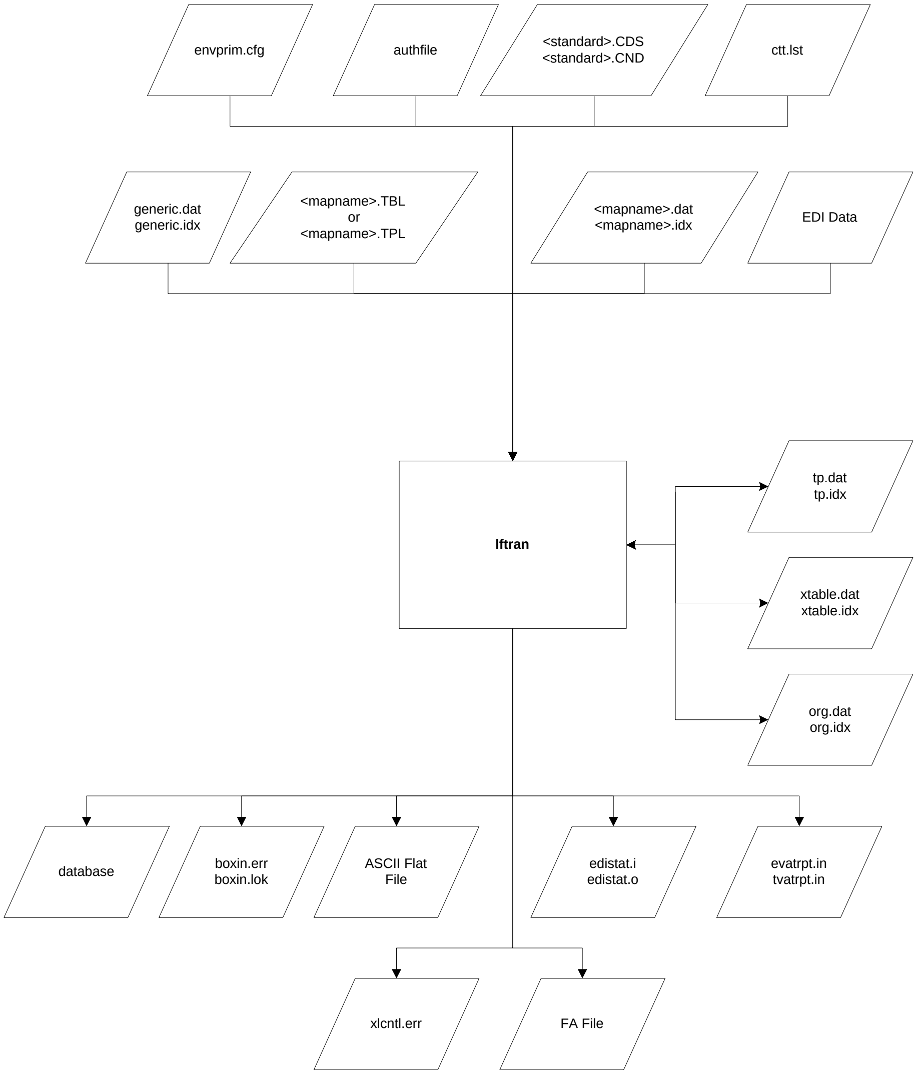
# Inbound Translation Processing

## Purpose

Use this process to convert data from an EDI Standard format to another format, such as an application data format or another EDI data format.

## Inputs and outputs

This illustration shows the inputs and outputs for the inbound translation process. The type of map used determines which inputs apply to the translation.



(Continued on next page)

## What happens during inbound translation

This table describes what happens during the inbound translation process.

| WHEN… | THEN lftran… |
|---|---|
| The translator opens *envprim.cfg* | Locates the directory paths required for input/ output and opens *xlcntl.err.* |
| The translator opens Authfile | Verifies the expiration date. |
| The translator opens the Organization Files | Locates the global control numbers for outbound translation. |
| The translator opens the Trading Partnership record | Opens the *tp.dat* and *tp.idx* files |
| The user has selected code validation and conditional compliance checking for translation | Opens the <*standard*>*.CDS* and the <*standard*>*.CND* files. |
| The translator opens the input file to obtain information to locate a TP record | Reads the data until the sets header segment is located. Then, locates the TP record based on the six key fields: Interchange Sender ID, Interchange Receiver ID, Group Sender ID, Group Receiver ID, Standard Version, and Set.<br>If there is a CTT and the Visual Mapper is used, lftran reads the file *ctt.lst* to determine what to sum for sets with a CTT segment.<br>In the Application Integration Editor, you must map the CTT yourself. |
| The translator processes the EDI data one segment at a time | Reads a segment for each set and searches for the mapping table file for the corresponding mapping.<br>If lftran finds the mapping, it then processes each element according to the mapping instructions. |
| The translator is unable to process any EDI data | Writes the un-translated EDI data to the *boxin.err* file. |

(Continued on next page)

| (Contd)<br>WHEN… | THEN lftran… |
|---|---|
| The translator is temporarily unable to process any EDI data because the required Trading Partnership record is locked | Writes the un-translated EDI data to the *boxin.lok* file.<br>This file is then available for the user to retry translating after the Trading Partnership record is unlocked. |
| The translator cannot find mapping instructions for an element | Goes to the next element.<br>Processes any substitutions (synonyms), sub-strings, data conversions, or calculations needed for each mapped element. When the data is ready to be written, one of the following can occur:<br>◗ If this is the first mapping to this record, lftran allocates memory, writes the element to the appropriate place in the memory record, and sets a flag to indicate this field has data in it.<br>◗ If the record exists in memory and the field has data in it, lftran creates a new memory record and copies retained data to the memory record. The program then places the element in the appropriate place in the memory record and sets a flag to indicate this field has data in it.<br>◗ If the record exists in memory and the field is empty in memory, lftran places the element in the appropriate place in the memory record and sets a flag to indicate this field has data in it. At the end of the set, lftran writes all memory records with data in them to the file. |
| The user has selected to save status records | Writes the files *edistat.i* and *edistat.o* for each set. |
| The input is EDIFACT data and the user wants to create a VAT report | Writes data to *evatrpt.in*. |
| The user has selected to generate acknowledgments | Writes any functional acknowledgments/ CONTRLs to the FA File. |
| Translation processing is completed | Closes the file *xlcntl.err*, which contains the status and error messages as a result of program execution. |

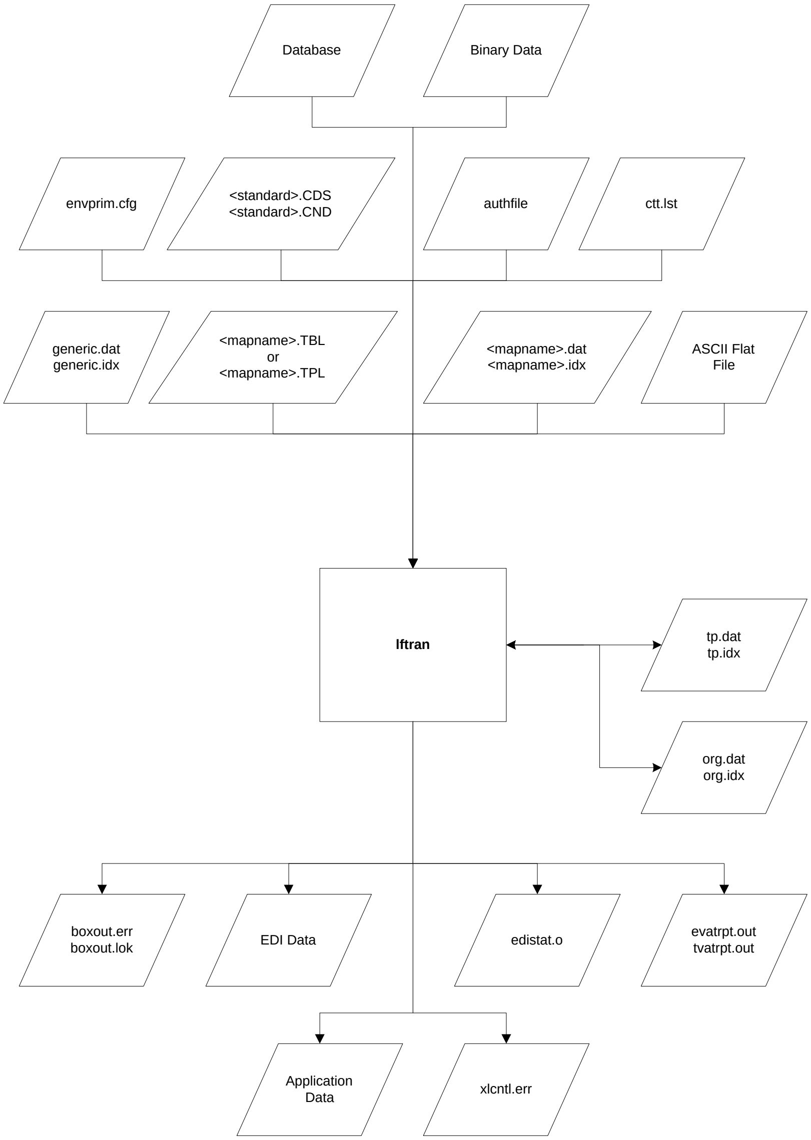
# Outbound Translation Processing

## Purpose

Use this process to convert data from an application format to another format, such as an EDI Standard data format or another application data format.

## Inputs and outputs

This illustration shows the inputs and outputs for the outbound translation process. The type of map used determines which inputs apply to the translation.



(Continued on next page)

## What happens during outbound translation

This table describes what happens during the outbound translation process.

| WHEN… | THEN lftran… |
|---|---|
| The translator opens *envprim.cfg* | Locates directory paths required for input/ output and opens *xlcntl.err*. |
| The translator opens authfile file | Verifies the expiration date. |
| The translator opens organization files | Maintains control numbers if global control numbers are maintained. |
| The translator opens the Trading Partnership File | Opens the *tp.dat* and *tp.idx* files. |
| The user has selected code validation and conditional compliance checking for translation | Opens the <*standard*>*.CDS* and the <*standard*>*.CND* files. |
| The translator opens the user-defined file | Obtains information to locate a TP record.<br>Reads the data until the set's header segment is located. |
| Binary data is being translated (841) | Opens the file containing the binary data. |
| The CTT is being generated | Reads the file *ctt.lst* to determine which segment and element to count and sum.<br>Processes the record file layout data one record at a time. |
| For each set, the translator reads a record and searches for the mapping table file for the corresponding mapping. | One of the following occurs.<br>◗ If lftran finds the mapping, it processes each field according to the mapping instructions.<br>◗ If lftran does not find mapping instructions for a field, it goes to the next element.<br>For each mapped field, lftran processes any substitutions (synonyms), sub-strings, data conversions, or calculations needed.<br>(Continued on next page) |

| (Contd)<br>WHEN… | THEN lftran… |
|---|---|
| The data is ready to be written; one of the following can occur: | |
| a. If this is the first mapping to this record | Allocates memory, writes the element to the appropriate place in the memory record and sets a flag to indicate that this field has data in it. |
| b. If the record exists in memory and the field has data in it | Creates a new memory record and copies retained data to the memory record. lftran then places the element in the appropriate place in the memory record and sets a flag to indicate that this field has data in it. |
| c. If the record exists in memory and the field is empty in memory | Places the element in the appropriate place in the memory record and sets a flag to indicate this field has data in it. At the end of the set, lftran writes all memory records with data in them to the file. |
| All the elements for a segment have been processed | 1. Builds the segment.<br>2. Places the segment ID in front and places the element separators, indicated in the Trading Partnership record, between each element.<br>3. Strips any extra trailing separators and, if there is data in the segment, places the terminator at the end of the segment.<br>4. Writes the segment to the output file. |
| The translator is unable to process any record file layout data | Writes the un-translated record file layout data to the *boxout.err* file. If the translator is temporarily unable to process any record file layout data because the required Trading Partnership record is locked, it writes the un-translated record file layout data to the *boxout.lok* file. This file is then available for the user to retry translating after the Trading Partnership record is unlocked.<br>lftran also writes to the file *edistat.o* for each set. |

(Continued on next page)

| (Contd)<br>**WHEN…** | **THEN lftran…** |
|---|---|
| The input is EDIFACT data and the user wants to create a VAT report | Writes data to *evatrpt.out*. |
| Translation processing is run | Produces or appends to the file *xlcntl.err*. This file contains the status and error messages resulting from program execution. |

# Translation Summary Report Processing

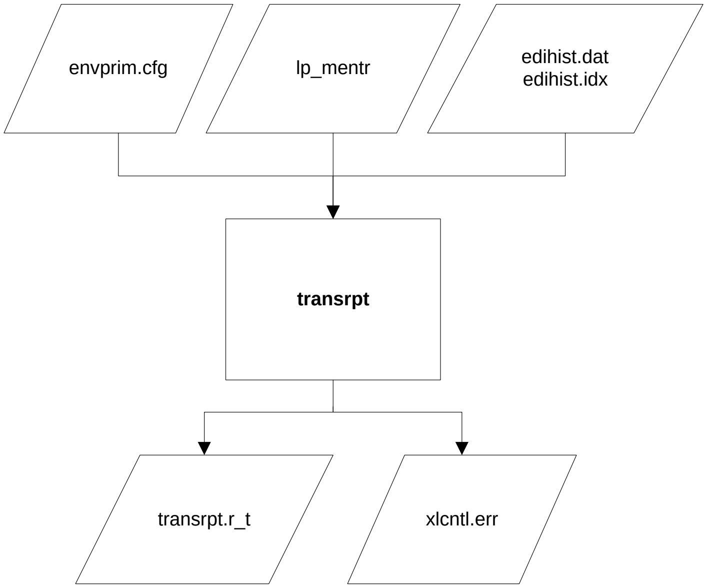
## Purpose

Use this process to read the Audit files (*edihist.idx* and *edihist.dat*), and print the status of translations, either single or selected via date range.

**Note**
This feature is available only for EDI data when converting from standard to application and XML to standard.

## Inputs and outputs

This illustration shows the inputs and outputs for the Translation Summary Report process.



(Continued on next page)

## What happens during Translation Summary reporting process

This table describes what happens during the Translation Summary reporting process.

| WHEN… | THEN… |
|---|---|
| The user selects parameters against which Audit records will be compared | The system stores the selected parameters. |
| The system reads *envprim.cfg* | **transrpt** locates the directory paths required for input/output and opens the files *transrpt.r_t* and *xlcntl.err*. |
| The system opens the audit files (*edihist.idx* and *edihist.dat*) and reads one record at a time. | **transrpt** checks each record against the user parameters. If they match, it reformats the record for the report and writes it to *transrpt.r_t* in the user directory. |
| The transrpt program writes data to *transrpt.r_t* and the user specified to print the report | **transrpt** reads the file *lp_mentor.bat* (DOS) or *lp_mentor* (UNIX) and routes the report to the printer. |
| Translation summary report processing is complete | **transrpt** generates the file *xlcntl.err,* which contains the status messages resulting from program execution. The *xlcntl.err* file is written to the *rpt* directory for Gentran:Server Workstation or the User’s *(tmp)* directory for Gentran:Server for UNIX. |

# Application Integration Processes

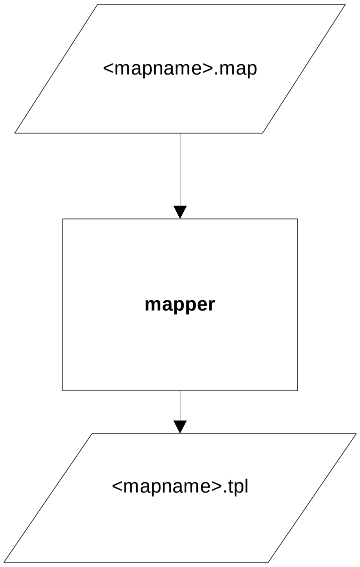
## Map Compiler Processing

**Purpose**

Use this process to convert the map file output from the Application Integration Mapper to the translation object format, performing extensive checks in the process.

**Inputs and outputs**

This illustration shows the input and output for the Map Compiler process.



(Continued on next page)

## What happens during map compiler processing

This table describes what happens during the Map Compiler process.

| WHEN… | THEN… |
|---|---|
| The compiler converts the *source map* file from the mapper (*.map*) format into the compiled *translation object* (*.tpl*) format | The compiler verifies that all extended rules are correct and free of syntax errors.<br>The following items in the source map are converted into a form that the translator can understand and the system compiles a translation object and names it *<mapname>.tpl*.<br>◗ Visual Map objects (groups, segments/ records, fields, etc.)<br>◗ Standard rules<br>◗ Extended rules<br>◗ Constants<br>◗ Field-level conditional relationships<br>◗ Code lists<br>◗ Syntax tokens |

**Reference**
See the How to Compile a Map topic in the Designing your Map chapter in the Gentran:Server *for UNIX and Workstation Application Integration User's Guide* for more information on compiling maps.

# Visual Mapper Processes

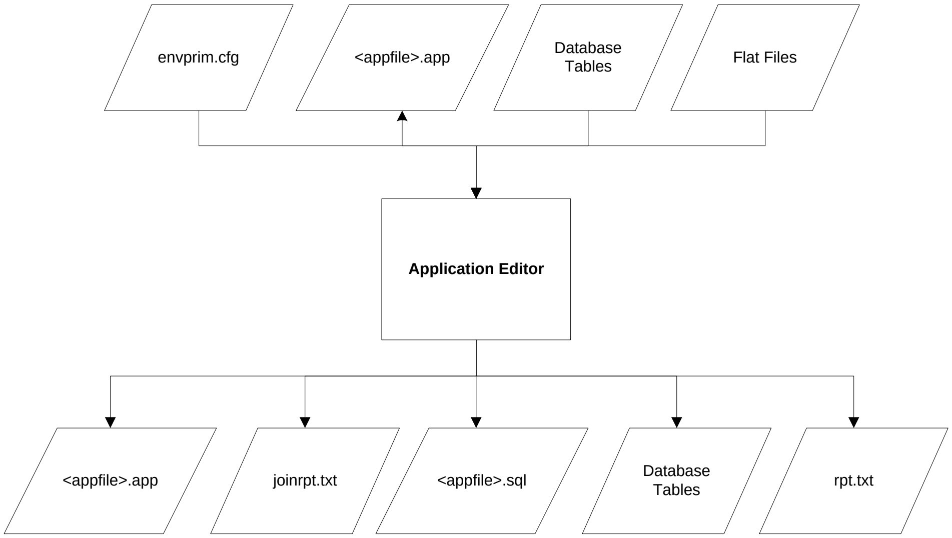
## Application Editor Processing

### Purpose

The Application Editor is used to describe records, fields, and loops in an application file.

### Inputs and outputs

This illustration shows the inputs and outputs for the Application Editor.



### What happens during Application Editor processing

This table describes the results of actions taken in the Application Editor process.

| WHEN… | THEN… |
|---|---|
| The application editor reads *envprim.cfg* | The application editor determines directory paths required for input/output. |
| A user opens the application editor to create a new application file | The system creates an empty *.app* file in memory on the UNIX client computer or the Windows PC. |

(Continued on next page)

| (Contd)<br>WHEN… | THEN… |
|---|---|
| A user loads an existing *.app* file | **Gentran:Server Workstation**<br>The application editor loads an existing *.app* file from the *apps* directory into the working area on the Windows computer. It also loads the *.app* file into memory on the Windows computer and displays it in the Application Editor window.<br>**Gentran:Server for UNIX**<br>The application editor loads an existing *.app* file from the *apps* directory on the UNIX host to the user's work area on the host. It also loads the *.app* file into memory from the user's work directory on the host and displays it in the Application Editor window on the client. |
| The application editor reads database tables | The application editor creates a *.app* file in memory on the Windows computer and displays it in the Application Editor window. |
| The application editor reads a flat file which describes the application | The application editor automatically creates a new *.app* file that contains records and fields matching the flat file description. The *.app* file is in memory on the Windows computer. |
| A user edits the *.app* file | The system stores the changes the user makes to the *.app* file in memory on the Windows computer. |
| A user saves changes to the *.app* file | The application editor saves changes to the *.app* file. If the user selected the Save As command, the *.app* file is saved to the name the user selects. |
| Prints the application | The system generates the file *rpt.txt* and writes to the *rpt* directory specified in *envprim.cfg*. |

# Standard/IG Editor Processing

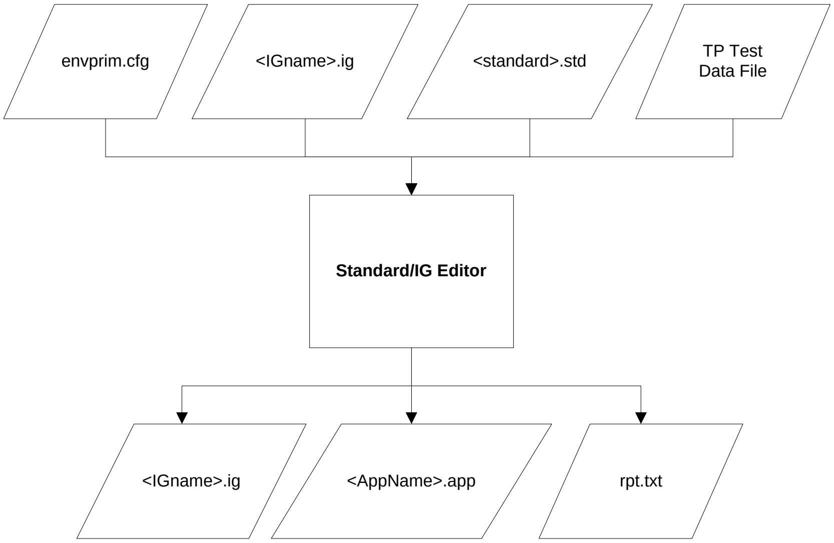
## Purpose

The Standard/IG Editor is used to describe segments, elements, and loops in a trading partner’s data. If a trading partner provides test data files, the Standard/IG Editor may also be used to AutoTrim a standard by comparing and keeping only the segments used in the trading partner’s test data file.

You can also use the Standard/IG Editor to modify an existing standard to create a new standard.

## Inputs and outputs

This illustration shows the inputs and outputs for the Standard/IG Editor.



(Continued on next page)

## What happens during Standard/IG Editor processing

This table describes the results of actions taken in the Standard/IG process.

| WHEN… | THEN… |
|---|---|
| The system reads *envprim.cfg* | The Standard/IG process locates the directory paths required for input/output. |
| The Standard Editor loads an existing *.ig* file | **Gentran:Server Workstation**<br>The application editor loads an existing *.ig* file from the *igs* directory into memory on the Windows computer and displays it in the Standard/IG Editor window.<br>**Gentran:Server for UNIX**<br>The application editor loads an existing *.ig* file from the *igs* directory on the UNIX host to the user's work area on the host. It also loads the *.ig* file into memory on the Windows client and displays it in the Standard/IG Editor window. |
| User loads an already existing implementation guide into memory and edits it | The Standard Editor closes the *.ig* file after loading it into memory.<br>All changes to the data occur in memory. The user must save the file (*<filename>.ig)* and replace existing the file of same name (Save) or as a new name (Save As) to save changes. |
| User loads a new standard into memory and edits it | The system reads the standards from the directory specified in the *envprim.cfg* file.<br>The new standard exists only in memory. The user must save it to an *.ig* file to use it again later.<br>(Continued on next page) |

| (Contd)<br>WHEN… | THEN… |
|---|---|
| User copies an existing standard to create a new standard (applies only to Visual Mapper) | The system attempts to match the standard name that the user supplies with a standard from the directory specified in the *envprim.cfg* file. If no match is found, the Standard Editor prompts for creating a new standard by copying one that does exist in the standards directory.<br>When the user copies the standard under a new name, the Standard Editor adds the name of the new standard to the standards directory. |
| User selects AutoTrim | Gentran:Server automatically creates an implementation guide from test data provided by your trading partner.<br>The user then saves the edited standard to a *.ig* file. |
| User selects AutoCreate Application Description | The system writes the new *.app* file to the application directory specified in *envprim.cfg.*<br>AutoCreate can automatically create an application description. The *.app* file will assume the name of the Standard/IG File (<*IG name>.app*). |
| User discards items from implementation guide | The implementation guide file retains items, but inactivates them. All file processing bypasses the items. |
| User prints standard or implementation guide | The system generates the file *rpt.txt* and writes it to the default Check-in/out directory specified in *envprim.cfg*. |

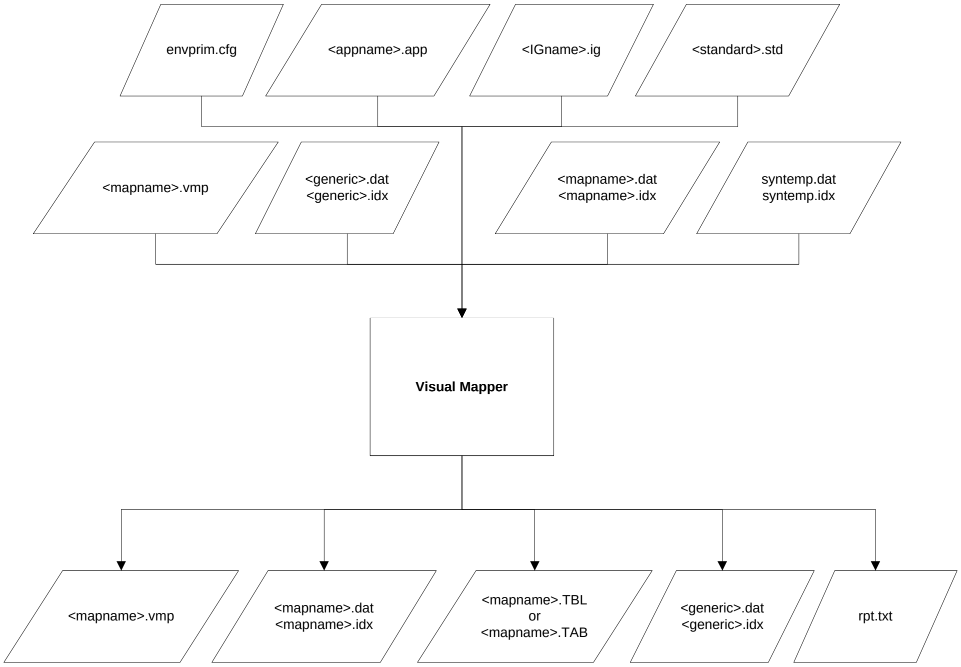
# Visual Mapper Processing

## Purpose

The Visual Mapper maps the relationship between a user's data and the trading partner's data. If a trading partner provides test data files for a standard, the Visual Mapper can attempt to map the user's data and the trading partner's data automatically.

## Inputs and outputs

This illustration shows the inputs and outputs for the Visual Mapper.



(Continued on next page)

## What happens during Visual Mapper processing

This table describes the results of actions taken in the process that uses the Visual Mapper.

| WHEN… | THEN… |
|---|---|
| The system reads *envprim.cfg* | The Visual Mapper locates the directory paths required for input/output. |
| User loads an existing *.app* into memory and maps it | The Visual Mapper makes any changes to the data only in memory; it closes the *.app* file after loading it into memory.<br>Once user saves changes, the Visual Mapper writes the changes to a map file instead of the originating *.app* file. (To save the changes to the *.app* file, the user must open the App Editor from within the Visual Mapper and save the *.app* file there.) |
| User loads an existing *.ig* into memory and maps it | The Visual Mapper keeps all changes to the data in memory and closes the *.ig* file after loading it into memory.<br>The user can save changes to just the map or to the map and the *.ig* file.<br>If the user saves changes only in the map, the system writes the changes only to the map file. If the user saves in the Standard/IG Editor as well, the system writes the changes to the *.ig* file, too. |
| User loads a new standard into memory and maps it | The system reads standards from the directory specified in the *envprim.cfg* file. The new standard will exist only in memory.<br>The Visual Mapper saves changes to the standard to an *.ig* file. |
| User loads an existing map file *(.vmp)* into memory and edits it | The Visual Mapper makes all changes to the data in memory only. It closes the map file after loading it into memory.<br>The user can save changes to the map file by selecting **Save**, or to a different map file name by using **Save As**. |

(Continued on next page)

| (Contd)<br>WHEN… | THEN… |
|---|---|
| Synonyms[a] exist for a map | The system creates two temporary Synonym files, *syntemp.dat* and *syntemp.idx*, to hold the synonyms.<br>If no synonyms exist for the map, the system creates two empty temporary Synonym files to handle a possible new synonym list. These files are stored in the temporary directory specified in *envprim.cfg*.<br>If the user makes any changes to the synonyms, the system stores the changes in the temporary Synonym files.<br>If the user chooses to save the changes, the system updates the original *.dat* and *.idx* Synonym files. |
| A user compiles a map | The system generates a mapping table file with the same name as the map but with a new extension, *.TBL*. The compiler writes the mapping table file to the directory specified for map files in *envprim.cfg*. |
| A user defines synonyms for specific map elements | The system writes the specific synonyms and generic synonyms to ISAM files.<br>◗ The specific synonyms are written to *<mapname>.idx* and *<mapname>.dat*.<br>◗ The generic synonyms are written to *generic.idx* and *generic.dat*. |
| A user prints a map | The system generates the file *rpt.txt* and writes it to the temporary directory specified in *envprim.cfg*. |

a. Maps may use synonyms to substitute one name for another (for example, colour for color). Synonyms specific to a map are stored in two files, <map name>.idx and <map name>.dat, where <map name> is the same file name as the map file. Maps may also use a generic synonym list, not specific to any map. Generic synonyms are stored in two files *generic.dat* and *generic.idx*. The specific and generic synonym files are written to the directory specified for maps in *envprim.cfg*.

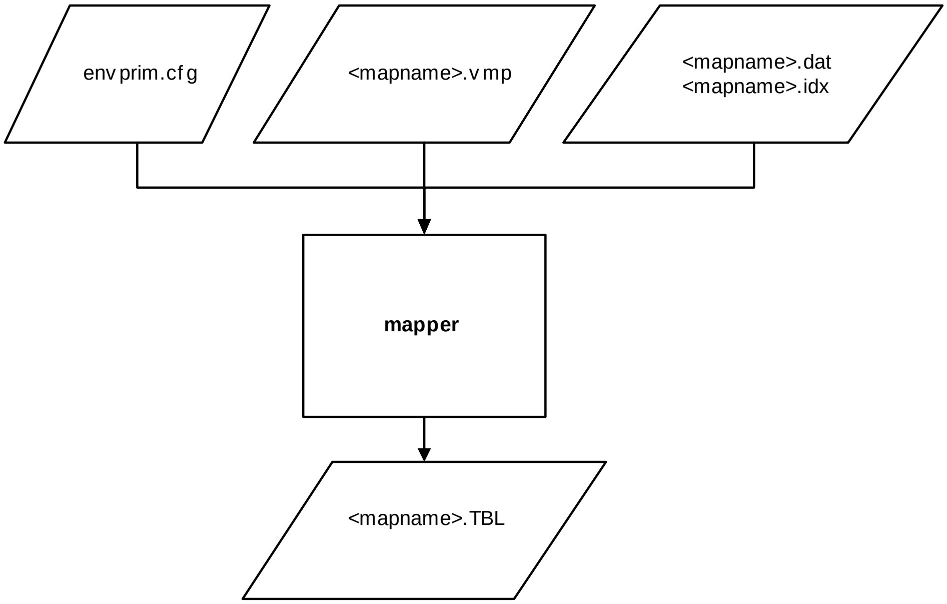
# Visual Map Compiler Processing

**Purpose**

The Map Compiler process converts the map file output from the Visual Mapper to a compiled map format, sorting the mapping instructions and doing extensive checking in the process.

**Inputs and outputs**

This illustration shows the inputs and outputs for the Map Compiler process.



(Continued on next page)

## What happens during map compiler processing

This table describes the results of actions taken in the Map Compiler process.

| WHEN… | THEN… |
|---|---|
| The compiler converts the map file from the Mapper (*vmp*) format into the map table format | The compiler checks that:<br>◗ Character values entered in mapping instructions are in the correct context.<br>◗ All selected source and destination elements are mapped.<br>◗ All required symbols are in the expression.<br>◗ The mapping instruction preserves the integrity of the data type.<br>◗ The destination is valid for the group.<br>◗ The source is valid for the group.<br>◗ There are no circular references made in a macro. There are no duplicate mappings. |
| | ◗ Operations are:<br>— Used in context<br>— Complete and no arguments are missing.<br>◗ Symbols (operators) used in mapping instructions are:<br>— Valid<br>— Used in context<br>— In the correct place within the syntax<br>— Valid for the operation<br>— Valid arguments for the operation<br>— Using data type that is valid for the argument.<br>◗ The value entered is:<br>— A valid operator, source, or destination<br>— A valid data type<br>— Flagged as used<br>— Valid for the expression. |

(Continued on next page)

| (Contd)<br>**WHEN…** | **THEN…** |
|---|---|
| Map is compiled | The system compiles a compiled map for the current platform and names it *<mapname>.TBL*. The file is stored in the working directory on the host. Use the Check-in process to move the file into the *maps* directory. |

CHAPTER
2

# Environment Variables

**Contents**

# Overview

## Introduction

This chapter contains descriptions of the environment variables used by Gentran:Server. The chapter describes both environment variables used to control UNIX host environments and environment variables used to control Gentran:Server processes. It also describes environment variables that apply only to specific product levels.

## When environment variables are set

The Gentran:Server for UNIX setup process prompts you for the information needed to set the required environment variables. The setup process on Gentran:Server for UNIX validates the values you provide and then creates a file containing all of the required variables.

You may need to set additional environment variables to control Gentran:Server commands such as **lftran**. See the sections in this chapter for descriptions of environment variables to control **lftran** or **edifrmat**.

### References

See the Command Reference chapter in this guide for instructions for configuring environment variables for listed commands.

See the Gentran:Server *for UNIX Data Flow Administration Guide* for instructions for configuring the behavior of data managers.

## How environment variables are used in UNIX

In the UNIX operating system, whenever you execute a shell program, the shell creates an entirely separate environment for that program to run in. Environment variables hold values that are transferred to the appropriate shell variables when a **setenv** command is executed. Once the values are in the shell variables, they provide certain defaults (like your TERMinal type and EDITOR) that will be in force for each UNIX session.

### Comment

In the C shell, changes in form setenv NAME value are added to the current environment. In the Bourne shell, environment variables are set by equating NAME to value, without using the keyword setenv, and then by exporting the environment variable.

| | |
|---|---|
| C shell | setenv NAME value |
| Korn or Bourne shell | NAME=value; export NAME |

(Continued on next page)

## Where UNIX environment variables are set

Environment variables for Gentran:Server should be set in the profile of the user that starts the processes for each environment within a particular installation. For example, in an installation with one copy of the Gentran:Server Security Administration Utility controlling two environments, named test and production, the appropriate environment variables need to be set within the profiles of:

- The user who owns the directory where the Security Administration Utility is installed and who starts the broker (**ltb_server**) process.
- The user who owns the directory where the test environment is installed, and who starts the **mhs_server**, **mhp_server**, and foreground manager (**fmgr**) processes for that environment.
- The user who owns the directory where the production environment is installed, and who starts the **mhs_server**, **mhp_server**, and **fmgr** processes for that environment.

In the Korn and Bourne shells, these variables are set within the *.profile* file. In the C shell, these variables are set within the *.login* file.

**Note**
The security setup process collects required environment variables in the file *gentran.security.vars* in the home directory of the user that installs security.

The EDI software setup process collects required environment variables in the file *gentran.server.vars* in the home directory of the user that installs the EDI software.

Because they have been tested by the setup processes, we recommend you use the values from those files.

## Processes

Each process has a set of environment variables associated with it. These variables can be used by the process or set within the process.

Environment variables are inherited by sub processes. Whenever you start a new process as a child of another, UNIX sets the environment variables of the child to those of its parents if you used the **setenv** or export command described above.

Variables set within a child process are only set for that child and do not affect the parent process. When a child process ends, the environment variable settings are not retained, and are therefore must be set again for the next child process.

## Names

Environment variables typically have all uppercase names. For example, TERM or PATH.

# General Environment Variables

## Variable names and descriptions

This table describes the environment variables used by the Gentran:Server.

All optional environment variables are clearly marked. All unmarked variables are required.

Environment variables for specific product levels are clearly marked. All others apply to all Gentran:Server product levels.

| Name | Description |
|---|---|
| EDI_ARCHIST | Establishes an active archive period for any data manager that is not configured for the number of days in the active archive period.<br>◗ The **default** period is **90** days<br>◗ This environment variable is **optional**.<br>◗ Applies to the Gentran:Server for UNIX with EC Workbench product level and higher. |
| EDI_ARCH_NEWLOC | Names the directory that holds retrieved archived data. If this environment variable is set, the specified directory is used as the default in the New Location field on the Archive Query screen.<br>◗ This environment variable is **optional**.<br>◗ Applies to the Gentran:Server for UNIX with EC Workbench product level and higher. |
| EDI_LOG | When set to NO this variable precludes entry of all normal zero '0' messages from the logs produced by scripts and data managers. All abnormal non-zero messages are still logged.<br>◗ This environment variable is **optional**.<br>◗ Applies to the Gentran:Server for UNIX with Process Control Manager product level and higher. |
| EDI_MAILBOX | This is the directory that contains the Advanced Data Distribution product files.<br>◗ Applies to Gentran:Server for UNIX with Advanced Data Distribution and higher product levels.<br>(Continued on next page) |

| (Contd)<br>Name | Description |
|---|---|
| EDI_MAILDET | This is the directory to which Advanced Data Distribution Life Cycle detail is sent. This variable allows the system to create the Advanced Data Distribution Life Cycle file.<br>◗ Applies to Gentran:Server for UNIX with Advanced Data Distribution and higher product levels. |
| EDI_ROOT | The home directory of Gentran:Server.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| EDITOR | This is the preferred editor.<br>◗ The default setting is **vi**<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| ENV_ROOT | The directory containing the configuration file *envprim.cfg*.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| LIB_PATH | The shared library path for AIX (IBM).<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| LD_LIBRARY_PATH | The shared library path for SUN.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| NAMEBROKER | The IP address or alias of the machine on which the broker is located.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |

(Continued on next page)

| (Contd)<br>Name | Description |
|---|---|
| PATH | For you to use the Gentran:Server, your system administrator must have added the $EDI_ROOT and $EDI_ROOT/bin colon-separated entry to the PATH environment variable in your user's *.profile* or *.login* shell script.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| SADMIN_ROOT | The home directory of the Gentran:Server Security Installation.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| SHLIB_PATH | The shared library path for HP.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| TERM | This is used to define your system's terminal environment.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| umask | The default file permissions used when creating new files.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| VVTERM | This is used to define Gentran:Server screen system's terminal environment differently from your system's terminal environment.<br>◗ This environment variable is **optional**<br>◗ Applies to all levels of Gentran:Server for UNIX. |
| VVTERMCAP | This is the path to the *vvtermcap* file (usually $*SADMIN_ROOT/vvtermcap*). This file contains descriptive information about your terminal such as key and color mapping.<br>◗ This environment variable is required<br>◗ Applies to all levels of Gentran:Server for UNIX. |

# edifrmat Environment Variables

This table describes environment variables you can set to control how **edifrmat** handles errors and output.

**Note**
These environment variables apply to Gentran:Server Workstation as well as all product levels of Gentran:Server for UNIX.

**Reference**
See the Command Reference chapter in this guide for instructions for setting these variables.

| Environment Variable | Function |
|---|---|
| XLCNTLERR | Prevents appending to the *xlcntl.err* file. Instead, Gentran:Server creates a new file each time. |
| EDIFRMATNOT | Prevents overwriting the *edifrmat.not* file and losing data created earlier in the session. |

# lftran Environment Variables

The table describes environment variables you can set to control the output from **lftran**.

- These environment variables apply to Gentran:Server Workstation as well as all product levels of Gentran:Server for UNIX.
- Each variable prevents the system from appending a to a specific file. Instead, Gentran:Server creates a new file each time.

**Reference**
See the Command Reference chapter in this guide for instructions for setting these variables.

| This Environment Variable | Prevents appending to the file |
|---|---|
| AUDITI | *edistat.i* |
| AUDITO | *edistat.o* |
| BOXINERR | *boxin.err* |
| BOXOUTERR | *boxout.err* |
| BOXINLOK | *boxin.lok* |
| BOXOUTLOK | *boxout.lok* |
| XLCNTLERR | *xlcntl.err* |

CHAPTER
3

# Command Reference

**Contents**

## Overview

## Alphabetical Listing of Commands

# Overview

## Introduction

### In this chapter

This chapter describes how to use the Gentran:Server programs and scripts. The chapter lists the programs and scripts in alphabetical order. Each section describes the purpose, syntax, and any associated parameters for a single program or script. The sections on the formatter (**edifrmat)** and the translator (**lftran**) also explain how to set environment variables for those programs.

You can run some programs or scripts described in this chapter from the Gentran:Server program interface, either from a menu or through the Run Program tool. You may also include many of them, except **mtimer**, in a DOS batch file or a UNIX script.

**References**
For instructions in running programs from the Run Program tool, see Using the Run Program Tool.

For information on DOS batch files and UNIX scripts, see the Using the Task Scheduler chapter of the Gentran:Server *for UNIX and Workstation Application Integration Guide* or the Gentran:Server *for UNIX and Workstation Mapping and Translation Guide*, if you use the Visual Mapper.

**Note**
Some programs and scripts are available only for specific Gentran:Server product levels. Within each topic, check for product level qualifications.

(Continued on next page)

## Before you begin

Before you begin using this chapter, please note the syntax conventions used within it. This table lists the conventions that identify which parameters are required, optional, and exclusive for each program or script.

| Convention | Indication |
|---|---|
| Angle Brackets (<>) | The parameter represents a variable. |
| Brackets([]) | The parameter is optional. |
| Braces ({}) | You must enter at least one of the parameters in the set. |
| Slashes (/) | You can enter only one of the parameters in the set. |

Do not include the brackets, slashes, or braces in the command you enter.

## Example 1

This example shows the syntax for **dcheck**:

```
dcheck {-b/d/e/h/i/k/l/n/o/q/x/y} <isamfile>
```

This is an example of the command as a user might enter it on the command line:

```
dcheck -b tp
```

## Example 2

This example shows the syntax for **edifrmat**:

**edifrmat** {<infile>} {<outfile>} [-cp <config path>] -[abcdeilsv]

This is an example of the command as a user might enter it on the command line to reformat an x12 file received from Commerce Net:

```
edifrmat /usr/gentran/data/cnet /usr/gentran/data/x_12 -cp $EDI_ROOT -ale
```

# Key to Programs

## Introduction

The tables in this section group the programs and scripts by purpose. Each table item provides the purpose and name of a program or script. For more information about the specified program or script, use the table of contents to locate the topic.

## Audit and archive data

This table describes Gentran:Server's programs and scripts used to audit and archive data.

| IF you want to… | THEN use… |
|---|---|
| Purge archived data manager audit data | cl_arch |
| Archive translator EDI Data | ediarc |
| Purge the indexed translation audit files | edipurge |
| Extract archived translation data | edixtrct |
| Create new Life Cycle database records and load them with data | lcld |
| Retrieve data from a data manager archive and move the data into a specified directory | rtv_arc |
| Update existing Life Cycle database records with data | xlld |

## Create empty files

This table describes the programs and scripts used to create new empty files, and lists where you can find them within this chapter.

| IF you want to create… | THEN use… |
|---|---|
| Empty application/Trading Partnership Rules Table (*apptptbl.dat* and *apptptbl.idx*) | apptptblcreat |
| Empty Application/Trading Partnership Code Cross Reference Table (*appxref.dat* and *appxref.idx*) | appxrefcreat |

(Continued on next page)

| (Contd)<br>IF you want to create… | THEN use… |
|---|---|
| Empty Trading Partnership category value records (*catvalue.dat* and *catvalue.idx*) and category type records (*cattype.dat*) | catcreat |
| Empty Trading Partnership Contact Files (*contact.dat* and *contact.idx*) | cocreate |
| Empty Audit Files (*edihist.dat* and *edihist.idx*) | edicreat |
| Empty EDI envelope files(*env.dat* and *env.idx*) | encreate |
| Empty ISAM files | isops |
| Empty Trading Partner Organization files (*org.dat* and *org.idx*) | orgcreat |
| Empty TRADACOM supplementary Trading Partnership files (*tradacom.dat* and *tradacom.idx*) | tccreate |
| Empty Trading Partnership files (*tp.dat* and *tp.idx*) | tpcreate |
| Empty Standard Cross Reference Table files (*xtable.dat* and *xtable.idx*) | xcreat |
| Empty XML splitting element table for the specified level (*xmlspl<n>.dat* and *xmlspl<n>.idx*)<br>**Note**<br>You must have the XML translation option to use this command. | xmlspl<n>creat |
| Empty XML Trading Partnership Rules Table (*xmltptbl.dat* and *xmltptbl.idx*)<br>**Note**<br>You must have the XML translation option to use this command. | xmltptblcreat |
| Empty XML Cross Reference Table (*xmlxref.dat* and *xmlxref.idx*)<br>**Note**<br>You must have the XML translation option to use this command. | xmlxrefcreat |

(Continued on next page)

## Create new files that contain data

This table describes the programs and scripts used to create new files that contain data and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Create the Gentran:Server configuration file, *envprim.cfg* | create_envprim_cfg |
| Build a list of files whose names are unique for the first specified number of bytes | newtype |

## Edit and format files

This table describes the programs and scripts used to edit and format files, and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Replace, insert, or remove characters. This is an ASCII Text Editor program | atext |
| Reformat the data in the *dtlLog.err* file and present it in an easy-to-read format | dumpXMLReport |
| Reformat EDI data so that it can be processed by the translator | edifrmat Overview |
| Reset the interchange, group, and set control numbers in an EDI document, and update the audit and Trading Partnership records | edirsnd |
| Consolidate like interchange envelopes into one interchange envelope | envelope |
| Reformat an input file into the standard log file structure | putlog |
| Reset counters in Gentran:Server ISAM files | rif |

(Continued on next page)

## Generate reports

This table describes the programs and scripts used to generate and print reports, and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Create a VAT Report for interchanges containing EDIFACT invoices | evatrpt |
| Print a Functional Acknowledgment Reconciliation Report | fareport |
| Generate statistical reports for translation traffic on a specified date | tracker |
| Run a translation summary report | transrpt |
| Create a VAT Report for interchanges containing TRADACOMS invoices | tvatrpt |
| Create a status report of the maps within an environment | unu_maps |
| Run a report that shows the records in the Standard Cross Reference Table | xtablerpt |

## Miscellaneous functions

This table describes miscellaneous Gentran:Server programs and scripts.

| IF you want to… | THEN use… |
|---|---|
| Validate Gentran:Server environments that contain the optional ODBC database drivers | chckodbc |
| Purge log data | cleanlog |
| Calculate a date using the current date as a reference point | datecalc |
| Check and/or repair ISAM files | dcheck |
| Generate the mailbag identification code | genmbid |
| Generate a unique file identification code | genuniqid |
| Run the translator from the command line or in a script | lftran |
| Compile maps | mentcomp |
| Access the Gentran:Server menu system on the client | mentor |

(Continued on next page)

| (Contd)<br>IF you want to… | THEN use… |
|---|---|
| Schedule and execute commands in a batch file (DOS command line only) | mtimer |
| Display and delete records with a specified trading partner code simultaneously from the tp, dm, and ds_tptbl ISAM files | tpops |
| Modify application file descriptions used by dnld data manager | udf4dnld |

## Move files

This table describes the programs and scripts used to load and unload indexed files, and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Load and unload Application/Trading Partnership Rules records to or from an ASCII file | apptptblmv |
| Load and unload Application/Trading Partnership Cross Reference Table records to or from an ASCII file | appxrefmv |
| Load and unload Trading Partnership Category value records to or from an ASCII file | catmv |
| Load and unload Trading Partnership contact records to or from ASCII files | conmv |
| Load and unload audit files to or from an ASCII file | edimv |
| Create indexed files, or load and unload them to or from an ASCII file | isops |
| Load and unload Trading Partnership organization records to or from an ASCII file | orgmv |
| Load and unload synonym records to or from an ASCII file | synmv |
| Load and unload supplementary TRADACOM Trading Partnership records to or from an ASCII file | tcmv |
| Load and unload the indexed Trading Partnership records to or from an ASCII file | tpmv |
| Load and unload Standard Cross Reference Table records to or from an ASCII file | xtablemv |

(Continued on next page)

## Start or stop processes

This table describes the programs and scripts used to start and stop Gentran:Server processes, and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Stop an FTP daemon process | ftpshut |
| Start the **ltb_server** process on a UNIX host computer | startnb.sh |
| Start the **mhs_server** and **mhp_server** programs for specific environments | startrpc.sh |
| Restart all data managers marked to autostart with **fmgr** | startserver |
| Start Gentran:Server background processes | startserver.sh |
| Stop **mhs_server** and **mhp_server** processes for specific environments | stoprpcs.sh |
| Stop all active data managers, including **fmgr** | stopserver |
| Stop Gentran:Server background processes | stopserver.sh |

## View contents of files

This table describes the programs and scripts used to view data, and lists where you can find them within this chapter.

| IF you want to view… | THEN use… |
|---|---|
| The directories specified in the Gentran:Server configuration file | chck_cfg |
| General information for a given map table file | chck_tbl |
| Information from the Trading Partner contact records | continfo |
| The contents of *edistat.i* or *edistat.o* | ediscope |
| Envelope records in the file *enscope.log* | enscope |
| Information about environment processes registered with the broker | ltb_info |
| Information from the Trading Partnership database | tp_info |

(Continued on next page)

## Work with queues

This table describes the programs and scripts used to work with queues, and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Cancel outbound queued items for a specific mailbag ID | cancel_mbid |
| Remove an entry from a processing queue | srvr_deq |
| Add an entry to a processing queue | srvr_enq |

## Work with scripts

This table describes the programs and scripts used to work with scripts, and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Return 0 | ret0 |
| Return 1 | ret1 |
| Execute the statements in a script | smgr |

## Work with transaction registers

This table describes the programs and scripts used to work with transaction registers, and lists where you can find them within this chapter.

| IF you want to… | THEN use… |
|---|---|
| Delete Transaction Register entries older than a set date and time | clean_trn |
| Mark for removal unwanted records from the Transaction Register | deltrn |
| Remove marked records from the Transaction Register | isops |

# Using the Run Program Tool

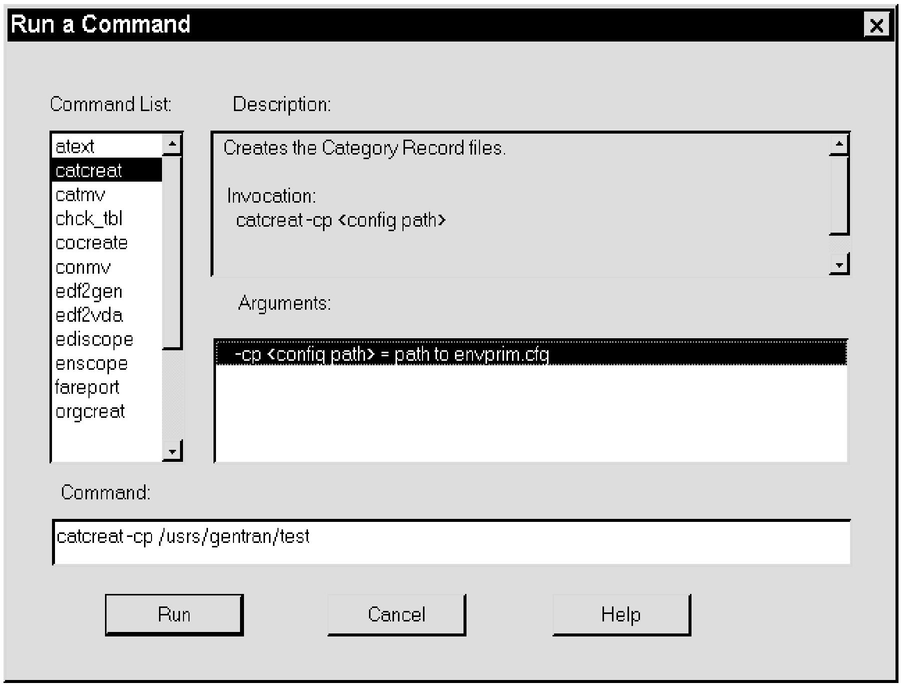
## Introduction

The Run Program tool enables you to start many of the commands from within the Gentran:Server interface. It provides an alternative to exiting Gentran:Server and typing commands on the UNIX or DOS command line. It helps you to enter the command using the proper syntax.

When you select **Run Program** from the Tool menu, Gentran:Server displays the Run a Command dialog box.

## Run a Command dialog box

This illustration shows the Run a Command dialog box.



Creates the Category Record files.

Invocation:
catcreat -cp <config path>

-cp <config path> = path to envprim.cfg

catcreat -cp /usrs/gentran/test

(Continued on next page)

## Run a Command dialog box fields and functions

This table lists the fields of the **Run a Command** dialog box and describes their functions.

| Field | Function |
|---|---|
| Command List | Lists all of the commands that can be run through this dialog. |
| Description | Explains the purpose of the selected command. Also displays the syntax for the command arguments. |
| Arguments | Lists and defines the arguments and parameters available for the selected command. |
| Command | Displays the selected command and any selected arguments. |

## Procedure

Use the following procedure to run a command from the Run Program tool.

| Step | Action |
|---|---|
| 1 | Select **Run Program** from the Tools menu on the Gentran:Server window.<br>**System Response**<br>Gentran:Server displays the **Run a Command** dialog box. |
| 2 | Double-click the Command List option you want to run.<br>**System Response**<br>The command name displays in the Command field, and the Description and Arguments boxes display information for the selected command. |
| 3 | Double-click any arguments you want from the displayed list.<br>**Note**<br>If you select a variable, such as ‘record number’ for the **ediscope** command, the Input Command Argument dialog box displays, allowing you to enter a value. |
| 4 | Click **Run**.<br>**System Response**<br>Gentran:Server runs the command. |

# Alphabetical Listing of Commands

## apptptblcreat

### Purpose

Creates an empty Application/Trading Partnership Rules (**apptptbl)** Table. This table stores rules that Gentran:Server uses to identify a trading partner in an application document when you use the multi-field Trading Partnership lookup function.

**CAUTION**

**Use with caution! This program overwrites any data in the existing apptptbl table.**

### Files created

This table lists the names and descriptions of the files created when you run **apptptblcreat**.

| File name | Description |
|---|---|
| apptptbl.dat | The table that stores the application Trading Partnership rules. |
| apptptbl.idx | The index file that relates to the *apptptbl.dat* file. |
| apptptblcreat.log | The log file that reports the status of this operation. |
| apptptbl.lck | The lock file generated when the command is run from the DOS prompt. |

### Starting the program

Start **apptptblcreat** from the UNIX host command line in all Gentran:Server for UNIX product levels.

Start **apptptblcreat** from the DOS command line in the Workstation product.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
apptptblcreat -cp <config path>
```

## Parameters and variables

This table lists the parameters and variables defined for the **apptptblcreat** command.

| Parameter | Definition |
|---|---|
| -cp < config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# apptptblmv

## Purpose

Loads and unloads Application/Trading Partnership Rules records in the **apptptbl** file.

## Files created

This table lists the name and description of the file created when you run **apptptblmv**.

| File name | Description |
|---|---|
| apptptbl.unl | The file created when the application Trading Partnership rules records are unloaded into ASCII format for moving from platform to platform. |
| apptptbl.log | The file that reports the status of the move operation.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation<br>◗ The *temp* directory for Gentran:Server for UNIX |
| apptptbl.lck | The lock file generated when the command is run from the DOS prompt. |

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
apptptblmv -[l/lo/u] -[cp <config path>]
```

**Note**
Do not include the brackets or slashes in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **apptptblmv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file. |
| lo | Load from ASCII file and update existing ISAM file (overwrite). |
| u | Unload from ISAM to ASCII file. |

# appxrefcreat

## Purpose

Creates an empty Application/Trading Partnership Code Cross Reference (**appxref)** Table.

**CAUTION**

**Use with caution! This program overwrites any data in the existing appxref table.**

## Files created

This table lists the names and descriptions of the files created when you run **appxrefcreat**.

| File name | Description |
|---|---|
| appxref.dat | The table that stores the application file cross-reference string as defined by the application Trading Partnership rules. |
| appxref.idx | The index file that relates to the *appxref.dat* file. |
| appxref.lck | The lock file generated when the command is run from the DOS prompt. |
| appxref.log | The log file that reports the status of this operation. |

## Starting the program

Start **appxrefcreat** from the UNIX host command line for all Gentran:Server for UNIX product levels.

Start **appxrefcreat** from the DOS command line for the Workstation product.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
apptptblmv -[l/lo/u] -[cp <config path>]
appxrefcreat -cp <config path>
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **appxrefcreat** command.

| Parameter | Definition |
|---|---|
| -cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# appxrefmv

## Purpose

Loads and unloads records in the Application/Trading Partnership Code Cross Reference (**appxref)** ISAM file.

## Files created

This table lists the name and description of the file created when you run **appxrefmv**.

| File name | Description |
|---|---|
| appxref.unl | The file created when the Application/Trading Partnership Cross Reference records (*appxref.dat* and *appxref.idx*) are unloaded into ASCII format for moving from platform to platform. |
| appxrefmv.log | The log file created by **appxrefmv**.<br>This file is created in:<br>◗ The host *temp* directory for Gentran:Server for UNIX<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
appxrefmv -[l/lo/u] [-cp <config path>]
```

**Note**
Do not include the braces or slashes in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **appxrefmv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file. |
| lo | Load from ASCII to ISAM file, while updating records (overwrite). |
| u | Unload from ISAM to ASCII file. |

# atext

## Purpose

Manipulates characters in a file.

## When to use

You can use **atext** to add, delete, or replace characters within a file. You can also use it to split one file into multiple files.

## Starting the program

Start **atext** from the Run Program option on the Main window Tools menu.

### Reference

See the Using the Run Program Tool topic in this chapter.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
atext <infile> <outfile> [-cp <config path>] -{a/b/B/d/i/f/p/r/s/S/
T/x} [c1] [c2] [c] [N] [N1] [N2]
```

### Note

Do not include the brackets, slashes, or braces in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **atext** command. The parameter you choose determines which of the values [c], [c1], [c2], [N], [N1], and [N2] you must enter.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| a <c1> <c2> | Insert character c2 after each occurrence of c1. |
| B <N> | Insert a line break (new line) every N bytes and pad the last record to N bytes. |
| b <c1> <c2> | Insert character c2 before each character c1. |
| d <N1> <N2> | Remove characters from position N1 to position N2 in each record. |

(Continued on next page)

| (Contd)<br>Parameter | Definition |
|---|---|
| i <c> <N> | Insert character c every N bytes in the file. |
| f <c> <N> | Pad records to N bytes with character c. |
| p <c> <N> | Insert character c at position N in every record. |
| r <c1> <c2> | Replace each occurrence of c1 with c2. |
| S <N> | Split the file by units of N times 1000 bytes into a new file. Output files are <outfile>.000, <outfile>.001, and so on. |
| s <N> | Strip a character every N bytes in the file. |
| T <c1> <c2> <N> | Replace each occurrence of c1 with <N> occurrences of character c2. |
| x <c> | Delete all occurrences of character c. |

**Rule 1**

When c1, c2, and c require a character, enter the hexadecimal value that corresponds to the ASCII character that you want to specify.

**Example**

The hexadecimal value for a space is “20” and the value for a null is “00”. Therefore, to substitute a space for each null character, you can enter the following command:

```
atext /usr/gentran/data/cnet /usr/gentran/data/x_12 -r 00 20
```

(Continued on next page)

## Rule 2

If you need to make multiple changes to a file, you need to run **atext** multiple times.

### Example

You want to edit a Trading Partner file. Trading Partner files are null-delimited, and therefore are displayed as only one line when you view them in an editor. You want to split the Trading Partner file into separate records to view it easily. First you use the replace option (-r) to substitute a visible character such as a tilde (~) for the null (you don't want to accidentally remove a null character while editing the file). Then you can use the atext block option (-b) to break the file into blocks exactly one Trading Partner Record in length (1012 bytes). Now the file is easily viewed and edited.

After you edit and save the file, you need to undo the actions you previously performed with **atext**. First you remove the newline character added by the block option, this time using the delete option (-x). Then you replace the visible character with null.

## Example commands

```
atext /usr/gentran/tp/tp.dat /usr/gentran/tp2.dat -r 00 7E
atext /usr/gentran/tp/tp2.dat /usr/gentran/tp3.dat -b 1012
```

(You open and edit the tp3.dat file, saving it as /usr/gentran/tp4.dat.)

```
atext /usr/gentran/tp/tp4.dat /usr/gentran/tp5.dat -x 0A
atext /usr/gentran/tp/tp5.dat /usr/gentran/tp.dat -r 00 7E
```

# cancel_mbid

## Purpose

The **cancel_mbid** command cancels outbound queued items for a specific mailbag ID. Removes items from the */togo* directories.

## Starting the program

Start **cancel_mbid** from the UNIX host command line.

**Note**
The **cancel_mbid** command is available only for the Gentran:Server Advanced Data Distribution product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
cancel_mbid {-r[d/dm]} {<mbagid>}
```

**Note**
Do not include the braces, slashes, or brackets in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **cancel_mbid** command.

| Parameter | Definition |
|---|---|
| d | Dequeue all queued items. |
| m <mbagid> | Mark all to be dequeued by **srv_deq**. This option requires the -d option.<br>**Reference**<br>See the Gentran:Server *for UNIX Advanced Data Distribution Guide* for information about marking. |
| r | Generate a report listing to the screen (stdout). |

# catcreat

## Purpose

Creates empty category value and category type records.

**CAUTION**

**Use with caution! This program overwrites any data in existing category value and category type files.**

## Files created

This table lists the name and description of the files created when you run **catcreat**.

| File name | Description |
|---|---|
| catcreat.log | The error message/log file created to report the status of the generation of the category files.<br>This file is created in:<br>◗ The host *temp* directory for Gentran:Server for UNIX<br>◗ The *rpt* directory for Workstation. |
| catvalue.dat | The Category Value file, which stores the category records.<br>This file is created in the trading partner directory. |
| catvalue.idx | The index file that relates to the *catvalue.dat* file.<br>This file is created in the trading partner directory. |
| catvalue.lck | The lock file generated when the command is run from the DOS prompt. |
| cattype.dat | The Category Type file. This file contains the category labels displayed in the Trading Partner explorer.<br>This file is created in the trading partner directory. |

(Continued on next page)

## Starting the program

Start this program from:

- The **Run Program** option on the Main window **Tools** menu.
- The **New Category File** option on the **Tools** menu within the Trading partner explorer.

### References

See the Using the Run Program Tool in this chapter.

See the Working with Trading Partnerships chapter in the Gentran:Server *for UNIX and Workstation Application Integration Guide*.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
catcreat [-cp <config path>]
```

### Note

Do not include the brackets in the command. Do include slashes as needed within the path.

### Example

```
catcreat -cp /usr/mentorcs
```

## Parameter and variable

This table lists the parameter and variable defined for the **catcreat** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# catmv

## Purpose

Loads and unloads Trading Partnership category value records and category labels.

## Files created

This table lists the name and description of the files created when you run **catmv**.

| File name | Description |
|---|---|
| catvalue.unl | The file created when the *catvalue.dat* and *catvalue.idx* files are unloaded into ASCII format.<br>This file is created in the trading partner directory. |
| catmv.log | The error/log file created by *catmv*.<br>This file is created in:<br>◗ The host *temp* directory for Gentran:Server for UNIX<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start **catmv** from:

- The **Run Program** option on the Main window **Tools** menu.
- The **Load/Unload**=>**Category Records** option on the Tools menu within the Trading Partner explorer.

### References

See the Using the Run Program Tool.

See the Working with Trading Partnerships chapter in the Gentran:Server *for UNIX and Workstation Application Integration Guide*.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
catmv -[l/lo/u] [-cp <config path>]
```

### Note

Do not include the brackets or slashes in the command. Do include any slashes needed within the path.

(Continued on next page)

## Example

```
catmv -u -cp /usr/mentorcs
```

## Parameters and variables

This table lists the parameters and variables defined for the **catmv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*)<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file |
| lo | Load from ASCII file to ISAM file, and update existing records (overwrite) |
| u | Unload from ISAM to ASCII file |

# chck_cfg

## Purpose

Displays the directories specified in the Gentran:Server Configuration File. Used when you cannot access the Setup Directories dialog box to view the directories that your system uses.

## Files created

This table lists the name and description of the files created when you run **chck_cfg**.

| File name | Description |
|---|---|
| chck_cfg.log | The log file that contains the results of running the program. This file is created in:<br>◗ The *temp* directory for Gentran:Server for UNIX<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start **chck_cfg** from:

- The UNIX host if you are running Gentran:Server for UNIX.
- A DOS command prompt if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
chck_cfg {-cp <config path>}
```

**Note**
Do not include the brackets in the command.

## Parameter and variable

This table lists the parameter and variable defined for the **chck_cfg** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |

# chckodbc

## Purpose

The **chckodbc** program validates Gentran:Server environments that contain the optional ODBC database drivers.

## When to use

After you install and set up ODBC drivers, use **chckodbc** to check environment variables on a UNIX host, ensure that the ODBC components are in the correct locations, and to test the connection between Gentran:Server and the database.

## Starting the program

Start **chckodbc** from:

- The UNIX host if you are running Gentran:Server for UNIX.
- A DOS command prompt if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
chckodbc {-i/ -c <dsn>/ -e} [-cp <config path>]
```

**Note**
Do not include the braces or brackets in the command.

## Locating the log file

The status of this operation is stored in the file *chck_cbj.log*. This file is located in the DOS directory *rpt* and the UNIX directory *temp*.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **chckodbc** command.

| Parameter | Definition |
|---|---|
| c <dsn> | Test the connection to the specified data source name.<br>This tests whether the machine you are on can connect to the database, and verifies that Gentran:Server can find other shared libraries need to connect. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| e | For Gentran:Server for UNIX, verify that the ODBC environment variables are set correctly.<br>**Note**<br>This parameter is not available on Gentran:Server Workstation. |
| i | Check that ODBC components for Gentran:Server version 6.1 are installed in the correct locations. |

# chck_tbl

## Purpose

Displays the informational record for a map.

## Files created

This table lists the name and description of the log file created when you run **chck_tbl**.

| File name | Description |
|---|---|
| chck_tbl.log | The log file that contains the results of running the program.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation.<br>◗ The *temp* directory for Gentran:Server for UNIX. |

## Starting the program

Start **chck_tbl** from the Run Program option on the Main window Tools menu.

**Reference**
See the Using the Run Program Tool.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
chck_tbl <mapname>.TPL | .TBL {-cp <config path>}
```

**Note**
Do not include the brackets in the command.

## Parameter and variable

This table lists the parameters and variables defined for the **chck_tbl** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| <mapname>.TBL | The map for which you want to view information. |

# cl_arch

## Purpose

Removes data manager archive data from the archives.

## Files created

This table lists the name and description of the file that **cl_arch** can create when running.

| File name | Description |
|---|---|
| cl_arch.l | The log file containing information about purged data.<br>This file is located in the $EDI_ROOT directory of the installation. |

## Starting the program

Start **cl_arch** from the UNIX host command line.

**Note**
The **cl_arch** command is available only for the Gentran:Server EC Workbench product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
cl_arch {-a<days>/-B<begin_date> -E<end_date> -N -D<agent_name>}
```

**Note**
Do not include the braces, slashes, or brackets in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **cl_arch** command.

| Parameter | Definition |
|---|---|
| a | Purge all archived data older than the number of days set in the data manager’s ARCHIVE_PERIOD initialization parameter.<br>If ARCHIVE_PERIOD is not set, **cl_arch** uses the number of days set in the EDI_ARCHIST environment variable.<br>If neither ARCHIVE_PERIOD or EDI_ARCHIST is set, then **cl_arch** uses 90 days. |
| a<days> | Purge all archived data older than a specific number of days. |
| B<begin_date> | Indicates a start date when purging a range of archived data for a specific date range. Use the format CCYYMMDD. |
| D<agent_name> | Indicates that archived data is to be purged for only the named intelligent agent (data manager). |
| E<end_date> | Indicates an end date when purging a range of archived data for a specific date range. Use the format CCYYMMDD. |
| N | Do not delete transaction register records. |

## Example 1

In this example, **cl_arch** deletes archive entries older than 60 days.

```
cl_arch -a60
```

## Example 2

In this example, **cl_arch** uses the value set in the ARCHIVE_PERIOD parameter of the data manager’s initialization file.

```
cl_arch -a
```

(Continued on next page)

## Example 3

In this example, **cl_arch** deletes archive entries with dates of January 15, 1998 through February 15, 1998.

```
cl_arch -B19980115 -E19980216
```

## Example 4

In this example, cl_arch deletes archive entries older than 120 days for the inxt data manager.

```
cl_arch -a120 -Dinxt
```

# cleanlog

## Purpose

Deletes log or script journal entries that are older than a set date and time and contained in the current directory.

**Note**
Log files have a *.l*, *.old*, or *.log* extension.

## Files created

This table lists the name and description of the files created when you run **cleanlog**.

| File name | Description |
|---|---|
| cleanlog.l | The log file containing information regarding the number of records purged and the number saved. |

## Starting the program

Start **cleanlog** from the UNIX host command line.

**Note**
The **cleanlog** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
cleanlog {<logname>} {-a<# of days>/-B<delete_from_date> -
E<delete_to_date>}
```

**Note**
Do not include the braces, slashes, or brackets in the command.

(Continued on next page)

## Automatically purging journals

To automate the purging of entries from journals, you can run the script **cleanlog.sh** from the Permanent Schedule.

**Reference**
For information about creating scripts, see the Working with Scripts chapter in the Gentran:Server *for UNIX with EC Workbench Data Flow Administration Guide*.

For information about adding a script to the Permanent Schedule, see the Running Scripts chapter in the Gentran:Server *for UNIX with EC Workbench Data Flow Administration Guide*.

## Parameters and variables

This table lists the parameters and variables defined for the **cleanlog** command.

| Parameter | Definition |
|---|---|
| a <# of days> | Delete log entries older than a specified number of days in a single log. The default is 90 days. |
| B <delete from date> | Identifies a start date when deleting log entries in a specific log for a specific date range. Specify the date in the format MMDDYY or MMDDCCYY. |
| E <delete to date> | Identifies an end date when deleting log entries in a specific log for a specific date range. Specify the date in the format MMDDYY or MMDDCCYY. |
| <logname> | The name of the log file from which you are deleting entries. |

# clean_trn

## Purpose

Marks Transaction Register entries for deletion if they are older than a set date and time. Once the entries are marked, you need to run **isops** to actually remove the entries from the *trn.idx* and *trn.dat* files.

## Starting the program

Start **clean_trn** from the UNIX host command line.

**Note**
The **clean_trn** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
clean_trn -d {<CCYYMMDD[HHMMSS]>}
```

**Note**
Do not include the braces or brackets in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **clean_trn** command.

| Parameter | Definition |
|---|---|
| d | Indicates the deletion of Transaction Register entries older than a set date and time. |
| <CCYYMMDD[HHMMSS]> | A specific cutoff date and time. |

# cocreate

## Purpose

Creates an empty Trading Partnership Contact File (*contact.dat, contact.idx*).

**CAUTION**

**Use with caution! This program overwrites any existing data in the contact files.**

## Files created

The table below lists the names and descriptions of the files created when you run **cocreate**.

| File name | Description |
|---|---|
| contact.dat | The Contact File, which stores the Contact records maintained by the Trading Partnership explorer.<br>This file is created in the trading partner directory. |
| contact.idx | The index file that relates to the *contact.dat* file.<br>This file is created in the trading partner directory. |
| cocreate.log | The error file created to report the status of the generation of the Contact Files (*contact.dat* and *contact.idx*).<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation<br>◗ The *temp* directory for Gentran:Server for UNIX |

## Starting the program

Start **cocreate** from

- The **New/Create** => **Contact File** from the **Tools** menu within the Trading Partnership explorer.
- The DOS command line, if you are running Gentran:Server Workstation.
- The UNIX host command line if you are running Gentran:Server for UNIX.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
cocreate [-cp <config path>]
```

## Parameters and variables

This table lists the parameters and variables defined for the **cocreate** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# conmv

## Purpose

Loads and unloads Trading Partnership contact records.

## Files created

This table lists the name and description of the file created when you run **conmv**.

| File name | Description |
|---|---|
| contact.unl | The file created when the Contact records (*contact.dat* and *contact.idx*) are unloaded into ASCII format for moving from platform to platform. This file is created in the trading partner directory. |
| conmv.log | The error/log file created by **conmv**.<br>This file is created in:<br>◗ The host *temp* directory for Gentran:Server for UNIX<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start **conmv** from:

- The **Run Program** option on the Main window Tools menu.
- The **Unload**=>**Contact Records** or **Load**=>**Contact Records** option from the **Tools** menu within the Trading Partnership explorer.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
conmv -[l/lo/u] [-cp <config path>]
```

**Note**
Do not include the braces or slashes in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **conmv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file. |
| lo | Load from ASCII to ISAM file, while updating records (overwrite). |
| u | Unload from ISAM to ASCII file. |

# continfo

## Purpose

The **continfo** command retrieves contact information for a given contact code and sends it to standard out, which is typically your display terminal.

## Starting the program

Start **continfo** from:

- The UNIX host command line for Gentran:Server for UNIX.
- The DOS command line for Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
continfo <contact_code> -l<level> -<output code> {-<output code> …}
[-cp <config path>]
```

## Parameters and variables

This table lists the parameters and variables defined for the **continfo** command.

| Parameter | Definition |
|---|---|
| <contact_code> | Specifies the contact code to search for within the Trading Partnership Contact Record. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| lx | Specifies the level of contact information **continfo** displays.<br>These are the valid level values:<br>1=Interchange contacts<br>2=Group contacts<br>3=TP contacts |
| <output code> | Specifies which contact information to include in the output. You can specify multiple output codes. |

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **continfo** command.

| Contact level | Definition |
|---|---|
| 1 | Interchange. This is the default. |
| 2 | Group |
| 3 | Trading Partnership |

## Output codes

This table list the output codes and the corresponding contact information that each code produces in the output.

| Output Code | Contact Information |
|---|---|
| a1 | Address line 1 |
| a2 | Address line 2 |
| c | City |
| c1 | Comment line |
| co | Country |
| e1 | Extension 1 (for phone 1) |
| e2 | Extension 2 (for phone 2) |
| f | Fax number |
| in | Internet address |
| n | Contact name |
| p1 | Phone 1 |
| p2 | Phone 2 |
| s | State |
| z | Zip code |

# create_envprim_cfg

## Purpose

Creates the *envprim.cfg* file. This file holds the names of all Gentran:Server directories. The program extracts the directory names from the EDI_ROOT directory.

**CAUTION**

**Use with caution! This program overwrites any data in an existing *envprim.cfg* file. You normally set values for this file through the Environment Administration functions in the Security Administration utility.**

## Files created

This table lists the name and description of the file created when you run **create_envprim_cfg**.

| File name | Description |
|---|---|
| envprim.cfg | The user configuration file.<br>This file is created in the EDI_ROOT directory. |

## Starting the program

Start **create_envprim_cfg** from the UNIX host command line.

**Note**
The **create_envprim_cfg** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
create_envprim_cfg
```

# datecalc

## Purpose

Calculates a date using the current date as the reference point.

## Starting the program

Start **datecalc** from the UNIX host command line.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
datecalc{+/-<days>} -[c]
```

**Note**
Do not include the braces or slashes in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **datecalc** command.

| Parameter | Definition |
|---|---|
| +/-<days> | Calculates a date that is the specified number of days forward(+) or backward(-) from the current date. |
| c | Includes the century in the result. (CCYYMMDD) |

# dcheck

## Purpose

Checks or repairs ISAM files.

## Starting the program

Start **dcheck** from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
dcheck -{B/b/f/h/i} <isamfile> [...]
```

**Note**
Do not include the braces or slashes in the command, except the slash when entering the path to the ISAM file.

## Parameters and variables

This table lists the parameters and variables defined for the **dcheck** command.

| Parameter | Definition |
|---|---|
| B | Rebuild a specific index. |
| b | Build new index from data. |
| f | Fix corrupt indexes |
| h | Display header only. |
| i | Check index only. Ignore data file. Requires the path to the ISAM file you are checking or repairing. |

# deltrn

## Purpose

Marks unwanted records for deletion from the Transaction Register. Once the entries are marked, you must run **isops** to actually remove the entries from the *trn.idx* and *trn.dat* files.

## Starting the program

Start **deltrn** from the UNIX host command line.

**Note**
The **deltrn** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
deltrn {-t<tp code>} {-d<doc_ref_num>} [-x<direction>]
[-l<trans_reg>]
```

**Note**
Do not include the braces, brackets, or slashes in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **deltrn** command.

| Parameter | Definition |
|---|---|
| a<audit_file> | Delete records from the Transaction Register based on translation errors stored in a specific audit file. Requires an audit file, either *edistat.i* or *edistat.o* that the translator produced. |
| d<doc_ref_num> | Indicates a document reference number. |
| l<trans_reg> | Indicates a specific Transaction Record directory. Requires the relative path from EDI_ROOT to the transaction register. Omit this argument if the transaction register is in EDI_ROOT. |
| t<tp code> | Indicates a Trading Partner code.<br>(Continued on next page) |

| (Contd)<br>Parameter | Definition |
|---|---|
| u<ID_num> | Delete all records up to a specific and unique ID number. Specify the ID that Gentran:Server appends to a destination file name. |
| x<direction> | Indicates whether to delete inbound or outbound records. |

## Example 1

Use the following command to delete the transaction register records based on translation errors noted in a specific audit file:

```
deltrn {-a<audit_file>}
```

## Example 2

Use the following command to delete records less than a specific ID:

```
deltrn {-u<ID_num>}
```

# dumpXMLReport

## Purpose

Reformats the data in the dtlLog.err file and presents it in an easy-to-read format.

## Starting the program

Start **dumpXMLreport** from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
dumpXMLreport <input file> <output file>
```

**Note**
Do not include the braces or slashes in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **dumpXMLreport** command.

| Parameter | Definition |
|---|---|
| <input file> | The path and name of the input file.<br>**Example**<br>rpt/dtlLog.err |
| <output file> | The path and name for the output file. |

# ediarc

## Purpose

The **ediarc** program archives translated EDI data to the Audit/Archive files and performs functional acknowledgment reconciliation as needed.

## Files created

This table lists the names and descriptions of the files created when you run **ediarc**.

| File name | Description |
|---|---|
| <uniqueid>.arc | The archived files that contain a copy of the EDI formatted data. These files are created in:<br>◗ The EDI History Audit directory for Gentran:Server Workstation (The default name for this directory is hisaud).<br>◗ The EDI History Audit directory for Gentran:Server for UNIX (The default name for this directory is edihist). |
| edistat.irr | An error file created to store the inbound Audit records of the data that could not be archived. This file is created in:<br>◗ The EDI Status Audit directory for Gentran:Server Workstation (The default name for this directory is stataud).<br>◗ The edistat directory for Gentran:Server for UNIX.<br>**Note**<br>The files *edistat.irr* are created only if errors are detected. |
| edistat.orr | An error file created to store the outbound Audit records of the data that could not be archived. This file is created in:<br>◗ The EDI Status Audit directory for Gentran:Server Workstation (The default name for this directory is stataud).<br>◗ The ediastat directory for Gentran:Server for UNIX.<br>**Note**<br>The files *edistat.orr* are created only if errors are detected. |

(Continued on next page)

| (Contd)<br>File name | Description |
|---|---|
| xlcntl.err | An error file created to report the status of **ediarc**. This file is created in:<br>◗ The Report/Log directory for Gentran:Server Workstation (The default name for this directory is Rpt).<br>◗ The Report/Log directory for Gentran:Server for UNIX on the client (The default directory is Rpt).<br>◗ The Temp directory for Gentran:Server for UNIX on the host (The default name for this directory is Temp). |
| edihist.dat/<br>edihist.idx | The archived files that point to the <uniqueid>.arc file. This file is created in:<br>◗ The EDI History Audit directory for Workstation (The default name for this directory is hisaud).<br>◗ The EDI History Audit directory for UNIX (The default name for this directory is edihist). |
| fa.stat | A SAP status message file created when processing functional acknowledgments. This file is created in:<br>◗ The $EDI_Root Temp directory for Gentran:Server for UNIX. |

(Continued on next page)

## Starting the program

Start **ediarc** from the **Archive** option on the Translate Documents dialog box.

## Using the command in a script or batch file

Use the following notation when including this command in a script, batch file, or command line:

```
ediarc [-cp <config path>] [-d offset] [-e] [-j] -[i/o] [-s]
```

**Note**
Do not include the slashes, brackets, or braces in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **ediarc** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The full path to the user configuration file (*envprim.cfg*). |
| d | Obtain the SAP IDOC number from the Document Reference Number. Offset this value from the start of the Document Reference Number (0<offset<23)<br>**Note**<br>This parameter is available only with the SAP extension. |
| e | Treat messages with numbers less than 200 as errors. Issue return code five instead of zero for those messages. |
| j | Process all sets in an inbound functional acknowledgement that do not have a related AK2. Do not process any set for which there is an AK2 that specifically indicates the set is in error. |
| i | Process inbound translation records. |
| o | Process outbound translation records. |
| s | Generates a SAP status message file when processing SAP functional acknowledgments. |

# edicreat

## Purpose

Creates empty *edihist.dat* and *edihist.idx* files.

**CAUTION**

**Use with caution! This program overwrites your existing permanent audit files, *edihist.dat* and *edihist.idx*.**

## Files created

This table lists the names and descriptions of the files created when you run **edicreat.**

| File name | Description |
|---|---|
| edihist.dat | The permanent Audit File, which stores the Audit records that are maintained by the **ediarc** program.<br>This file is created in:<br>◗ The *hisaud* directory for Gentran:Server Workstation.<br>◗ The *edihist* directory for Gentran:Server for UNIX. |
| edihist.idx | The index file that relates to the *edihist.dat* file.<br>This file is created in:<br>◗ The *hisaud* directory for Gentran:Server Workstation.<br>◗ The *edihist* directory for Gentran:Server for UNIX. |
| edicreat.log | The file in which the program records errors.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation.<br>◗ The *temp* directory for Gentran:Server for UNIX. |

## Starting the program

Start **edicreat** from:

- The **New/Create Archive Files** option on the Tools menu on the Main window.
- The **New/Create Archive Files** option on the Tools menu within the Audit/ Archive utility.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
edicreat [-cp <config path>]
```

**Note**

Do not include the brackets in the command.

## Parameters

This table lists the parameters defined for the **edicreat** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |

# edifrmat Overview

## Introduction

The next two sections explain how to use the **edifrmat** command to format incoming EDI data for the translator. The first section gives the command syntax and lists the parameters for the command. The second section explains how to force **edifrmat** to write new files instead of appending to or overwriting existing files.

## Files created

This table lists the names and descriptions of the files created when you run **edifrmat**.

| File name | Description |
|---|---|
| edifrmat.not | A temporary file created to store non-EDI data.<br>This file is created in the *temp* directory. |
| <filename>.$$$ | A temporary file created to store binary data.<br>This file is created in:<br>◗ The directory you specified for temporary files in your *.profile* (which is probably not the same as the Server temporary directory) for Gentran:Server for UNIX.<br>◗ The directory from which you run **edifrmat** for Gentran:Server Workstation. |
| output/<filename>.frm | The output file generated as a result of running **edifrmat**.<br>These files are created in the *temp* directory. |
| edifrmat.log | A file created to report the status of **edifrmat**. This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation.<br>◗ The *temp* directory for Gentran:Server for UNIX. |

# edifrmat Syntax

## Purpose

Formats incoming EDI data for the translator. This includes stripping non-EDI characters and changing the segment terminator to a new line.

## Starting the program

Start **edifrmat** from:

- The **Format Input Document** option on the Translate Documents dialog box.
- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
edifrmat {<infile>} {<outfile>} [-cp <config path>] -[a/b/c/d/e/i/
l/s/v]
```

**Note**
Do not include the braces or brackets in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **edifrmat** command.

| Parameter | Definition |
|---|---|
| <infile> | The input file. |
| <outfile> | The output file for EDI data. |
| a | Append to output file(s), *edifrmat.log*, and *edifrmat.not*. |
| b | Consider lowercase letters or invalid UNOA characters in input file to be an error and write entire interchange containing lowercase letters or invalid UNOA characters to *edifrmat.not.* |
| c | Accept empty segments as valid data. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| d | Write debugging information to the screen. |
| e | Leave extended characters. |

(Continued on next page)

| (Contd)<br>Parameter | Definition |
|---|---|
| i | Insert a new line character between each interchange (usually used only on foreign networks). If using the i parameter, run **edifrmat** before putting network envelopes around your data. |
| l | Strip all new line characters from input file. |
| s | Maintain the existing record separator instead of using newline substitution. Used for routing EDIFACT/GENCOD/VDA data. |
| v | Verify set and group counts. |

# Controlling how edifrmat Writes Files

## Introduction

To change how **edifrmat** writes files, you can set environment variables to specify a path and/or file name different than the defaults described.

## Default operation

In normal operation, each time it runs **edifrmat** overwrites the *edifrmat.not* file in the *temp* directory, the *edifrmat.log* file in the *temp* directory (*rpt* directory in the Workstation version), and the output file. If **edifrmat** is run with the -a parameter, it appends the *edifrmat.not, edifrmat.log,* and output files.

## The environment variables

You can force **edifrmat** to write new *.not* or *.log* files, or both, each time it executes by setting selected environment variables immediately before each time you run **edifrmat**. The **edifrmat** program works with the environment variables listed in this table.

| Environment Variable | Function |
|---|---|
| EDIFRMATLOG | Creates a new *.log* file with the specified name. |
| EDIFRMATNOT | Creates a new *.not* file with the specified name. |

## Setting the environment variables

Use this table to find the syntax of the environment commands you need to use.

| To run edifrmat from a… | Then use the syntax… |
|---|---|
| C shell script | setenv EDIFRMATLOG ./<directory>/<filename><br>OR<br>setenv EDIFRMATNOT ./<directory>/<filename> |

(Continued on next page)

| To run edifrmat from a… | Then use the syntax… |
|---|---|
| DOS batch file | set EDIFRMATLOG=.\<directory>\<filename><br>OR<br>set EDIFRMATNOT=.\<directory>\<filename><br>before the **edifrmat** command. |
| Korn or Bourne shell script | export EDIFRMATLOG=./<directory>/<filename><br>OR<br>export EDIFRMATNOT=./<directory>/<filename> |

## Rules for usage

Follow these rules when using the EDIFRMATNOT or EDIFRMATLOG environment variables.

- Place the environment command(s) before the **edifrmat** command.
- Whenever you want to run **edifrmat** repeatedly within the same DOS batch file or UNIX script, you must reset the environment variable before each **edifrmat** command.
- After you run a DOS batch file or UNIX script you must reset the environment variable to the default.

# edimv

## Purpose

Loads and unloads archive data records.

## Files created

This table lists the name and description of the file created when you run **edimv**.

| File name | Description |
|---|---|
| edihist.unl | The file created when the Audit records are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the *temp* directory. |
| edimv.log | The log/error file created by **edimv**.<br>This file is created in the *temp* directory. |

## Starting the program

Start **edimv** from:

- The **Load Archive Files** option from the Archive menu in the Main screen.
- The **Unload Archive Files** option from the Archive menu in the Main screen.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
edimv -[l/p/u] [-cp <config path>]
```

**Note**
Do not include the brackets or slashes in the command.

(Continued on next page)

## Parameters and variables

This table defines the parameters and variables defined for the **edimv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file. |
| p | Unload from ISAM to flat file and log archive files for purging. |
| u | Unload from ISAM to ASCII file. |

# edipurge

## Purpose

Deletes translation archive records from the *edihist.dat* and *edihist.idx* files. Also deletes associated data archive files.

## Files created

This table lists the name and description of the file created when you run **edipurge**.

| File name | Description |
|---|---|
| edipurge.log | The log/error file created by **edipurge**.<br>This file is created in:<br>◗ The *temp* directory for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start **Purge Archive=>edipurge** from the Archive menu in the Main window.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
edipurge {[-O] [-I]} [-T <tpcode>] [-IC <intchg>] [-G <grp>]
[-S <set_id>] [-B <beg_dt>] [-E <end_dt>] {-F -[ADELNPRVYZ]}
[-cp <config path>]
```

**Note**
Do not include the brackets, slashes, or braces in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **edipurge** command.

| Parameter | Definition |
|---|---|
| B <beg_dt> | Indicates a translation date at which you want to start deleting. Leave blank to delete everything less than the ending date (CCYYMMDD). After the date you may also enter + or - and an integer to indicate a number of days following or preceding the specified date.<br>**Example**<br>19990303-5 |
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| E <end_dt> | Indicates a translation date at which you want to end deleting. Leave blank to delete everything after the begin date (CCYYMMDD). After the date you may also enter + or - and an integer to indicate a number of days following or preceding the specified date.<br>**Example**<br>19990303-5 |
| F | Indicates one or more FA status codes to delete. You can specify multiple statuses in one command. |
| G <grp> | Indicates a specific group control number to delete from the audit/archive file. |
| I | Purge inbound records. To purge both inbound and outbound records with one command, specify both the I and the O parameters. |
| IC <intchg> | Indicates a specific interchange control number to delete from the audit/archive file. |
| O | Purge outbound records. To purge both inbound and outbound records with one command, specify both the I and the O parameters. |
| S <set_id> | Indicates a specific document identification code (Set ID). |
| T <tpcode> | Indicates a specific Trading Partnership Code (required with option G). |

(Continued on next page)

## Status codes

This table lists the audit record status codes defined for the **edipurge** command.

| Status Code | Definition |
|---|---|
| A | Acknowledgment arrived okay. |
| E | Acknowledgment arrived with errors and was accepted. |
| L | Acknowledgment is currently late. |
| N | Acknowledgment not expected. |
| P | Acknowledgment arrived with errors and was partially accepted. |
| R | Acknowledgment arrived with errors and was rejected. |
| V | Verbal acknowledgment arrived okay. |
| Y | Acknowledgment expected, but has not yet arrived. |
| Z | Verbal acknowledgment arrived with errors. |

## Example

This example purges all inbound and outbound acknowledgments that arrived with errors and were rejected.

```
edipurge -O -I -F -R
```

# edirsnd

## Purpose

To resend a document that your trading partner did not receive or that contained incorrect information (for example, compliance errors).

The **edirsnd** command gives you the option of resetting the interchange, group, and set control numbers in an EDI document; creates a new Audit record; and updates the Trading Partnership records. You should use this command only when resending outbound data to a Trading Partner.

## Files created

This table lists the name and description of the file created when you run **edirsnd**.

| File name | Description |
|---|---|
| edirsnd.log | The log/error file created by **edirsnd**.<br>This file is created in:<br>◗ The *temp* directory for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

**edirsnd** is run automatically when you save an archive document after you edit it to resend.

**Reference**
See the How to Extract Archived EDI Documents and the How to Prepare Documents to Resend topics in the Gentran:Server *for UNIX and Workstation Application Integration Guide*.

You may also start **edirsnd** from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
edirsnd <infile> <tpcode> [-o <outfile>] [-I +|<set>] [-G +|<grp>]
[-S +|<set>] [-cp <config path>]
```

**Note**
Do not include the brackets in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **edirsnd** command.

| Parameter | Definition |
|---|---|
| <outfile> | The output file name. |
| <tpcode> | The Trading Partnership Code. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| G <grp> | Resets the group control number to the specified number. Instead of a number, you can use a plus sign (+) to increment the Trading Partnership record group control number counter. |
| infile | The input file name. |
| I <icn> | Resets the interchange control number. Instead of a number, you can use a plus sign (+) to increment the Trading Partnership record interchange control number counter. |
| S <set> | Resets the set control number to the specified value. Instead of a number, you can use a plus sign (+) to increment the Trading Partnership record set control number counter. |

# ediscope

## Purpose

Displays the contents of *edistat.i* or *edistat.o*, displaying each record field by field.

**Note**
This is a diagnostics program. Do not run this program unless a Sterling Commerce customer support representative asks you to run it.

## Files created

This table lists the names and descriptions of the files created when you run **ediscope**.

| File name | Description |
|---|---|
| ediscope.log | The log file generated as a result of running **ediscope**.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation.<br>◗ The *temp* directory for Gentran:Server for UNIX. |

## Starting the program

Start **Run Program=>ediscope** on the Main window Tools menu.

**Reference**
See the Using the Run Program Tool topic in this chapter.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
ediscope -{i/o} [-r <record_number>] {-cp <config path>}
```

**Note**
Do not include the brackets or braces in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **ediscope** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| i | Read *edistat.i* in the *stataud* directory for Gentran:Server Workstation, or in the *edistat* directory for Gentran:Server for UNIX. |
| o | Read *edistat.o* in the *stataud* directory for Gentran:Server Workstation, or in the *edistat* directory for Gentran:Server for UNIX*.* |
| r <record_number> | Display the contents of the specified record. |

# edixtrct

## Purpose

Extracts data from an archive file based on information from the audit record.

## Files created

This table lists the names and descriptions of the files created when you run **edixtrct**.

| File name | Description |
|---|---|
| edixtrct.log | The log file that contains the results of running the program. This file is located in the report/log directory. |

## Starting the program

Start the **edixtrct** program from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
edixtrct -{i/o} {-C <“isndr, ircvr, gsndr, grcvr”>} {-I <intchg#>}
[-G <grp#> [-S <Set#>]] {-P <outfile>} [-cp <config path>]
```

**Note**
Do not include the slashes, brackets, or braces in the command. However, do include the double-quotes (“) and commas with the -C parameter.

(Continued on next page)

## Parameters and variables

The following table lists the parameters and variables defined for the **edixtrct** command.

| Parameter | Definition |
|---|---|
| C <“isndr, ircvr, gsndr, grcvr”> | Extract data for the specified Interchange Sender, Interchange Receiver, Group Sender, and Group Receiver codes.<br>**Notes**<br>This parameter is required.<br>Enclose the codes with double quotes (“). Within the quotes, separate the codes with commas. |
| cp <config path> | The path to the configuration file (*envprim.cfg*). |
| G <grp#> | Extract data for a specific group.<br>**Note**<br>If you use the -G parameter you must also use the -I parameter. |
| I <intchg#> | Extract data for a specific interchange control number.<br>**Note**<br>This parameter is required. |
| i | Extract inbound data from an archive file.<br>**Note**<br>You must specify either -i or -o. |
| o | Extract outbound data from an archive file.<br>**Note**<br>You must specify either -i or -o. |
| P <outfile> | Save the extracted records to the specified file.<br>**Note**<br>This parameter is required. |
| S <set#> | Extract data for a specific set control number.<br>**Note**<br>If you use the -S parameter you must also use both the -I and -G parameters. |

# encreate

## Purpose

Creates a temporary file that the envelope program uses.

## Rules for use

Use only under the direction of a Gentran:Server customer support representative.

## Files created

This table lists the names and descriptions of the files created when you run **encreate**.

| File name | Description |
|---|---|
| env.dat | The file that contains the names of the files to be enveloped.<br>This file is created in the temporary directory. |
| env.idx | The index file that relates to the *env.dat* file.<br>This file is created in the *temp* directory. |

## Starting the program

This option is run automatically when you use the envelope option on the Translate Outbound Documents or Translate Inbound Documents dialog box.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
encreate [-cp <config path>]
```

**Note**
Do not include the brackets in the command.

(Continued on next page)

## Parameter and variable

This table lists the parameter and variable defined for the **encreate** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# enscope

## Purpose

A diagnostic program that prints the records of the envelope files (*env.dat* and *env.idx*) to the screen when run from the UNIX host. When run from DOS, this program prints the records to the file *enscope.log*.

**CAUTION**

**Only run this program when requested to do so by Sterling Support.**

## Files created

This table lists the name and description of the log file created when you run **enscope**.

| File name | Description |
|---|---|
| enscope.log | The log file to which the program prints the program results.<br>This file is created in the *temp* directory. |

## Starting the program

Start **Run Program=>enscope** on the Main window **Tools** menu.

**Reference**
See the Using the Run Program tool topic in this chapter.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
enscope [-cp <config path>]
```

**Note**
Do not include the brackets in the command.

(Continued on next page)

## Parameter and variable

This table lists the parameter and variable defined for the **enscope** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# envelope

## Purpose

Consolidates like interchange envelopes into one interchange envelope.

## Files created

This table lists the names and descriptions of the files created when you run the **envelope** command.

| File name | Description |
|---|---|
| env.dat | Contains the names of the files to be enveloped.<br>This file is created in the *temp* directory. |
| env.idx | The index file that relates to the *env.dat* file.<br>This file is created in the *temp* directory. |
| <filename>.env | The EDI Enveloped Data File. The **envelope** program uses *edistat.o* to determine what to use for <filename>.<br>This file is created in:<br>◗ The *stataud* directory for Gentran:Server Workstation.<br>◗ The *edistat* directory for Gentran:Server for UNIX. |
| xlcntl.err | An error file created to report the status of **envelope**.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation.<br>◗ The *temp* directory for Gentran:Server for UNIX. |

## Starting the program

Start **envelope** from:

- The **Envelope Outbound Data** option on the Translate Outbound Documents dialog box.
- The **Envelope Functional Acknowledgment** option on the Translate Inbound Documents dialog box.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
envelope {-cp <config path>}
```

**Note**
Do not include the braces in the command.

## Parameter and variable

This table lists the parameter and variable defined for the **envelope** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |

# evatrpt

## Purpose

Creates and prints a VAT Report for interchanges containing EDIFACT invoices.

**Note**
If you want to maintain the *evatrpt.r_t* file in the Gentran:Server temporary directory or to route the *evatrpt.r_t* file somewhere other than to the default printer, alter the lp_mentr script or batch file.

## Files created

This table lists the name and description of the file created when you run **evatrpt.**

| File name | Description |
|---|---|
| evatrpt.r_t | The EDIFACT VAT Report File, which stores the VAT Report for EDIFACT invoices. This report is created from data in the *evatrpt.in* and/or *evatrpt.out* files stored in the data directory.<br>The *evatrpt.r_t* file is created in the *temp* directory. |

## Starting the program

Start **evatrpt** from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
evatrpt -{i/o} [-s <start_date>] [-e <end_date>] [-l <locationID>]
[-g <gen_#>] [-f <filename>] [-cp <config path>]
```

**Note**
Do not include the braces, slashes, or brackets in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **evatrpt** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| e <end_dt> | Report EDIFACT invoices for a range of time ending with a specific end date. Specify the date to which you want to report on EDIFACT invoices in CCYYMMDD format (defaults to current date). |
| f <filename> | Report EDIFACT invoices from Archive File with the specified file name. |
| g <gen_#> | Report EDIFACT invoices with a specific file generation number. |
| i | Report incoming EDIFACT invoices. |
| l <location_id> | Report EDIFACT invoices from a specific sender’s or receiver’s location. |
| o | Report outbound EDIFACT invoices. |
| s <stdt> | Report EDIFACT invoices beginning with a specific start date. Enter the date from which you want to report on EDIFACT invoices, in CCYYMMDD format (defaults to earliest EDIFACT invoice). |

# fareport

## Purpose

Prints a Functional Acknowledgment Reconciliation Report.

**Note**

If you want to maintain the *fareport.r_t* file in the Gentran:Server temporary directory or to route the *fareport.r_t* file somewhere other than to the default printer, alter the *lp_mentr* script or batch file.

## Files created

This table lists the name and description of the file created when you run **fareport**.

| File name | Description |
|---|---|
| fareport.r_t | The FA Reconciliation Report File, which stores the FA Reconciliation Report.<br>This file is created in the temporary directory. |

## Starting the program

Start **fareport** from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
fareport -{i/o} [-T <tp_code>] [-M <set_id>] [-U<{ADELNPRVWZ}>]
[-S <st_dt>] [-B][-E <end_dt>] [-A] [-P/p] [-cp <config path>]
```

**Note**
Do not include the brackets, slashes, or braces in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **fareport** command.

| Parameter | Definition |
|---|---|
| A | Include translation and functional acknowledgment times under translation and functional acknowledgment date for each item. |
| B | Generate the report at the set level. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| E <end_date> | Print all records up to the following ending date. Requires the ending document date for the report. Use one of the following formats:<br>◗ CCYYMMDD<br>◗ null (last date in file). |
| i | Inbound functional acknowledgments. |
| M <set_id> | Print all records with a specific transaction set ID. |
| o | Outbound functional acknowledgments. |
| P | Print to standard output. |
| p | Print to a printer. |
| S <st_dt> | Print all records from the following date forward. Requires the starting document date for the report. Use one of the following formats:<br>◗ CCYYMMDD<br>◗ today (current system date)<br>◗ null (earliest date). |
| T <tp_code> | Print all records with the specified Trading Partnership Code. |
| U <status> | Print all records with the specified status code. |

(Continued on next page)

## Status codes

This table lists the valid status codes for the **fareport** command.

| Status code | Definition |
|---|---|
| A | Acknowledgment arrived okay. |
| D | Acknowledgement delivered okay. |
| E | Acknowledgment arrived with errors and was accepted. |
| L | Acknowledgment is currently late. |
| N | Acknowledgment not expected. |
| P | Acknowledgment arrived with errors and was partially accepted. |
| R | Acknowledgment arrived with errors and was rejected. |
| V | Verbal acknowledgment arrived okay. |
| W | Acknowledgment expected, but has not yet arrived. |
| Z | Verbal acknowledgment arrived with errors. |

# ftpshut

## Purpose

The **ftpshut** command is used to stop the FTP daemon. It provides an automated shutdown procedure that enables you to notify FTP Daemon users when the Daemon is shutting down.

## Files created

This table lists the name and description of the file created when you run **ftpshut**.

| File name | Description |
|---|---|
| <shutdown> | Contains all of the shutdown information from the **ftpshut** command. The FTP Daemon regularly checks for the existence of this file and will refuse new connections and notify users if shutdown is planned. The FTP Daemon remains inactive until the <*shutdown*> file is removed.<br>This file is created with the name and in the location specified by the shutdown parameter in the *ftpaccess* file. |

## Starting the program

Start the **ftpshut** command from the command line on the UNIX host.

**Note**
The **ftpshut** command is available only for the Gentran:Server with Advanced Data Distribution product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
ftpshut -m $EDI_MAILBOX [-l <min>] [-d <min>] <time>
["<warning-message>"]
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **ftpshut** command.

| Parameter | Definition |
|---|---|
| <time> | The time when the FTP Daemon will shut down. Specify the time in one of these formats:<br>“Now”=immediate shutdown<br>+n=shutdown “n” minutes from now.<br>HHMM=shutdown at the specified time of day, using a 24 hour clock format. You can only specify a time between now and 23:59. |
| <warning message> | Specifies a user-defined warning message that will appear at the end of the standard shutdown messages. The message is formatted to be 75 characters in length.<br>**Note**<br>You can include tokens that will automatically be replaced by a specific text string by the FTP Daemon. |
| d <min> | The amount of time until current connections will be disconnected. The default is five minutes, or immediately if <time> is set to less than five minutes. |
| l <min> | The amount of time until new FTP access will be disabled. The default is 10 minutes, or immediately if <time> is set to less than 10 minutes. |
| m $**EDI_MAILBOX** | Specifies the path to the *mb* directory on the UNIX host. You may type in the full path to the directory, or type the $EDI_MAILBOX environment variable to have the system determine the value. |

## Warning message tokens

This table lists the tokens available for the warning message parameter.

| Token | Definition |
|---|---|
| %C | The current working directory. |
| %d | The time current connections will be dropped. |

(Continued on next page)

| (Contd)<br>Token | Definition |
|---|---|
| %E | The FTP Daemon maintainer’s e-mail address, as defined in the *ftpaccess* file. |
| %L | The local host name. |
| %M | The maximum number of users allowed in this class. |
| %N | The current number of users in this class. |
| %r | The time new connections will be denied. |
| %R | The remote host name. |
| %s | The time the system is going to shut down. |
| %T | The local time presented in the following format:<br>Thu Nov 15 17:12:42 1990) |
| %U | The username given at login. |

# genmbid

## Purpose

The **genmbid** utility generates the unique mailbag identification code (**mbagid**) to track a data file throughout its life in Gentran:Server.

## Rules for usage

The **genmbid** command works only if your organization uses the Mailbag Identification Filename Convention.

**Reference**
See the Data Tracking topic in the Gentran:Server *for UNIX Advanced Data Distribution Guide* for details.

## Starting the program

Start **genmbid** from:

- A script.
- The UNIX command line.

**Note**
The **genmbid** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
genmbid
```

# genuniqid

## Purpose

The **genuniqid** command generates a unique code for a file and optionally appends the code to a specified file name.

## Rules for usage

Use the **genuniqid** command when you need a unique file name for files that would have a duplicate prefix. It is often used in the Advanced Data Distribution product where more than one line manager receives input from the trading partner. Because all the line managers receiving the input will use the same file name, it is important to have a way to add a unique code to the name of each file to distinguish them from each other.

## Starting the program

Start **genuniqid** from:

- The **checkit** shell script
- Gentran:Server scripts
- UNIX shell scripts

**Note**
The **genuniqid** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
genuniqid -d
```

## Parameters and variables

This table lists the parameters and variables defined for the **genuniqid** command.

| Parameter | Definition |
|---|---|
| d | Returns a nine character base-ten unique code. The default is to return a six character unique code. |

# isops

## Purpose

Creates, loads, and unloads the indexed sequential (ISAM) files used within Gentran:Server. The unload option converts ISAM files to ASCII format with a .unl extension. The load option converts the ASCII .unl file back to ISAM files.

## When to use

Use **isops** during installation and configuration to create ISAM files and to load default configuration information. You can use **isops** to unload files so that you can edit them, and then reload them after you finish editing. Additionally, the program is used in conjunction with the patterns and melding feature to load entries to a data manager configuration record. The files are new-line terminated, variable-length records. Fields are delimited by vertical bars. You can change the vertical bar delimiter to another symbol.

## Starting the program

Start **isops** from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation
- The **Run Program** option on the Main window Tools menu.

## Using the Run Program tool

You must have security access to run **isops** from the Run Program tool. When you select **isops** from the Command List, Gentran:Server displays the argument list appropriate for your product level.

Note that the e <errfile> parameter is not available from the Run Program option.

When **isops** is finished, the system displays the log file *isops.log*. This file shows the number of new, duplicate, and bad records processed. **isops** creates the log file:

- In the user temp directory on a host machine, when you run the program using client-based tools.
- In the directory **$EDI_ROOT/temp** on a host machine, when you run the program using UNIX-based tools.
- In the report directory defined in the *envprim.cfg* file when you run the program on a Workstation PC.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
isops -[l/ln/lw/u] {-f <fname>} [-dx] [-c] [-e <errfile>]
{-w <workfile> < <stdin> > <stdout> > <stderr>}
```

**Note**
Do not include the slashes, braces, or brackets in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **isops** command.

| Parameter | Definition |
|---|---|
| c | Creates destination directories (used with -l, for “dm” file only). |
| dx | Delimits input/output with “x” (default is “\|”). |
| e <errfile> | Redirects stderr messages to specified file. Available at the command line only. |
| f <fname> | Specifies the name of the file that you want to create, load, or unload. |
| l | Converts the ASCII .unl file to ISAM format and loads the data to the existing ISAM files. The program creates the ISAM file if it doesn’t exist. If you load a duplicate record, it overwrites the existing record. The program reads records from standard input, stdin. |
| ln | Converts the ASCII .unl file to ISAM format and loads the data to the existing ISAM files. The program creates the ISAM file if it doesn’t exist. If you load a duplicate record, it skips (does not load) the duplicate record. |
| lw | Converts the ASCII .unl file to ISAM format and loads the data to the existing ISAM files. The program creates the ISAM file if it doesn’t exist. It loads duplicate records if you allow duplicates. |
| u | Unloads data from the ISAM files and converts the data to an ASCII .unl file. The program writes records to stdout. |

(Continued on next page)

| (Contd) Parameter | Definition |
|---|---|
| w <workfile> | When used with -l, redirects the input file from which the program reads records. When used with -u, redirects the output file to which the program writes records.<br>**Note**<br>You can use this argument in place of the redirection symbols < and >. |
| **<** stdin | Redirects the standard input file from which the program reads records (used with -l).<br>**Note**<br>You must include the redirection symbol (<). |
| **>** stdout | Redirects the standard output file to which the program writes records (used with -u).<br>**Note**<br>You must include the redirection symbol (>). |
| **>** stderr | Redirects the standard error file to which the program writes all user messages.<br>**Note**<br>You must include the redirection symbol (>). |

## Configuration files

This table lists the file names defined for the **isops** command and the product levels to which they apply.

| File name | Definition | Product Level |
|---|---|---|
| apptptbl | Application Trading Partnership Rules Value file | Gentran:Server for UNIX with Process Control Manager and higher. |
| appxref | Application Trading Partnership Code Cross Reference Value file | Gentran:Server for UNIX with Process Control Manager and higher. |
| catvalue | Trading Partner category values file. | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX.<br>(Continued on next page) |

| (Contd)<br>File name | Definition | Product Level |
|---|---|---|
| contact | Trading Partner contact file. | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |
| dlname | The distribution list file. | Gentran:Server for UNIX with Advanced Data Distribution. |
| dm | The data manager configuration record file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| <dm>_arch | The archive file for the specified data manager. | Gentran:Server for UNIX with Process Control Manager and higher. |
| .dmcfg | The data manager configuration file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| dstlst | The distribution mailbox list file. | Gentran:Server for UNIX with Advanced Data Distribution. |
| ds_map | The EDI document reference number specifier mapping file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| ds_map_app | The APP document reference number specifier mapping file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| ds_map_ncp | NCPDP document reference number specifier mapping file | Gentran:Server for UNIX with Process Control Manager and higher. |
| ds_map_xml | XML document reference number specifier mapping file | Gentran:Server Workstation with the XML translation option.<br>All levels of Gentran:Server for UNIX with the XML translation option. |
| ds_name | The document reference number specifier file. | Gentran:Server for UNIX with Process Control Manager and higher.<br>(Continued on next page) |

| (Contd)<br>File name | Definition | Product Level |
|---|---|---|
| ds_tptbl | The EDI document reference number specifier Trading Partnership file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| ds_tptbl_app | The APP document reference number specifier Trading Partnership file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| ds_tptbl_ncp | The NCPDP document reference number specifier Trading Partnership file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| ds_tptbl_xml | The XML document reference number specifier Trading Partnership file. | Gentran:Server Workstation with the XML translation option.<br>All levels of Gentran:Server for UNIX with the XML trans-lation option. |
| org | The Trading Partner organization file. | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX |
| pat | Pattern file | Gentran:Server for UNIX with Process Control Manager and higher. |
| patlst | Pattern list file | Gentran:Server for UNIX with Process Control Manager and higher. |
| .q | The queue file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| .scrcfg | The script configuration file. | Gentran:Server for UNIX with Process Control Manager and higher. |
| stdin | The standard input file from which the program reads records (used with -l). | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |
| stdout | The standard output file to which the program writes records (used with -u). | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |

(Continued on next page)

| (Contd)<br>File name | Definition | Product Level |
|---|---|---|
| stderr | The standard error file to which the program writes all user messages. | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |
| tp | The Trading Partner file. | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |
| tpmisc | Miscellaneous Trading Partner information file. | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |
| tprecon | Trading Partner reconciliation file. | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |
| tradacom | TRADACOM Trading Partner files | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |
| trn | The transaction register file | Gentran:Server for UNIX with Process Control Manager and higher. |
| xmlspl<n> | The XML splitting element table for a specific level of element, where <n> is 1, 2, or 3. | Gentran:Server Workstation with the XML translation option.<br>All levels of Gentran:Server for UNIX with the XML translation option. |
| xmltptpl | The XML TP Rule table | Gentran:Server Workstation with the XML translation option.<br>All levels of Gentran:Server for UNIX with the XML translation option.<br>(Continued on next page) |

| (Contd)<br>File name | Definition | Product Level |
|---|---|---|
| xmlxref | The XML cross-reference table | Gentran:Server Workstation with the XML translation option.<br>All levels of Gentran:Server for UNIX with the XML translation option. |
| xtable | The Standard Cross Reference Table file | Gentran:Server Workstation.<br>All levels of Gentran:Server for UNIX. |

## isops examples

Here are several examples of ways to use **isops**.

### Example 1

This **isops** command displays the contents of the data manager file to the screen.

```
isops -u -f dm
```

### Example 2

This **isops** command unloads the file ds_name to ds_name.unl.

```
isops -u -f ds_name > ds_name.unl
```

### Example 3

This **isops** command loads the file ds_name from ds_name.unl.

```
isops -l -f ds_name < ds_name.unl
```

### Example 4

This series of **isops** commands shows how to use **isops** on the *trn.dat* and *trn.idx* files after you mark them using **clean_trn**.

```
isops -u -f trn > trn.unl
rm trn.dat trn.idx
isops -l -f trn < trn.unl
```

### Example 5

This **isops** commands shows how to use **isops** to delete entry markers for deletion by **clean_trn.** You should back up your data before initiating this command to prevent data loss.

```
isops -u -f trn > trn.unl
```

# lcld

## Purpose

The **lcld** script loads event records (*dmnm.v*) to the Life Cycle database tables, *lc221* and *LCDestInfo*.

### Comment

The lcld executable is created during the process of setting up the Life Cycle utility.

### Reference

See the Gentran:Server *for UNIX Data Flow Administration Guide* for Life Cycle setup instructions.

## Files created

This table lists the names and descriptions of the files created when you run **lcld**.

| File name | Description |
|---|---|
| <dmname>.l | The data manager log file for the data manager whose event record is being loaded by Life Cycle. Each entry consists of the last 25 lines from *<dmname>.log*.<br>This file is appended to in the *EDI_ROOT* directory. |
| <dmname>.log | The log file containing the status of the database load process.<br>This file is created in the *EDI_ROOT/lcld* directory. |
| <dmname>.v.last | Only created when the database load process succeeds, this log file indicates the last agent event file that **lcld** loaded.<br>This file is created in the *EDI_ROOT/lcld* directory. |
| <dmname>_<uniquid>.v.date | Only created when the database load process fails, this log file indicates the last agent event file that **lcld** attempted to load.<br>This file is created in the *EDI_ROOT* directory. |

(Continued on next page)

## Starting the program

Start **lcld** from the UNIX host command line.

**Note**
The **lcld** command is available only for the Gentran:Server with Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
lcld {-f<filename>}
```

**Notes**
Do not include the braces in the command.

## Parameter and variable

This table lists the parameter and variable defined for the **lcld** command.

| Parameter | Definition |
|---|---|
| f<filename> | Loads the specified data manager event file. The location of the event file is defined in each data manager’s initialization file. The LIFE_CYCLE_DIR parameter sets the location. The default is the *lcld* directory off of the directory specified for $EDI_ROOT. |

# lftran Overview

## Introduction

The **lftran** program works in conjunction with the Gentran:Server translator components to translate data. It supports both the Application Integration mapper and the Visual Mapper separately.

**CAUTION**

**The translator can process a single input file that contains data for both mappers. This applies to data formatted to the X.12, EDIFACT, VDA, and GENCOD standards.**

**The translator is unable to process a single input file that contains data for both mappers if the data is formatted to the TRADACOM standard. You must split the file (for example, with a data manager) to process the data.**

## lftran topics

See these topics for information about lftran.

| For information about… | See… |
|---|---|
| The files that lftran creates during translation | Files Created During Translation |
| The command syntax and the parameters for the command | lftran Syntax |
| How to force **lftran** to write new files instead of appending to or overwriting existing files | Changing How lftran Writes Files |
| How to manage the control numbers generated during translation | Managing Control Numbers |

# Files Created During Translation

## Introduction

This topic describes all possible files that lftran creates during inbound or outbound translation.

## Document reference number

The **lftran** program generates the document reference number for:

- All translations in which the input document is in a Standard format.
- Application and XML translation where the output of the translation is in a Standard format.

## Files created: inbound translation

This table lists and describes the files created during an inbound translation.

| File name | Description |
|---|---|
| boxin.err | An error file created to store EDI data that the translator was unable to process.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| boxin.lok | A temporary file created to store EDI data that the translator is temporarily unable to process until the required Trading Partnership record (or associated *org.dat/.idx* file) is unlocked.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| edistat.i | The temporary Audit File created for the inbound EDI document(s) and used by **ediarc**.<br>This file is created in:<br>◗ The specified EDI Status directory for Gentran:Server Workstation<br>◗ The *edistat* directory for Gentran:Server for UNIX. |

(Continued on next page)

| (Contd)<br>File name | Description |
|---|---|
| edistat.o | The temporary Audit File created for the outbound acknowledgment (if the Trading Partnership record is set to send a functional acknowledgment) and used by **ediarc**.<br>This file is created in:<br>◗ The specified EDI Status Audit directory for Gentran:Server Workstation<br>◗ The *edistat* directory for Gentran:Server for UNIX. |
| evatrpt.in | The file that contains the Control Report record for each EDIFACT interchange if the user wants to generate a VAT report.<br>This file is created in the *data* directory. |
| <Record file layout> | The output file generated as a result of inbound translation.<br>This file is created with the name and in the directory specified in the Trading Partnership record. |
| sigout.err | The error file generated to store unenveloped sets with an invalid Trading Partnership marker.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| sigout.lok | The error file created to store un-enveloped sets with Trading Partnership markers we could not process because the Trading Partnershp file was locked.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| tvatrpt.in | The file that contains the Control Report record for each TRADACOMS interchange.<br>This file is created in the *data* directory. |
| xlcntl.err | An error file created to report the status of translation.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation.<br>◗ The specified user files directory for Gentran:Server for UNIX.<br>(Continued on next page) |

| (Contd)<br>File name | Description |
|---|---|
| transopt.in | A file that contains only the parameter settings of the programs that were executed during translation. This file is created only when translation is run from the Translate menu.<br>This file is created in:<br>◗ The local temporary directory on the Windows client for Gentran:Server for UNIX.<br>◗ The specified user files directory for Gentran:Server Workstation. |
| trans.ord | A file that contains the Diagnostics Report. This report indicates the order in which Gentran:Server executes your mapping instructions.<br>This file is created in:<br>◗ The specified user files directory on the Windows client for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Files created: outbound translation

This table lists and describes all possible files created during an outbound translation.

| File name | Description |
|---|---|
| boxout.err | An error file created to store record file layout data that the translator was unable to process.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| boxout.lok | A temporary file created to store record file layout data that the translator is temporarily unable to process until the required Trading Partnership record (or associated *org.dat/ .idx* file) is unlocked.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX.<br>(Continued on next page) |

| (Contd)<br>File name | Description |
|---|---|
| EDI file | The output file generated as a result of outbound translation.<br>This file is created with the name and in the directory specified in the Trading Partnership record. |
| edistat.o | The temporary Audit File created for the outbound document(s) and used by **ediarc**.<br>This file is created in:<br>◗ The specified EDI Status Audit directory for Gentran:Server Workstation.<br>◗ The *edistat* directory for Gentran:Server for UNIX. |
| evatrpt.out | The file that contains the Control Report record for each EDIFACT interchange if the user wants to generate a VAT report.<br>This file is created in the *data* directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| sigout.err | The error file generated to store unenveloped sets with an invalid Trading Partnership marker.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| sigout.lok | The error file created to store un-enveloped sets with Trading Partnership markers we could not process because the Trading Partnershp file was locked.<br>This file is created in the specified user files directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| tvatrpt.out | The file that contains the Control Report record for each TRADACOMS interchange.<br>This file is created in the *data* directory for both Gentran:Server Workstation and Gentran:Server for UNIX. |
| xlcntl.err | An error file created to report the status of translation.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation.<br>◗ The specified user files directory for Gentran:Server for UNIX.<br>(Continued on next page) |

| (Contd)<br>File name | Description |
|---|---|
| transopt.in | A file that contains only the parameter settings of the programs that were executed during translation. This file is created only when translation is run from the Translate menu.<br>This file is created in:<br>◗ The local temporary directory on the Windows client for Gentran:Server for UNIX.<br>◗ The specified user files directory for Gentran:Server Workstation. |
| trans.ord | A file that contains the Diagnostics Report. This report indicates the order in which Gentran:Server executes your mapping instructions.<br>This file is created in:<br>◗ The specified user files directory on the Windows client for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

# Iftran Syntax

## Purpose

The **Iftran** program works in conjunction with the Gentran:Server translator components to translate data. It supports both the Application Integration mapper and the Visual Mapper.

## Starting the program from the menu

Select **Translate Document** from the Translate menu.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
lftran <infile> -[i/o/y] {A<appfile> a b<dsname> C# c D d E e F f g
H H0 H1 h I J j K k L# l M# m n OP p Q q R r S s# T<tpcode> T<x>,<y>
t U u V v w X Z z} [<outfile>] [-cp <config path>]
```

### Notes

- Some translation options are not used with Application Integration maps.
- Do not include the slashes, brackets, or braces in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **Iftran** command. An X or parameter in the VM (Visual Mapper) or AI (Application Integration) indicates the mapper type for which each option is used.

| Parameter | Definition | VM | AI |
|---|---|---|---|
| i/o/y | The direction of translation. This parameter is required.<br>◗ i = inbound translation (input file is in an EDI standard format)<br>◗ o = outbound translation (if input is in an EDI standard format, the output is also)<br>◗ y = XML translation (input file is in XML format)<br>**Note**<br>Use only one direction parameter at a time.<br>(Continued on next page) | i, o | i, o, y |

| (Contd)<br>Parameter | Definition | VM | AI |
|---|---|---|---|
| A<appfile> | Look in the specified application file definition to find the Trading Partnershp Code.<br>**Note**<br>If lftran cannot find the Trading Partnership Code, it looks for a set of rules in the apptptbl table to extract the code. | X | X |
| a | Do not generate or write audit records. | X | X |
| b | Perform database translation using the data source specified in the application definition used by the map. | X | X |
| b<dsname> | Perform database translation using the specified data source name. | X | |
| C1 | Total the number of line items in the set and write the sum in the outbound CTT01 element.<br>**Comment**<br>Do not use if you mapped the CTT segment. | X | |
| C2 | Total the number of line items in the set and write the sum in the outbound CTT01 element.<br>Also calculate the hash total and write the result in the CTT02 element.<br>**Comment**<br>Do not use if you mapped the CTT segment.<br>(Continued on next page) | X | |

| (Contd)<br>Parameter | Definition | VM | AI |
|---|---|---|---|
| c | Perform compliance checking on the EDI data file. Use the map and the *.CND* file for the standard to identify:<br>◗ Missing mandatory segments or elements<br>◗ Elements longer than the maximum or shorter than the minimum<br>◗ Invalid dates or times<br>◗ Alphanumeric data in a numeric field<br>◗ Too many elements in a segment<br>◗ Segment count and SE01 do not match<br>◗ Presence and validity of conditional elements<br>◗ Occurrences of a segment exceed Max Use.<br>**Note**<br>In the Application Integration subsystem, compliance checking is enabled at the map level. Integration map, Gentran:Server ignores it. | X | |
| cp <config path> | The path to the configuration file (*envprim.cfg*).<br>**Note**<br>This applies only to the Visual Mapper. | X | |
| D | Generate *dbaudit.i* file based on translation to a database. | X | |
| d | Write segment IDs to *xlcntl.err* during translation and write the Diagnostics Report File, *trans.ord*, to the Gentran:Server temporary directory.<br>**Note**<br>For Application Integration translation, *trans.ord* contains limited information. | X | X |
| E | Do not write application error messages to *xlcntl.err.*<br>**Comment**<br>Because application errors are warnings, not errors, many sites prefer not to record them. | X | |

(Continued on next page)

| (Contd)<br>Parameter | Definition | VM | AI |
|---|---|---|---|
| e | Cancel creation of an output file if translation finds application errors (message numbers lower than 100) in the data.<br>For accurate application data:<br>◗ Generate output<br>◗ Set the audit record result field to 0.<br>For erroneous application data:<br>◗ Do not generate output<br>◗ Write warning messages to the *xlcntl.err* file<br>◗ Set the audit record result field to 1.<br>**Comment**<br>The default for erroneous application data is:<br>◗ Generate output<br>◗ Write warning messages to the *xlcntl.err* file<br>◗ Set the audit record result field to 0. | X | X |
| F | Disable flushing output records at beginning of new input loop. Instead, retain values from previous loops to fill empty fields in the current record before creating a new record. | X | |
| f | Override Trading Partnership output file name with output file specified in <outfile> variable. | X | X |
| g | Fill the B501 element for a 999 set with the set ID from the last set encountered.<br>**Comment**<br>Because the B501 element is not required, creating 999s normally does not fill that element. | X | X |

(Continued on next page)

| (Contd) Parameter | Definition | VM | AI |
|---|---|---|---|
| H | Add envelope segments to the specified file and create an audit record for each set.<br>**Comment**<br>The expected input file format is a record containing the Trading Partnership code associated with the data in the file, followed by one or more properly formatted EDI documents (minus any envelope segments). The punctuation defined in the specified Trading Partnership record, such as separators and terminators, must match the punctuation in the formatted EDI documents. | X | X |
| H0 | Do not add envelope segments to the output transactions at this time, but do generate a Trading Partnership signature to enable delayed enveloping.<br>**Comment**<br>This option is used in conjunction with the H1 option to enable delayed enveloping. | X | X |
| H1 | Generate interchange and group envelopes based on the Trading Partnership signatures in the transactions sets and add the envelopes to the transaction sets. Lock the Trading Partnership records during enveloping to update the control numbers.<br>**Comment**<br>This option is used in conjunction with the H0 option to enable delayed enveloping. | X | X |
| h | Validates totals in the CTT segments (inbound).<br>If it finds errors in the CTT totals, translation displays one or both of the following messages:<br>◗ Incorrect line item count in CTT01<br>◗ Incorrect hash total in CTT02<br>**Comment**<br>This option implies the c option (compliance checking). | X | |

(Continued on next page)

| (Contd)<br>Parameter | Definition | VM | AI |
|---|---|---|---|
| I | Envelope each individual transaction set with interchange and group envelopes.<br>**Comment**<br>This is the default enveloping option. | X | X |
| <infile> | Pull input from the specified file. | X | X |
| J | For EDIFACT processing, generate new version of CONTRL message. | X | X |
| j | Only write AK2 loop in 997 if all sets accepted without errors. | X | X |
| K | If Trading Partner record is locked when translation is attempted, retain lock until a new interchange is started. For inbound translation, send entire interchange to *boxin.lok* file. For outbound translation, send entire file except translated portion to *boxout.lok* file.<br>**Note**<br>Normally, lftran wraps each set written to *boxin.lok* inside a separate interchange. When you use -K, lftran writes sets in the same interchange structure as in the input file. If the Trading Partner record is locked when lftran processes an interchange containing five sets, it writes the interchange and group start headers, followed by the five sets, followed by the group and interchange end headers.<br>**Comment**<br>Use this parameter when all of the following are true:<br>◗ You are running multiple translation processes at the same time<br>◗ Each input file for outbound translation uses only one Trading Partner record<br>◗ Within the input file for inbound translation, each interchange uses only one Trading Partner record<br>◗ You want to keep interchanges or files together in the event that a Trading Partner record is not available at the time the translation tries to access it.<br>(Continued on next page) | X | X |

| (Contd)<br>Parameter | Definition | VM | AI |
|---|---|---|---|
| k | Perform compliance checking without checking date fields.<br>**Note**<br>In Application Integration, compliance checking is set within the map. | X | |
| L# | Enable multiple attempts to lock Trading Partnership record if first attempt fails, where # is the maximum number of lock attempts at one second intervals. | X | X |
| l | Create records/segments that consist of only literal mappings.<br>**Comment**<br>Normal processing does not create records/ segments that consist of only literal mappings. | X | |
| M# | Use the # format for the CONTRL message, where:<br>◗ M1 = Use the pre-1993 format<br>◗ M2 = Use the 1993 and later format.<br>**Comment**<br>Normal processing uses the format appropriate for the year of the standard version used in the document you are acknowledging. | X | X |
| m | If an X12 standard is used, append the character found in ISA15 (P/T) to the output file name.<br>If an EDIFACT standard is used, append the character found in UNB-0035 (1/0) to output file name.<br>**Comments**<br>Do not use if you intend to override the output file name. | X | X |
| n | Allow translator to zero fill fixed length real/ numeric flat file fields. | X | X |
| O | Use alternate processing of GENCOD loops and dates. This parameter allows the translator to process a conditional loop data is found in the second segment of the loop but not in the loop header.<br>(Continued on next page) | X | |

| (Contd)<br>Parameter | Definition | VM | AI |
|---|---|---|---|
| <outfile> | Write output to the specified file. | X | |
| P | Use alternate Trading Partnership lookups to find key fields specific to EDIFACT. | X | X |
| p | Append “P” (for production) to output file name. | X | X |
| q | Remove the ID elements for a segment if there are no data elements following the ID element in the segment. | X | |
| Q | Create UCM segment in CONTRL message, even if no errors exist in the set. | X | X |
| R | Check all EDIFACT and ODETTE, and TRADACOMS data items for special characters that are also used as element separators or terminators. Insert a release character before each special character within the data to allow proper processing. | X | |
| r | Allow different handling of application error found during translation. If an application error occurs anywhere within the condition, the entire condition fails, and the affected segment is not written. | X | |
| S | Treat control number sequence violations as errors instead of warnings, and cancel writing output. Write input data to *boxin.err.*<br>(Continued on next page) | X | X |

| (Contd)<br>Parameter | Definition | VM | AI |
|---|---|---|---|
| s# | For inbound translation use a wildcard character ($) in a key field # in Trading Partnership record where:<br>◗ s1=Use wildcard in **Your Interchange ID** key field<br>◗ s2=Use wildcard in **Partner's Interchange ID** key field<br>◗ s3=Use wildcard in **Your Group ID** key field<br>◗ s4=Use wildcard in **Partner's Group ID** key field<br>◗ s5=Use wildcard in **Standard Version** key field<br>◗ s6=Use wildcard in **Document Version** key field<br>**Comment**<br>The translator looks for Trading Partner Records that contain a wildcard in the specified field. | X | X |
| T<tpcode> | Use the specified Trading Partnership Code for all sets in the input file. | X | X |
| T<x>,<y> | Use <x> and <y> to find Trading Partner-ship Code in application file record where:<br><x>=Starting position of Trading Partnership Code<br><y>=Length of Trading Partnership Code | X | X |
| t | Append "T" (for test) to output file name.<br>**Comment**<br>Do not use this parameter if you override the output file name. | X | X |
| U | Do not lock Trading Partnership records during translation.<br>**Caution**<br>Use of this parameter can cause duplicate control numbers. | X | X |
| u | For EDIFACT processing, do not create INVTLR information from TAXCON message. | X | X |

(Continued on next page)

| (Contd) Parameter | Definition | VM | AI |
|---|---|---|---|
| V | Generate EDIFACT VAT Report from TAXCON messages. | X | X |
| v | Use the .CDS standard file to validate ID codes sent or received against codes allowed by the standard being used.<br>**Comments**<br>This option automatically activates compliance checking for Visual Mapper maps.<br>This feature is built into Application Integration maps, so Gentran:Server ignores this setting. | X | |
| w | Write all current records or segments to the output file as soon as a new copy of the record or segment is created.<br>**Comment**<br>If you have exceptionally large documents, using this parameter frees up memory after each record is written. The normal process reads in an entire set before writing it out. | X | |
| X | For EDIFACT processing, do not create a TAXCON message. | X | X |
| Z | Translate using the checked-out version (the working compiled map file/translation object) of the *.TBL or .TPL* file, instead of the *.TBL or .TPL* file from the master directory.<br>**Comment**<br>If you use this parameter when you run lftran from the command line, you must use the -cp <config path> parameter to specify the path to the user who has the file checked out. This enables lftran to check the *envprim.cfg* file of that user to find the correct working maps directory. | X | X |
| z | If there is no mapped data, do not zero-fill date fields in fixed length application files. | X | |

# lftran Return Codes

**lftran** This table describes the lftran return codes.

| Return Code | Meaning | Symbol |
|---|---|---|
| 1 | Reached the end of the file unexpectedly | FILEEND |
| 2 | Error in invocation of translation | ARGERR |
| 3 | Unable to load translation table (compiled map/translation object) | TBLERR |
| 4 | Unable to open a file | OPNERR |
| 5 | Could not read, write, or update Trading Partnership record | TPERR |
| 6 | Application file is not formatted as expected. Could be corrupt. | APPFMTERR |
| 7 | Could not find Trading Partnership Code in the application file | APPTPERR |
| 8 | Ran out of memory | ALLOC_FAIL |
| 9 | Call to database failed | DBERR |
| 10 | The version of the translator does not match the product version | VERSION |
| 12 | Error occurred during compliance checking | CMPCHKERR |
| 13 | Failed to load the ID code file into memory | IDCODEERR |
| 14 | Error in application data | APPDATAERR |
| 15 | Error parsing NCPDP file | RECIDERR |
| 16 | Input file is not new-line terminated | NO_NEWLINE |
| 17 | Could not read, write, or update organization records | ORGERR |

(Continued on next page)

| (Contd) Return Code | Meaning | Symbol |
|---|---|---|
| 20 | Incoming set sequence number not as expected | SEQUENCE_ERROR |
| 25 | Error parsing the XML file | XMLFILEERR |
| 26 | Translation error. See the dtlLog.err file for explanation. | COMMON_LIB_ERROR |
| 50 | Duplicate sets found | DUPLICATE_SET |
| 80 | Unable to determine the type of the input record | UNDEF_REC_TYPE |

# Changing How lftran Writes Files

## Introduction

To change how **lftran** writes files, you can set environment variables to specify a path and/or file name different than the defaults shown in the invocation.

## Default operation

In normal operation **lftran** overwrites the *edifrmat.not* file each time it runs.

## The environment variables

You can prevent **lftran** from appending to files by setting selected environment variables immediately before you run the program. Instead, **lftran** will write new files each time it executes. **lftran** works with these environment variables.

| Environment Variable | Prevents appending to the |
|---|---|
| AUDITI | *edistat.i* file |
| AUDITO | *edistat.o* file |
| BOXINERR | *boxin.err* file |
| BOXOUTERR | *boxout.err* file |
| BOXINLOK | *boxin.lok* file |
| BOXOUTLOK | *boxout.lok* |
| XLCNTLERR | *xlcntl.err* |

(Continued on next page)

## Setting the environment variables

Use this table to find the syntax of the environment commands you need to use.

| IF you run lftran from… | AND you use a… | THEN use the syntax… |
|---|---|---|
| Gentran:Server Workstation | -- | set <ENVIRONMENTAL_VARIABLE>=.\<directory>\<new file name><br>**Note**<br>Upon completion of the command, you must reset or clear the file name environment variables to the default values. |
| Gentran:Server for UNIX | Korn or Bourne shell script | BOXOUTERR=./<directory>/<filename>;export BOXOUTERR |
| | C shell script | setenv BOXOUTERR ./<directory>/<filename> |

# Managing Control Numbers

## Introduction

You can specify the interchange and application/group control number to establish or override the last used.

## XLCTLNUM environment variable

The XLCTLNUM environment variable uses the same number for both the interchange and the application/group control number. Gentran:Server increments this number for each interchange envelope created during the current translation session, but does not write this number back to the Trading Partnership record.

If you interactively reset the XLCTLNUM environment variable each time immediately before you run **lftran**, you can run **lftran** repeatedly in a single session and always have complete control over the control numbers.

## Using XLCTLNUM with U parameter

This environment variable is especially useful when you use the U parameter with the **lftran** program. The U parameter prevents locking of trading partner records during the translation process; thus the translator never waits for a locked trading partner record. However, the U parameter also prevents Gentran:Server from writing control numbers back to the Trading Partnership record. This may allow duplicate control numbers. To avoid duplicate control numbers, use the XLCTLNUM environment variable.

## Setting the environment variables

Use this table to find the syntax of the environment commands you need to use.

| IF you run lftran from a… | THEN use the syntax… |
|---|---|
| DOS batch file | set XLCTLNUM=<number> before the **lftran** command. |
| Korn or Bourne shell script | XLCTLNUM=<number>;export XLCTLNUM |
| C shell script | setenv XLCTLNUM ./<directory>/<filename> |

(Continued on next page)

## Rules for usage

Follow these rules when using the XLCTLNUM environment command.

- Place the XLCTLNUM environment command before the **lftran** command.
- Whenever you want to run **lftran** repeatedly within the same DOS batch file or UNIX script, you must reset the XLCTLNUM environment variable before each **lftran** command.
- After you run a DOS batch file you must reset the XLCTLNUM environment variable to the default.

# ltb_info

### Purpose

The **ltb_info** command displays information about the environment processes (**mhp_server** and **mhs_server**) registered with a particular broker process.

### When to use

Use the **ltb_info** command to determine which environments have processes running on the UNIX host.

You may also use **ltb_info** to identify and then delete environment processes that remain registered with a broker process after either the broker process or the environment processes stop abnormally.

For example, if someone forgets and uses the UNIX **kill -9** command to stop **mhs_server** and **mhp_server** processes, they remain registered with the broker process and cannot be restarted until they are deleted.

**CAUTION**

**Do not use the kill -9 command to stop Gentran:Server processes. We recommend that you use the stoprpcs.sh command to stop mhs_server and mhp_server processes. See the "stoprpcs.sh" topic in this chapter.**

### Starting the program

Start **ltb_info** from the /broker directory of the UNIX command line.

**Note**
The **ltb_info** process is available with Gentran:Server for UNIX and higher product levels.

### Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
ltb_info -b <broker> -l -{p <progname>/v <ver_name>/a <attributes>}
-d [<host> <pnum>] [vnum=1]
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **ltb_info** command.

| Parameter | Definition |
|---|---|
| <host> | System name or IP address of host containing the process to delete. |
| <pnum> | Program number to delete.<br>**Comment**<br>You can run the **ltb_info -l** command to obtain the program number. It is listed in the second column of the data that is displayed on screen. |
| a <attributes> | Reserved for future use. |
| b <broker> | List all environment processes registered with the specified broker process. Specify the broker using either the IP address or the system name of the host on which the broker process runs. |
| d | Delete specified server processes.<br>**Comment**<br>If you use the -d option, do not use the -l option.<br>Use the -d option only to remove server processes that have been halted but remain registered with the broker. Normally you should use the stoprpcs.sh command to halt environment processes. |
| l | List **mhs_server** and **mhp_server** processes registered with the specified broker. The -l option used alone lists all registered processes.<br>**Comment**<br>If you use the -l option, do not use the -d option. |
| p <progname> | List server processes with the specified name and registered with the specified broker. Valid program names are:<br>◗ MHS= list registered **mhs_server** processes<br>◗ MHP= list registered **mhp_server** processes |
| v <ver_name> | Reserved for future use. |
| vnum=1 | Reserved for future use. |

# mentcomp

## Purpose

Compiles Visual Mapper maps for either the platform from which it is called or for another platform. The compiler automatically sorts the mapping instructions.

## Files created

This table lists the names and descriptions of files created when you compile Visual Mapper maps.

| File name | Description |
|---|---|
| <mapname>.TBL | The mapping table file that contains the result of compiling the map file named *<mapname>.vmp.*<br>This file is created in the user’s working directory.<br>**Note**<br>When you check in the map, you save the working copy to the *maps* directory. |
| <mapname>.TAB | The mapping table temporary file, which contains the ‘mapname’ map that was compiled for a platform other than the current platform.<br>This file is created in the user’s working directory.<br>**Note**<br>When you check in the map, you save the working copy to the *maps* directory. |
| mentcomp.err | The log file created by the **mentcomp** command.<br>This file is created in the *rpt* directory. |

## Starting the program

Start **mentcomp** from:

- The **Compile** option on the File menu in either the Main window or the Visual Mapper for Gentran:Server Workstation
- The **Compile** option on the Translate menu in either the Main window or the Visual Mapper for Gentran:Server for UNIX
- The **Compile** Toolbar button in the Visual Mapper.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
mentcomp {-0/1/2} [f <map_list>/l <map_list_file>/d <map_dir>/a/v/
w/] {list of maps/file with list of maps/name of directory}
```

**Note**
Do not include the slashes, brackets, or braces in the command.

## Rules

Use the following rules when entering the **mentcomp** command.

- Do not use the Workstation version of **mentcomp** from within a nested batch file. Nested batch files should contain only DOS commands and DOS command line programs.
- Before using the Client/server version of **mentcomp,** be sure that MOTIF is running.

## Parameters and variables

This table lists parameters and variables defined for the **mentcomp** command.

| Parameter | Definition |
|---|---|
| 0 | Compile the specified maps for INTEL processor-based platforms (for example, Windows, Windows 95, etc.). |
| 1 | Compile the specified maps for RISC processor-based platforms (for example, most UNIX platforms: HP, RS6000, SOLARIS, SUN etc.). |
| 2 | Compiles the specified maps for the 64-bit processor-based platforms (for example, DEC ALPHA). |
| a | Compile all maps in the directory you specified for maps in the Location of Files dialog box. |
| c <path> | Compile all maps in the specified location. |
| d <map_dir> | Compile all maps in the specified directory. Requires the path and name of a specific directory containing a list of maps. |
| f <map_list> | Compile one or more specific map(s). Requires the path and name of one or maps (for example,*<filename1>.map <filename2>.map <filename3>.map)* that you want to compile. |

(Continued on next page)

| (Contd)<br>Parameter | Definition |
|---|---|
| l <map_list_file> | Compile the maps listed in a specific file. Requires the path and name of a file containing a list of maps that you want to compile. |
| P <dir> | Compile all maps in specified directory. |
| Q | Do not display compiler messages to screen. |
| v | Display the version of the compiler and write it to the file *xlcntl.err.* |
| w | Write the compiler version number into the file *compvers.tmp*, but do not display on the screen. |

# mentor

## Purpose

Starts the Gentran:Server menu system.

## Rules for use

Use this command if you are unable to start Gentran:Server by double-clicking on the Gentran:Server icon.

## Starting the program

Start **mentor** from:

- The Server icon within the Gentran:Server options on the Windows Start menu
- The Server icon in the Server program group.
- A DOS command window on the Windows computer

**Note**
The **mentor** command will not run in a script or batch file, or if you are running in DOS mode.

## Using the DOS command line

Use the following notation when including this command in a script or batch file:

```
mentor
```

# mtimer

## Purpose

Starts the process of searching for the batch file specified in the Task Scheduler. The program waits until the specified time, then executes the commands within that batch file.

## Rules for use

You *cannot* use this program within a batch file.

## Starting the program

Start **mtimer** from:

- The **Run Task Scheduler** option on the Tools menu within the Main window.
- A DOS command window.

**Note**
For Gentran:Server for UNIX, you must first set up the *crontab.dat* file using the **Schedule Tasks** option on the Tools menu.

## Using the command in a DOS command window

Use the following notation when including this command in a script or batch file:

```
mtimer
```

# newtype

## Purpose

Builds a list of files whose names are unique for the first specified number of bytes.

The unique names may be assembled by using the first c1 bytes only or the entire name of the first occurrence. The default is the whole file name of the first occurrence.

## Files created

This table lists the name and description of the file created when you run **newtype**.

| File name | Description |
|---|---|
| <outfile> | The file containing a list of file names which are unique up to a certain position.<br>By default, this file is created in the current directory. If you specify a full path in the command, the file will be created in the specified location. |

## Starting the program

Start **newtype** from the UNIX host command line.

**Note**

The **newtype** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
newtype {-c1 -c2 <infile> <outfile>}
```

**Note**

Do not include the braces in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **newtype** command.

| Parameter | Definition |
|---|---|
| <infile> | The source file containing the original list. |
| <outfile> | The destination file for the altered list. |
| c1 | Contains the number of bytes to compare. |
| c2 | Contains the option switch -k for key portion of the file name in the outfile. |

# orgcreat

## Purpose

Creates empty Trading Partnership Organization Files.

**WARNING**

**Use with caution! This program overwrites any data in existing Trading Partnership Organization files.**

## Files created

This table lists the names and descriptions of the empty files created when you run **orgcreat**.

| File name | Description |
|---|---|
| org.dat | The Organization File, which stores the Organization records that are maintained by the Trading Partner explorer.<br>This file is created in the trading partner directory. |
| org.idx | The index file that relates to the *org.dat* files.<br>This file is created in the trading partner directory. |
| org.lck | The lock file generated when the command is run from Windows. |
| orgcreat.log | The error file created to report the status of the generation of the Organization Files (*org.dat* and *org.idx*).<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation<br>◗ The *temp* directory for Gentran:Server for UNIX |

## Starting the program

Start **orgcreat** from:

- The **Run Program** option on the Main window Tools menu.
- The **New/Create**=>**Organization File** option from the Tools menu within the Trading Partnership explorer.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
orgcreat [-cp <config path>]
```

## Parameters and variables

This table lists the parameters and variables defined for the **orgcreat** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# orgmv

## Purpose

Loads and unloads Trading Partnership Organization records.

## Files created

This table lists the name and description of the file created when you run **orgmv**.

| File name | Description |
|---|---|
| org.unl | The file created when the Organization records are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the trading partner directory. |
| orgmv.log | The error/log file created by **orgmv**.<br>This file is created in:<br>◗ The host *temp* directory for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start **orgmv** from:

- The **Run Program** option on the Main window Tools menu.
- The **Unload**=>**Organization Records** option from the Tools menu within the Trading Partnership explorer.
- The **Load**=>**Organization Records** option from the Tools menu within the Trading Partnership explorer.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
orgmv -[l/lo/u] [-cp <config path>]
```

**Note**
Do not include the braces or slashes in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **orgmv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file. |
| lo | Load from ASCII and update existing records (overwrite). |
| u | Unload from ISAM to ASCII file. |

# putlog

## Purpose

Reformats any input file into the standard Gentran:Server log file structure.

## When to use

Use the **putlog** utility to convert non-standard error files that you want to retain with date and time information into Gentran:Server log files. For example, you can use it to reformat the *xlcntl.err* file, which reports the status of inbound or outbound translation.

## Files created

This table lists the name and description of the file created when you run **putlog**.

| File name | Description |
|---|---|
| <output> | The reformatted log file.<br>By default, this file is created in the user's working directory. If you specify a full path in the command, the file will be created in the specified location. |

## Starting the program

Start **putlog** from the UNIX host command line.

**Note**

The **putlog** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
putlog {<return_code>} {<program_name>} {<process_id>} [<width>]
{< <infile> > <outfile>}
```

**Note**

Do not include the braces or brackets in the command. Be sure to include the UNIX redirection symbols (< or >) preceding <infile> and <outfile>.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **putlog** command.

| Parameter | Definition |
|---|---|
| <return_code> | An integer that represents the success of the log. |
| <program_name> | The name of the calling program. Maximum: 9 characters. |
| <process_id> | The process identification number. Maximum value: 2 billion. |
| <width> | The length of the message string. This is an integer value greater than 10. The default is 50. |
| **<** <infile> | The name of the input file.<br>**Note**<br>You must include the redirection symbol (<). |
| **>** <outfile> | The name of the output file.<br>**Note**<br>You must include the redirection symbol (>). |

## Example

**putlog** 2 test 90908 <test.err>test.log

## Sample input file

A sample input file follows (test.err):

```
RVR_SRC/valid/searchidx.sh Valid
Searching all IDX/DAT files for Valid
on ncr at /usr/srvr3/qa
by jane on Wed Nov 6 15:28:55 EST 1996
Searching    88 .idx/.dat files
ds_map:
Validation|850|BEG|3||1|20|1|m|||||
Validation|850|GS|6||1|20|21|n|||||
ds_name:
Valid|for Validation Test|EDI||
```

(Continued on next page)

## Sample output file

A sample output file follows (test.log):

```
test:90908: 19970127:170230: 2:RVR_SRC/valid/searchidx.sh Valid :
test:90908: 19970127:170230: 2:Searching all IDX/DAT files for
Valid :
test:90908: 19970127:170230: 2:on ncr at /usr3/srvr3/qa :
test:90908: 19970127:170230: 2: by jane on Wed Nov 6 15:28:55 EST
1996 :
test:90908: 19970127:170230: 2: Searching    88 .idx/.dat files :
test:90908: 19970127:170230: 2:ds_map: :
test:90908: 19970127:170230: 2:Validation|850|BEG|3||1|20|1|m|||||
:
test:90908: 19970127:170230: 2:Validation|850|GS|6||1|20|21|n|||||
:
test:90908: 19970127:170230: 2:ds_name: :
test:90908: 19970127:170230: 2:Valid|for Validation Test|EDI|| :
```

# ret0

## Purpose

The **ret0** program sets the UNIX return code to 0.

## When to use

Use **ret0** within a script to force a TRUE condition for a logical test.

**Reference**
See the ret1 topic in this chapter for suggested usage.

## Starting the program

Include this command in scripts on the UNIX host.

**Note**
The **ret0** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
ret0
```

# ret1

## Purpose

The **ret1** program sets the UNIX return code to 1. Use **ret1** within a script to force a FALSE condition for a logical test.

## How to use

You can use **ret1** with **ret0** to test a script. For example, set **ret1** before a section that you suspect is not working. Set **ret0** just after the suspect command. Use **echo$?** to view the returned value and see whether the suspect command completed successfully.

## Starting the program

Include this command in scripts on the UNIX host.

**Note**
The **ret1** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
ret1
```

# rif

## Purpose

Resets file counters in Gentran:Server archive or data manager ISAM files to a specified number. The file counters are automatically generated incremental numbers that determine the unique ID suffix in a file name. This program is used to reset the file counters when you move data from test to production.

## When to use

Use the **rif** command when you need to reset the counters for the **genuniqid** or **genmbid** programs.

## Starting the program

Start this program from the UNIX host command line.

### Note

The **rif** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
rif {-f <fname>} [-r <reset_val>]
```

### Note

Do not include the braces or brackets in the command.

### Example

```
rif -f arch -r 1
```

This example resets the file counter in the *arch.dat* ISAM file to “1”.

## Parameters and variables

This table lists the parameters and variables defined for the **rif** command.

| Parameter | Definition |
|---|---|
| f <fname> | Name of the ISAM file to reset.<br>Use *.scrcfg* to reset mailbag IDs.<br>Use *ds_name* to reset unique IDs. |
| r <reset_val> | The value with which you want increments to start. The value must be greater than or equal to 1. Default =1. |

# rtv_arc

## Purpose

The **rtv_arc** program is a background retrieval process that extracts selected archived data from an archive file and deposits the data into a directory that you specify.

## Files created

This table lists the name and description of the files created when you run **rtv_arc**.

| File name | Description |
|---|---|
| rtv_arc.l | A log file that contains information on the retrieved archive data and any errors that occurred.<br>This file is created in the user's working directory. |

## Starting the program

Start **rtv_arc** from the UNIX host command line.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
rtv_arc -t <tpcode> -d <specnum> -b <start> -e <end> -a <arcdir> -o
<out_fn> -r -undue <in_fn/agentname>
```

## Parameters and variables

This table lists the parameters and variables defined for the **rtv_arc** command.

| Parameter | Definition |
|---|---|
| <in_fn/dmname> | Input file name or data manager name. This parameter is required. |
| a <arcdir> | Archive file directory. The default directory specified in the data manager's initialization file. |

(Continued on next page)

| (Contd)<br>Parameter | Definition |
|---|---|
| b <start> | Starting date for retrieval. Specify the date in the format CCYYMMDD or YYMMDD. The default value is 970101. |
| d <specnum> | Document specifier number. |
| e <end> | Ending date for retrieval. Specify the date in the format CCYYMMDD or YYMMDD. The default value is the current date. |
| o <out_fn> | Output file name. The default value is stdout. |
| r | Label the entry to be removed from the transaction registry. |
| t <tpcode> | Trading partner code. |
| undue | Cancel the action label preceding the entry. |

# smgr

## Purpose

Executes the statements in a script.

## Files created

This table lists the name and description of the file created when you run **smgr**.

| File name | Description |
|---|---|
| <script>.old | The completed log file. This file is created in the *EDI_ROOT/script* directory. |
| <script>.l | The currently running log file.This file is created in the *EDI_ROOT/script* directory. |

## Starting the program

Start **smgr** from the UNIX host command line.

**Note**
The **smgr** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
smgr [-s<script>] [-d/D] [-e<env_name>=<env_value>]
```

**Note**
Do not include the braces, slashes, or brackets in the command.

## Parameters

This table lists the parameters defined for the **smgr** command.

| Parameter | Definition |
|---|---|
| d | Includes step execution in the script log file. |
| D | Displays step execution on screen and includes it in the script log file. |

(Continued on next page)

| (Contd)<br>Parameter | Definition |
|---|---|
| e<env_name>=<env_value> | Set the specified environment variable to the specified value. |
| s<script> | Identifies the script you want to execute. Do not include the extension (*.scr*). The script must reside in EDI_ROOT/script. |

# srvr_deq

## Purpose

Deletes an entry in a processing queue.

## Starting the program

Start **srvr_deq** from the UNIX host command line.

**Note**
The **srvr_deq** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
srvr_deq {-q<queue dir>} {-j<filename>} {-d<directory>} {-g<group>}
```

**Note**
Do not include the braces in the command.

The **srvr_deq** command must be invoked from *$EDI_ROOT*.

## Parameters and variables

This table lists the parameters and variables defined for the **srvr_deq** command.

| Parameter | Definition |
|---|---|
| d<directory> | Identifies the directory location of the file listed in the queue |
| g<group> | Identifies a resource group **srvr_deq** will use. Specify the name of the resource group as defined in the initialization file of the data manager that scans the queue. |
| j<filename> | Identifies the file you want to remove from a queue. |
| q<queue dir> | Identifies a queue directory (for example, “xltii”). |

# srvr_enq

## Purpose

Adds entries to a processing queue.

## Starting the program

Start **srvr_enq** from the UNIX host command line.

**Note**
The **srvr_enq** command is available only for the Gentran:Server Process Control Manager product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
srvr_enq {-q<queue dir> -g<group> -j<filename> -d<file dir>}
[-p<priority>] [-n<max number>]
```

**Notes**

Do not include the braces or brackets in the command.

The **srvr_enq** command must be invoked from $EDI_ROOT. The RESOURCE_GROUP and WORK_TYPE initialization parameters must be set in the initialization file of the data manager that will be reading from this queue.

## Parameters and variables

This table lists the parameters and variables defined for the **srvr_enq** command.

| Parameter | Definition |
|---|---|
| d<file dir> | Specifies the path to the file you want to add to the queue |
| g<group> | Identifies a resource group **srvr_enq** will use. Specifies the name of the resource group as defined in the initialization file of the data manager that scans the queue. |
| j<filename> | Identifies the file you want to add to a queue. |
| n<max number> | Identifies the maximum number of entries permitted for this resource group. |

(Continued on next page)

| (Contd)<br>Parameter | Definition |
|---|---|
| q<queue dir> | Identifies a queue directory (for example, “xlti”). |
| p<priority> | Identifies a destination queue processing priority for the file (0 - 9, with 9 being lowest priority). The default value is 9. |

# startnb.sh

## Purpose

Starts the namebroker process (**ltb_server**).

## Starting the program

Start **startnb.sh** from the UNIX host command line at the *$SADMIN_ROOT/broker* directory.

**Note**
The **startnb.sh** command is available for all levels of Gentran:Server for UNIX. It is not available for Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
startnb.sh
```

# startrpc.sh

## Purpose

Use **startrpc.sh** to start the **mhs_server** and **mhp_server** programs for specific environments. The **startrpc.sh** script prompts for environment variable values, sets other necessary environment variables, and starts the **mhs_server** and **mhp_server** programs.

## Starting the script

Start **startrpc.sh** from the UNIX host command line.

**Note**
The **startrpc.sh** script is available only for the Gentran:Server for UNIX product level and higher. It is *not* available with Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
startrpc.sh
```

## Environment variables

The **startrpc.sh** script prompts for the paths to the following environment variables.

| Parameter | Definition |
|---|---|
| EDI_ROOT | The location where the Gentran:Server EDI software is installed. |
| SADMIN_ROOT | The location where the Gentran:Server Security Administration software is installed. |

# startserver

## Purpose

Brings up a single data manager; or the Foreground Manager, **fmgr**, which then starts all data managers marked to autostart.

## When to use

**startserver** needs to be run whenever the UNIX host is restarted. To run **startserver** on startup, have your UNIX system administrator put the command in the *inittab* or *rc* startup file, or see the startserver.sh script command.

## Starting the program

Start **startserver** from within a script or batch file.

**Note**
The **startserver** command is available only for the Gentran:Server for UNIX with Process Control Manager product level and higher.

## Using startserver in a script, batch file, or at the command line

Use the following notation when including this command in a script or batch file:

```
startserver [<dmname>]
```

For command line usage, refer to the startserver.sh script.

## Parameter and variable

This table lists the parameter and variable defined for the **startserver** command.

| Parameter | Definition |
|---|---|
| | If no parameter is specified, then **startserver** brings up **fmgr**, which then starts all agents marked to autostart. |
| <dmname> | The name of a data manager to bring up in addition to **fmgr**.<br>**Note**<br>When this parameter is used, if **fmgr** is not already running all agents marked to autostart will start. |

# startserver.sh

## Purpose

Use the **startserver.sh** script to start Gentran:Server background processes. At the Gentran:Server for UNIX with Process Control Manager product level and higher, **startserver.sh** uses the **startserver** command to bring up a single agent; or the Foreground Manager, **fmgr**, which then starts all data managers marked to autostart. In addition to running **startserver**, **startserver.sh** also runs the recovery script **recover.scr**. Optionally, **startserver.sh** -a can be used in a non-interactive manner to start all Gentran:Server background precesses, including the PRC servers and the database server for the optional SAP extension.

## When to use

**startserver.sh** or **startserver** needs to be run whenever the UNIX host is restarted. To run **startserver.sh** on startup, have your UNIX system administrator put the command in the *inittab* or *rc* startup file.

## Starting the script

Start **startserver.sh** from the UNIX host command line.

**Note**
The **startserver.sh** script is available for all Gentran:Server for UNIX product levels. It is *not* available with Gentran:Server Workstation.

## Using startserver.sh in a script or batch file

Use the following notation when including this command in a script or batch file:

```
startserver.sh [-a <varfile>] [-d <dmname>] [EDI_ROOT]
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **startserver.sh** script.

| Parameter | Definition |
|---|---|
| | At the Gentran:Server for UNIX with Process Control Manager product level and higher, if no parameter is specified, then **startserver.sh** brings up **fmgr**, which then starts all agents marked to autostart. |
| a <varfile> | Start all necessary local Gentran:Server background processes for a specific environment. The <varfile> portion of this parameter is a fully qualified path to a file that defines certain environment variables. |
| d <dmname> | The name of a data manager to bring up in addition to **fmgr**.<br>**Note**<br>This parameter is only available for the Gentran:Server for UNIX with Process Control Manager product level and higher. When this parameter is used, if **fmgr** is not already running all agents marked to autostart will start. |
| EDI_ROOT | The location where the Gentran:Server EDI software is installed. |

The file <varfile> specified in the **startserver.sh** -a parameter must contain the parameters listed in the following table.

| Parameter | Definition |
|---|---|
| NAMEBROKER | The source of the Gentran:Server broker process. |
| EDI_ROOT | The location where the Gentran:Server EDI software is installed. |
| SADMIN_ROOT | The location where the Gentran:Server Security Administration software is installed. |
| NBNOTLOCAL | Optional parameter to prevent startserver.sh from attempting to start **ltb_server**. The default behavior is to always start **ltb_server** locally.<br>**Note**<br>This option is not used by **stopserver.sh** |

(Continued on next page)

**Note**
**startserver.sh** used with the -a parameter can start the following Gentran:Server background processes:

- **mhp_server**, **mhs_server**, and **ltb_server** (all product levels)
- **fmgr** and all data managers, if applicable (with PCM or higher only)
- **mysqld**, if applicable (requires SAP extension)

## Example

```
startserver.sh -a $EDI_ROOT/envi.vars
```

A sample <varfile> file, in this case $EDI_ROOT/envi.vars, follows:

```
NAMEBROKER=hpk360
EDI_ROOT=l3/srvr/61/610/server
SADMIN_ROOT=l3/srvr/61/security
NBNOTLOCAL=true
```

The file specified by <varfile> can exist anywhere and can have any name. $EDI_ROOT and envi.vars are just examples.

# stoprpcs.sh

## Purpose

Use the **stoprpcs.sh** script to stop **mhs_server** and **mhp_server** programs for specific environments. The **stoprpcs.sh script** prompts for environment variable values, displays a list of the environments that are running, and shuts down the specified environments.

## Starting the script

Start **stoprpcs.sh** from the UNIX host command line.

**Note**
The **stoprpcs.sh** script is available only for the Gentran:Server for UNIX product level and higher. It is *not* available with Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
stoprpcs.sh
```

## Environment variables

The **stoprpcs.sh** script prompts for the paths to the following environment variables.

| Parameter | Definition |
|---|---|
| EDI_ROOT | The location where the Gentran:Server EDI software is installed. |
| SADMIN_ROOT | The location where the Gentran:Server Security Administration software is installed. |

# stopserver

## Purpose

Stops all active data managers, including the Foreground Manager, **fmgr**.

## When to use

You should use **stopserver** when you need to stop active data managers and then **fmgr**. The **stopserver** command allows each data manager to finish processing any files it is currently processing before it is shut down.

## Starting the program

Start this program from within a script or batch file.

**Note**
The **stopserver** command is available only for the Gentran:Server for UNIX with Process Control Manager product level and higher.

## Using the command in a script, batch file, or at the command line

Use the following notation when including this command in a script or batch file:

```
stopserver [<dmname>]
```

For command line usage, refer to the stopserver.sh script.

## Parameter and variable

This table lists the parameter and variable defined for the **stopserver** command.

| Parameter | Definition |
|---|---|
| | If no parameter is specified, then **stopserver** stops all of the data managers that are running, and stops the foreground manager, **fmgr**. |
| <dmname> | The name of a single data manager to stop. |

# stopserver.sh

## Purpose

Use the **stopserver.sh** script to stop Gentran:Server background processes. At the Gentran:Server for UNIX with Process Control Manager product level and higher, **stopserver.sh** uses the **stopserver** command to stop all active data managers, including the Foreground Manager, **fmgr**. In addition to running **stopserver**, **stopserver.sh** will kill **fmgr** and its child processes if they are not responding. Optionally, **stopserver.sh** with the -a parameter can be used in a non-interactive manner to stop all Gentran:Server background processes, including the PRC servers and the database server for the optional SAP extension.

## When to use

You should use **stopserver.sh** when you need to stop background processes, active data managers, and **fmgr**. The **stopserver.sh** script allows each data manager to finish processing any files it is currently processing before it is shut down.

## Starting the script

Start this script from the host command line.

**Note**
The **stopserver.sh** script is available for all Gentran:Server for UNIX product levels. It is *not* available with Gentran:Server Workstation.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
stopserver.sh [-a <varfile>] [-d <dmname>] [-i] [-q]
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **stopserver.sh** script.

| Parameter | Definition |
|---|---|
| | At the Gentran:Server for UNIX with Process Control Manager product level and higher, if no parameter is specified, then **stopserver.sh** stops all of the data managers that are running, and stops the foreground manager, **fmgr**. |
| a <varfile> | Stops all local Gentran:Server background processes for a specific environment. The <varfile> portion of this parameter is a fully qualified path to a file that defines certain environment variables.<br>**Note**<br>The **stopserver.sh** script can only be run with the -a parameter if startserver.sh has previously been run with the -a parameter.<br>The **stopserver.sh** script must read a file named *startserver.pids* to obtain the process IDs of the processes started by **startserver.sh**. This file is only created when **startserver.sh** is run with the -a parameter. |
| d <dmname> | The name of a single data manager to stop.<br>**Note**<br>This parameter is only available for the Gentran:Server for UNIX with Process Control Manager product level and higher. |
| i | Turns off interactive mode. |
| q | Same as -i parameter. |

The file <varfile> specified in the **stopserver.sh** -a parameter must contain the parameters listed in the following table.

| Parameter | Definition |
|---|---|
| NAMEBROKER | The source of the Gentran:Server broker process. |
| EDI_ROOT | The location where the Gentran:Server EDI software is installed. |
| SADMIN_ROOT | The location where the Gentran:Server Security Administration software is installed. |

**Note**

**stopserver.sh** used with the -a parameter can stop the following Gentran:Server background processes:

- **mhp_server**, **mhs_server**, and **ltb_server** (all product levels)
- **fmgr** and all data managers, if applicable (with PCM or higher only)
- **mysqld**, if applicable (requires SAP extension)

## Example

```
stopserver.sh -a $EDI_ROOT/envi.vars
```

A sample <varfile> file, in this case $EDI_ROOT/envi.vars, follows:

```
NAMEBROKER=hpk360
EDI_ROOT=l3/srvr/61/610/server
SADMIN_ROOT=l3/srvr/61/security
```

The file specified by <varfile> can exist anywhere and can have any name. $EDI_ROOT and envi.vars are just examples.

# synmv

## Purpose

Loads and unloads generic or specific map synonym lists. Used to convert ISAM files to ASCII text so that they can be moved to a machine with a different hardware platform or operating system and converted back to ISAM files. This command is used only with the Visual Mapper.

## Files created

This table lists the names and descriptions of the files created when you run **synmv**.

| File name | Description |
|---|---|
| <mapname>.unl | The file created when the Specific Synonym Files (*.dat* and *.idx*) for a map are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the *maps* directory.The default file is *syntemp.unl* in the temporary directory. |
| generic.unl | The file created when the Generic Synonym Files are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the maps directory. |
| synmv.log | The message file that records the results of the **synmv** operation.<br>This file is created in the *rpt* directory. |
| thesaurs.unl | The file created when the Thesaurus Files are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the maps directory. |

## Starting the program

Start **synmv** from:

- The **Unload File** or **Load File** option on the File menu within either the Synonym Editor or the Thesaurus Editor.
- The DOS command line, if you are running Gentran:Server Workstation.
- The UNIX command line, if you are running Gentran:Server for UNIX.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
synmv <path/filename> -[l/lo/d/u] [-cp <config path>]
```

**Note**
Do not include the braces, slashes, or brackets in the command.

## Rules

Use the following rules when entering the **synmv** command.

- When using a UNIX operating system, use forward slashes (/) in the path.
- When using a DOS operating system, use backslashes (\) in the path.

## Parameters and variables

This table lists the parameters and variables defined for the **synmv** command.

| Parameter | Definition |
|---|---|
| <path/filename> | The location and name of the Synonym Files (with no extension) to process. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| d | Delete the records defined in the ASCII file from the ISAM file. |
| l | Convert the ASCII .unl file back to ISAM files and load to the new machine. Do not overwrite duplicate records. |
| lo | Convert the ASCII .unl file to back to ISAM files and load to the new machine. Overwrite duplicate records. |
| u | Unload the Synonym files, converting from ISAM format to ASCII, creating a .unl file. |

## Example 1

To unload the generic synonym files *generic.dat/idx* and create the *generic .unl* file in the maps directory, type this command at the $EDI_ROOT directory:

```
synmv <map_directory>/generic -u -cp <config path>
```

(Continued on next page)

### Example 2

To load the *generic.unl* file, converting it back to an ISAM format and overwriting the duplicate generic synonym records in the maps directory, type this command at the $EDI_ROOT directory:

```
synmv <map_directory>/generic -lo -cp <config path>
```

### Example 3

To load the *generic.unl* file, converting it back to an ISAM format and appending to the existing generic synonym files in the maps directory without overwriting duplicate records, type this command at the $EDI_ROOT directory:

```
synmv <map_directory>/generic -l -cp <config path>
```

# tccreate

## Purpose

Creates new empty indexed TRADACOMS supplementary Trading Partnership Files (*tradacom.dat* and *tradacom.idx*).

**WARNING**

**Use with caution! This program overwrites any data in existing TRADACOMS supplementary Trading Partnership Files.**

## Files created

This table lists the names and descriptions of the files created when you run **tccreate**.

| File name | Description |
|---|---|
| tradacom.dat | The TRADACOMS Supplementary Trading Partnership File, which stores the supplementary Trading Partnership data necessary for partnerships using TRADACOMS standards.<br>This file is created in the trading partner directory. |
| tradacom.idx | The index file that relates to the *tradacom.dat* file.<br>This file is created in the trading partner directory. |
| tccreate.log | The error file created to report the status of the generation of the TRADACOMS Supplementary Trading Partnership Files (*tradacom.dat* and *tradacom.idx*).<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation<br>◗ The *temp* directory for Gentran:Server for UNIX |

(Continued on next page)

## Starting the program

Start **tccreate** from:

- The **Run Program** option on the Main window Tools menu.
- The **New/Create=>Trading Partner** option on the Trading Partner Administration Tools menu. This method creates all of the trading partner files (*tp.*, tpmisc.*, tprecon.*, tradacom.**)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tccreate [-cp <config path>]
```

## Parameters and variables

This table lists the parameters and variables defined for the **tccreate** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# tcmv

## Purpose

Loads and unloads TRADACOMS supplementary Trading Partnership records.

## Files created

This table lists the name and description of the file created when you run **tcmv**.

| File name | Description |
|---|---|
| tradacom.unl | The file created when the TRADACOMS Supplementary Trading Partnership records are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the trading partner directory. |
| tcmv.log | The error file created to report the status of the move operation.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation<br>◗ The *temp* directory for Gentran:Server for UNIX |

## Starting the program

Start **tcmv** from:

- The **Unload=>Trading Partnership Records** option on the Tool menu in Trading Partner Administration.
- The **Load=>Trading Partnership Records** option on the Tool menu in Trading Partner Administration.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tcmv -[l/lo/u] -[cp <config path>]
```

### Note

Do not include the brackets or slashes in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **tcmv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file. |
| lo | Load from ASCII file and update existing ISAM file (overwrite). |
| u | Unload from ISAM to ASCII file. |

# tpcreate

## Purpose

Creates empty indexed Trading Partnership files and Reconciliation ID files.

**WARNING**

**Use with caution! This program overwrites your existing Trading Partnership and Reconciliation ID files.**

## Files created

This table lists the name and description of the file created when you run **tpcreate.** These files are created in the trading partner directory.

| File name | Description |
|---|---|
| tp.dat | The Trading Partnership File, which stores the Trading Partnership records that are maintained by the Trading Partner Maintenance program (tpfm).<br>This file is created in the trading partner directory. |
| tp.idx | The index file that relates to the *tp.dat* file.<br>This file is created in the trading partner directory. |
| tpcreate.log | The error file created to report the status of the generation of the Trading Partnership Files (*tp.dat* and *tp.idx*).<br>This file is created in the *temp* directory. |
| tpmisc.dat | The Miscellaneous file which stores the miscellaneous records that are maintained by the Trading Partner Maintenance program.<br>This file is created in the trading partner directory. |
| tpmisc.idx | The index file that relates to the *tpmisc.dat* file.<br>This file is created in the trading partner directory. |
| tprecon.dat | The Reconciliation ID File, which stores the Reconciliation ID records that are maintained by the trading partner maintenance program.<br>This file is created in the trading partner directory. |
| tprecon.idx | The index file that relates to the *tprecon.dat file*.<br>This file is created in the trading partner directory. |

## Starting the program

Start **tpcreate** from:

- The **New/Create**=>**Trading Partnership File** option on the Tools menu within the Trading Partnership explorer.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tpcreate [-cp <config path>]
```

## Parameters and variables

This table lists the parameters and variables defined for the **tpcreate** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# tp_info

## Purpose

Use the **tp_info** command to display selected information from the Trading Partnership database. The selected information is sent to standard out, which is typically your display terminal.

## Starting the program

Start **tp_info** from:

- The UNIX host command line if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tp_info <outputcode> -<search_code> <search_value>
{-<search_code> <srch_value> …} [-cp <config path>]
```

## Parameters and variables

This table lists the parameters and variables defined for the **tp_info** command.

| Parameter | Definition |
|---|---|
| <outputcode> | Display the Trading Partnership fields specified by the output code. |
| <search_code> | Specifies the fields to examine within the Trading Partnership Records. |
| <search_value> | Select records containing the specified search value within the field specified by the search code. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

(Continued on next page)

## Output codes

This table lists the output codes defined for the **tp_info** command.

| Output Code | Definition |
|---|---|
| c1 | Trading Partnership category number 1 |
| c2 | Trading Partnership category number 2 |
| c3 | Trading Partnership category number 3 |
| c4 | Trading Partnership category number 4 |
| c5 | Trading Partnership category number 5 |
| d | Transaction set and message ID |
| n | Trading Partnership name and description |
| pi | Trading Partner’s Interchange ID |
| pg | Trading Partner’s Group ID |
| t | Trading Partner Code |
| ti | Trading Partner Code and the Interchange information |
| tg | Trading Partner Code and the Group information |
| ui | User’s Interchange ID |
| ug | User’s Group ID |
| v | Standard version ID |

## Search codes

This table lists the search codes defined for the **tp_info** command.

| Search Code | Definition |
|---|---|
| a | All Trading Partner Codes (overrides -t option) |
| c1 | Trading Partnership category number 1 |
| c2 | Trading Partnership category number 2 |
| c3 | Trading Partnership category number 3 |

(Continued on next page)

| (Contd)<br>Search Code | Definition |
|---|---|
| c4 | Trading Partnership category number 4 |
| c5 | Trading Partnership category number 5 |
| d | Transaction set and message ID |
| n | Trading Partnership name and description |
| pi | Trading Partner’s Interchange ID |
| pg | Trading Partner’s Group ID |
| t | Trading Partner Code |
| ui | User’s Interchange ID |
| ug | User’s Group ID |
| v | Standard version ID |

# tpmv

## Purpose

Moves Trading Partnership records from one platform to another (DOS to UNIX, UNIX to DOS, or one UNIX platform to another). Loads/unloads Reconciliation ID records.

## Files created

This table lists the names and descriptions of the files created when you run **tpmv.** These files are created in the trading partner directory.

| File name | Description |
|---|---|
| tp.unl | The file created when the Trading Partnership records are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the trading partner directory. |
| tprecon.unl | The file created when the Reconciliation ID records are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the trading partner directory. |
| tpmisc.unl | The supplementary file created when the Trading Partnership records are unloaded into ASCII format for moving from platform to platform.<br>This file is created in the trading partner directory. |
| tpmv.log | The error file created to report the status of the move operation.<br>This file is created in:<br>◗ The *rpt* directory for Gentran:Server Workstation<br>◗ The *temp* directory for Gentran:Server for UNIX |

(Continued on next page)

## Starting the program

Start **tpmv** from:

- **Unload**=> **Trading Partnership Records** from the Tools menu within the Trading Partnership explorer.
- **Load** => **Trading Partnership Records** from the Tools menu within the Trading Partnership explorer.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tpmv -[l/lo/u] -[cp <config path>]
```

**Note**
Do not include the slashes or brackets in the command.

## Parameters

This table lists the parameters defined for the **tpmv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| F | Assign a filename root ending in "tp". |
| l | Load from ASCII to ISAM file. |
| lo | Load from ASCII and update existing records. |
| u | Unload from ISAM to ASCII file. |

# tpops

## Purpose

Displays or deletes records with a specified trading partner code or trading partner code string from up to four ISAM files (*tp*, *dm*, *ds_tptbl*, and *ds_tptbl_app*) simultaneously. Includes an option to back up the operation.

## Starting the program

Start this program from the command line. For Gentran:Server product installations, run this on the host.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tpops -[b/d/l/i/v/w] [-t <tp code> <isam_file1> {isam_file2}]
```

**Note**
Do not include the braces, slashes, or brackets in the command.

### Output

If you use the backup option, the output is in a format that you can reload into your trading partner database with the **isops** or **tpmv** program.

### Example 1

```
tpops -lv -t OUTB2856 /srvr/30/12/server/tp /srvr/30/13/server/tp /
srvr/30/13/server/dm /srvr/30/13/server/ds_tptbl
```

Displays all the records with trading partner codes equal to OUTB2856 in the four ISAM files specified.

### Example 2

```
tpops -lwv -t OUT tp ds_tptbl dm
```

Displays all the records with trading partner codes that contain the string OUT in the three specified ISAM files.

### Example 3

```
tpops -dvi -t OUTB2856 tp ds_tptbl dm
```

Deletes records with trading partner codes of OUTB2856 from the three specified ISAM files (*tp*, *ds_tptbl*, and *dm*). The i option prints each record to the screen, presents a confirmation prompt, and requires a response to continue.

### Example 4

```
tpops -dw -t OUT tp ds_tptbl dm
```

Deletes records with trading partner codes that contain the string OUT in the three specified ISAM files.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **tpops** command.

| Parameter | Definition |
|---|---|
| <isam_file> | The full path and file name of an ISAM file, or the ISAM file name if the file is in the current directory. |
| b | Back up the operation. Print removed records to the standard output in a format that enables the **isops** or **tpmv** commands to reload the records into the corresponding ISAM files.<br>The default setting is Off. |
| d | Display or delete records that contain the specified trading partner code from the listed ISAM files.<br>The default setting is Off.<br>**CAUTION**<br>**If you use this option, do not use the -l option.** |
| i | Prompt for confirmation before deleting a record.<br>The prompt is:<br>Delete (y/n)? ><br>The default setting is Off. |
| l | List records that match the specified trading partner code.<br>The default setting is On. |
| t <tp code> | The Trading Partner code. |
| v | Turn on the verbose mode. Print removed records to the standard output in a format that allows the **isops** or **tpmv** commands to reload the records into the corresponding ISAM files.<br>The default setting is Off. |
| w | Display or delete records that have the specified string in the trading partner code.<br>The default setting is off.<br>Because the search is performed record by record, this is a slower process than using the -d option.<br>**Note**<br>This option should be used with either the -d or -l option. |

# tracker

## Purpose

When the Life Cycle feature is functional, the tracker program interacts with supported databases and generates statistical reports about translation traffic for a specified date. The tracker program:

- Accumulates the number of good and bad sets for inbound and outbound translation and lists the numbers by Trading Partner Code, the data manager name, or both, as requested.
- Accumulates the total number of segments and total number of characters processed for the listed Trading Partner Code and/or data manager name.
- Lists the file names (if any) that are in error.

The program retrieves the information from the Oracle, Sybase, or Informix database table lc221 and organizes it according to the reports requested.

**Note**: Support for Oracle databases does not include support for the Oracle Exadata platform.

### Comment

You must build this executable before you can use it. See the Gentran:Server *for UNIX Data Flow Administration Guide* for instructions.

## Files created

This table lists the name and description of the file created when you run **tracker**.

| File name | Description |
|---|---|
| <date>.rpt | The output report file. This file is created in the user’s working directory. |

## Starting the program

Start this program from the UNIX host command line.

### Note

The **tracker** command is available only for the Gentran:Server Process Control Manager product level and higher.

(Continued on next page)

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tracker {-d <date>} {-u <username>} {-p <password>} [-t <tp code>]
[-a <dmname>] [-b] [-l <printer_name>] [-v]
```

**Note**
Do not include the braces or brackets in the command.

## Parameters and variables

This table lists the parameters and variables defined for the **tracker** command.

| Parameter | Definition |
|---|---|
| a <Dmname> | Use the specified data manager name. |
| b | Generate the report by Trading Partnership Code and data manager name. |
| d <date> | Use the specified date of translation.<br>Enter the date in the format DD-MMM-YYYY. For example, you would enter June 12, 1999 as: 12-jun-1999 |
| l <printer_name> | Print the report to the listed printer. |
| u <username> | Use the specified database user’s login name. |
| p <password> | Use the specified database user’s password. |
| t <tp code> | Use the specified Trading Partner code. |
| v | Display the report on screen. |

(Continued on next page)

### Sample report

```
Traffic Report for 14-dec-96 (By Trading Partnership Code)

 TP Code          Good_in    Good_Out     Bad_In     Bad_Out
 INBND210         22         0            0          0
 INBND837         4          0            0          0
 INBND850         22         0            0          0
 OUTBND02856      0          6            0          0
 OUTBND03856      0          6            0          0
 TDCC204-1        0          66           0          0
 TDCC204-2        0          66           0          0


 Total Traffic:   198
 Total Inbound:   48     Good:    48          Bad:    0
 Total Outbound:  150    Good:    150         Bad:    0
 Total Other      0

 Total Segments:  1422
 Inbound:         1302
 Outbound:        120

 Total Characters: 201170
 Inbound:         158594
 Outbound:        42576

 Listing of files in error:
```

# transrpt

## Purpose

The **transrpt** program runs a Translation Summary Report.

**Note**

If you want to maintain the *transrpt.r_t* file in the Gentran:Server temporary directory or to route the *transrpt.r_t* file somewhere other than to the default printer, alter the lp_mentr script or batch file.

## Files created

This table lists the name and description of the file created when you run **transrpt**.

| File name | Description |
|---|---|
| transrpt.r_t | The Translation Summary Report File, which stores the Translation Summary Report.<br>This file is created in:<br>◗ The *temp* directory for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

To start this program, select **Translation Summary** from the Translate menu on the Main window.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
transrpt i/o [-T <tpcode>] [-M <set_id>] [-S <st_dt>] [-E <end_dt>]
[-L <lines>] [-P] [-cp <config path>]
```

**Note**

Do not include the slashes, brackets, or braces in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **transrpt** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| E <end_dt> | End the report as of a specific document date. The ending document date for the report. Use one of the following formats:<br>◗ CCYYMMDD<br>◗ null (last date in file). |
| i | Inbound data. |
| L <lines> | Indicates a number of lines per page. |
| M <set_id> | Indicates a specific Transaction Set ID. |
| o | Outbound data. |
| P | Queue the report to the printer. |
| S <stdt> | Start the report as of a specific document date. The starting document date for the report. Use one of the following formats:<br>◗ CCYYMMDD<br>◗ today (current system date)<br>◗ null (earliest date). |
| T <tpcode> | Indicates a specific Trading Partnership Code. |

# tvatrpt

## Purpose

Creates a VAT Report for interchanges containing TRADACOMS invoices. The name of the file containing the VAT Report is *tvatrpt.r_t*, and it resides in the directory for Gentran:Server temporary files.

### Note

If you want to maintain the *tvatrpt.r_t* file in the Gentran:Server temporary directory or to route the *tvatrpt.r_t* file somewhere other than to the default printer, alter the lp_mentr script or batch file.

## File created

This table lists the name and description of the file created when you run **tvatrpt**.

| File name | Description |
|---|---|
| tvatrpt.r_t | The TRADACOMS VAT Report File, which stores the VAT Report for TRADACOMS invoices. This report is created from data in the *tvatrpt.in* and/or *tvatrpt.out* files stored in the data directory.<br>This file is created in:<br>◗ The *temp* directory for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start this program from the command line. For client/server installations, run this on the host. For workstation installations, open a command prompt window and change directories to your server installation directory.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
tvatrpt -i/o [-s <st_dt>] [-e <end_dt>] [-l <loc_ID>] [-g <gen#>]
[-f <filename>] [-cp <config path>]
```

### Note

Do not include the slashes or braces in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **tvatrpt** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |
| e <end_dt> | Report TRADACOMS invoices up to a specific end date. Use one of the following formats:<br>◗ CCYYMMDD<br>◗ null (current date). |
| f <filename> | Report TRADACOMS invoices from a specific archive file. |
| g <gen#> | Report TRADACOMS invoices with a specific file generation number. |
| i | Report all incoming invoices. |
| l <loc_ID> | Report TRADACOMS invoices from a specific sender or receiver location ID. |
| o | Report all outgoing invoices. |
| s <st_dt> | Report TRADACOMS invoices starting at a specific date. Accepts a specific start date. Use the format CCYYMMDD. |

# udf4dnld

## Purpose

Modifies UDF file so that the dnld data manager will accept it. Removes all null characters from a user-defined file and places a modifier in tp position 15 in the user-defined data. The options for specifying the modifier's location are: -r for the record number where the modifier is located and -p for the character position of the modifier. The position begins at the first position of the record.

## Files created

This table lists the name and description of the file created when you run **udf4dnld.**

| File name | Description |
|---|---|
| <outfile> | The output report file.This file is created in the working directory. |

## Starting the program

Start **udf4dnld** from the UNIX host command line.

**Note**
The **udf4dnld** command is available only for the Gentran:Server EC Workbench product level and higher.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
udf4dnld {-r<rec_num>} {-p<mod_position>} {-i<infile>}
{-o<outfile>}
```

**Note**
Do not include the braces in the command.

**Example**

```
udf4dnld -r01 -p10 -iinfile -ooutfile
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **udf4dnld** command.

| Parameter | Definition |
|---|---|
| i<infile> | Defines the read-only input file. File is not deleted. |
| o<outfile> | Defines the output file to open for append. |
| p<mod_position> | Defines the position in the record where the modifier is to be found. Requires a position value from 1 to 1024. |
| r<rec_num> | Defines the record number that is the beginning record for a set. A value from 00 to 99. The default value is 01. |

# unu_maps

## Purpose

Reports the usage status of the maps within an environment. The report lists which maps have been compiled, which have no associated .TPL files, which are referenced by a trading partner record, and which are not used by any trading partners.

## Files created

This table lists the name and description of the file created when you run **unu_maps**.

| File name | Description |
|---|---|
| unu_maps.log | The output report file.<br>This file is created in:<br>◗ The *report/log* directory for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start **unu_maps** from:

- The **Run Program** option on the Main window Tools menu.
- The UNIX host command line at the EDI_ROOT directory if you are running Gentran:Server for UNIX.
- The DOS command line if you are running Gentran:Server Workstation

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
unu_maps
```

(Continued on next page)

## Sample report

```
MAP STATUS REPORT

Current environment is /hpbox/edi

************************************************************
     MAPS WHICH ARE NOT REFERENCED BY TP RECORDS
************************************************************
P3ORD93I           (Map is not compiled)
S4810I                   (Map is not compiled)
S4810O                   (Map is not compiled)
S4850I                   (Map is not compiled)

************************************************************
     MAPS WHICH ARE REFERENCED BY TP RECORDS BUT ARE NOT
                  IN THE MAPS DIRECTORY
************************************************************
Map Name TP Code
******** *******
4097              4097out
1911INB              INBND1911
2879              eta
************************************************************
MAPS WHICH ARE REFERENCED BY TP RECORDS BUT ARE NOT COMPILED
************************************************************
Map Name TP Code
******** *******

************************************************************
     MAPS WHICH ARE REFERENCED BY TP RECORDS
************************************************************
Map Name TP Code
******** *******
4010outb 4010
4010fa         4010fa
4097         4097out
210inb         INBND210
837inb         INBND837
850inb         INBND850
856outb         OUTBND02856
856outb         OUTBND03856
```

# xcreat

## Purpose

Creates new empty indexed Standard Cross Reference Table files (*xtable.dat* and *xtable.idx*).

**WARNING**

**Use with caution! This program overwrites any data in existing Standard Cross Reference Table files.**

## Files created

This table lists the names and descriptions of the files created when you run **xcreat**.

| File name | Description |
|---|---|
| xtable.dat | The Standard Cross Reference Table file, which stores the cross references between a value for a standard that the system extracts from inbound documents and the actual standard that you want the system to use to identify the Trading Partnership record.<br>This file is created in the trading partner directory. |
| xtable.idx | The index file that relates to the *xtable.dat* file.<br>This file is created in the trading partner directory. |

## Starting from the command line

Use the following notation at the command line:

```
xcreat [-cp <config path>]
```

## Parameters and variables

This table lists the parameters and variables defined for the **xcreat** command.

| Parameter | Definition |
|---|---|
| -cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# xlld

## Purpose

The **xlld** script updates the Life Cycle tables *lc221* and *lcdestinfo*.

## Files created

This table lists the names and descriptions of the files created when you run **xlld.**

| File name | Description |
|---|---|
| <eventfile>.l | The log file that contains the results of successful **xlld** transactions.<br>This file is created in the directory of the file name entered with the “f” parameter. |
| <eventfile>.YYMMDDhhmmss | The error file that contains the results of failed **xlld** transactions.<br>This file is created in the directory of the file name entered with the “f” parameter. |

## Starting the program

Start **xlld** from the UNIX host command line.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
xlld -f<filename> -u[<userID>] -p[<password>] -e -o<ODBCevent> -
i<ODBCini> -s<ODBCdsn> -a<ODBCalias>
```

**Note**
Do not include the slashes, brackets, or braces in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **xlld** command.

| Parameter | Definition |
|---|---|
| a<ODBCalias> | Defines the meaning of the entries in the ODBC initialization file. |
| e | Save to the error file only the records that are in error. The default behavior is to save the entire event file. |
| f<filename> | The xltr/lftran event file name, specified without the *.v* extension. |
| i<ODBCini> | Use the specified ODBC initialization file (*odbc.ini*) to determine:<br>◗ The database name<br>◗ The server name<br>◗ The user login ID<br>◗ The database description |
| o<ODBCevent> | The ODBC event file name, specified without the *.v* extension. |
| p[password] | Access the database using the specified database account password. |
| s<ODBCdsn> | The ODBC data source name. |
| u<userID> | Access the database using the specified database account user ID. |

# xmlspl<n>creat

## Purpose

Creates an empty splitting element table for the level specified (xmlspl1, xmlspl2, or xmlspl3). The value <*n*> represents the level of element and must be 1, 2, or 3.

**Note**
You must have the XML translation option in the Map Editor to use this command. This command is not available for use with the Visual Mapper.

**CAUTION**

**Use with caution! This program overwrites any data in the existing xmlspl table for the level specified.**

## Files created

This table lists the names and descriptions of the files created when you run **xmlspl<n>creat**.

| File name | Description |
|---|---|
| xmlspl<n>.dat | The table that stores the XML splitting elements.<br>This file is created in the trading partner directory. |
| xmlspl<n>.idx | The index file that relates to the *xmlspl<n>.dat* file.<br>This file is created in the trading partner directory. |

## Starting the program

Start **xmlspl<n>creat** from the UNIX host command line for all Gentran:Server for UNIX product levels.

Start **xmlspl<n>creat** from the DOS command line for the Workstation product.

## Using the command in a script or batch file

‘Use the following notation when including this command in a script or batch file:

```
xmlspl<n>creat -cp <config path>
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **xmlspl<n>creat** command.

| Parameter | Definition |
|---|---|
| <n> | The level of the element. This value must be 1, 2, or 3. |
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# xmltptblcreat

## Purpose

Creates an empty **xmltptbl** table. This table stores rules that Gentran:Server uses to identify a trading partner in an XML document.

**Note**
You must have the XML translation option in the Map Editor to use this command. This command is not available for use with the Visual Mapper.

**CAUTION**

**Use with caution! This program overwrites any data in the existing xmltptbl table.**

## Files created

This table lists the names and descriptions of the files created when you run **xmltptbl**.

| File name | Description |
|---|---|
| xmltptbl.dat | The table that stores the XML Trading Partnership (Trading Partnership) rules.<br>This file is created in the trading partner directory. |
| xmltptbl.idx | The index file that relates to the *xmltptblcreat.dat* file. This file is created in the trading partner directory. |

## Starting the program

Start **xmltptblcreat** from the UNIX host command line in all Gentran:Server for UNIX product levels.

Start **xmltptblcreat** from the DOS command line in the Workstation product.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
xmltptblcreat -cp <config path>
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **xmltptblcreat** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# xmlxrefcreat

## Purpose

Creates an empty **xmlxref** table.

**Note**
You must have the XML translation option in the Map Editor to use this command. This command is not available for use with the Visual Mapper.

**CAUTION**

**Use with caution! This program overwrites any data in the existing xmlxref table.**

## Files created

This table lists the names and descriptions of the files created when you run **xmlxref**creat.

| File name | Description |
|---|---|
| xmlxref.dat | The table that stores the XML cross-reference string as defined by the XML Trading Partnership rules.<br>This file is created in the trading partner directory. |
| xmlxref.idx | The index file that relates to the *xmlxref.dat* file.<br>This file is created in the trading partner directory. |

## Starting the program

Start **xmlxrefcreat** from the UNIX host command line for all Gentran:Server for UNIX product levels.

Start **xmlxrefcreat** from the DOS command line for the Workstation product.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
xmlxrefcreat -cp <config path>
```

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **xmlxrefcreat** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |

# xtablemv

## Purpose

Loads and unloads Standard Cross Reference Table records.

## Files created

This table lists the name and description of the file created when you run **xtablemv**.

| File name | Description |
|---|---|
| xtable.unl | The file created when the Standard Cross Reference Table records (*xtable.dat* and *xtable.idx*) are unloaded into ASCII format for moving from platform to platform. This file is created in the trading partner directory. |
| xtable.err | The error/log file created by **xtablemv**.<br>This file is created in:<br>◗ The host *temp* directory for Gentran:Server for UNIX<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start **xtablemv** from the **Unload**=>**Cross Reference Records** or **Load**=>**Cross Reference Records** option from the Tools menu within the Trading Partnership explorer.

## Using the command in a script or batch file

Use the following notation when including this command in a script or batch file:

```
xtablemv -[l/lo/u] [-cp <config path>]
```

**Note**
Do not include the braces or slashes in the command.

(Continued on next page)

## Parameters and variables

This table lists the parameters and variables defined for the **xtablemv** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*).<br>**Note**<br>This parameter is required. |
| l | Load from ASCII to ISAM file. |
| lo | Load from ASCII to ISAM file, while updating records (overwrite). |
| u | Unload from ISAM to ASCII file. |

# xtablerpt

## Purpose

The **xtablerpt** program runs a report that shows the records in the Standard Cross Reference Table.

## Files created

This table lists the name and description of the file created when you run **xtablerpt**.

| File name | Description |
|---|---|
| xtable.r_t | The Standard Cross Reference Table, which stores the Standard Cross Reference Table Report.<br>This file is created in:<br>◗ The *temp* directory for Gentran:Server for UNIX.<br>◗ The *rpt* directory for Gentran:Server Workstation. |

## Starting the program

Start this program from the command line. You can also use the command in a script or batch file.

**xtablerpt** {-cp <config path>}

## Parameters and variables

This table lists the parameters and variables defined for the **xtablerpt** command.

| Parameter | Definition |
|---|---|
| cp <config path> | The path to the user configuration file (*envprim.cfg*). |

(Continued on next page)

## Sample report

This is a sample of the Standard Cross Reference Table report.

```
                     STANDARD CROSS REFERENCE TABLE RECORD
                     -------------------------------------
DATE: 02/04/2000                                                    PAGE:     1
TIME: 09:50:58
INT CODE         GROUP CODE      REF VALUE 1   REF VALUE 2  STANDARD      TYPE

_______________  _______________ ____________  ___________  ____________  ________
05538293         773GY88          0            0            091001        EDIFACT
05538293         773GY88          4010                      C2/7          X12
```

CHAPTER
4

# File Record Layouts

**Contents** **Overview**

# Overview

## Chapter contents

This chapter describes the file record layouts for specific files used in Gentran:Server.

## File name extensions

This table explains the file types associated with the file name extensions of the files in this chapter.

| Extension | File Type |
|---|---|
| .cds | Standard codes file |
| .cfg | Configuration file |
| .dat | Indexed sequential access method (ISAM) file containing data |
| .idx | Indexed sequential access method (ISAM) file containing an index to the records in the .dat file |
| .std | Standards file |
| .ord | Diagnostic report file |
| .unl | Delimited, ASCII version of a .dat/idx file |

## .unl file layouts

The .unl versions of ISAM (.dat/idx) files contain the same record layout as their ISAM counterparts. The only difference is that the records in the .unl files are separated with delimiters.

## Abbreviations

These abbreviations are used in this chapter:

| Abbreviation | Meaning |
|---|---|
| APP | Application description |
| FA | Functional acknowledgment |

(Continued on next page)

| (Contd)<br>Abbreviation | Meaning |
|---|---|
| IG | Implementation guide |
| TP | Trading Partnership |

# apptptbl.dat/idx

## General information

This table contains general information about *apptptbl.dat*.

| Information | Details |
|---|---|
| File Type | Application Trading Partnership Rules records |
| File Name | *apptptbl.dat* |
| Record Types | 1 |

## Record lengths

The record length is 529, plus one newline character.

## Record layouts

This table contains record layout information for *apptptbl.dat*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | uniqid | 0 | 15 | Unique ID |
| char | appname | 15 | 61 | Application name |
| char | recordid | 76 | 129 | Record ID |
| char | fieldname | 205 | 129 | Field name |
| char | fld_loc | 334 | 5 | 0-based field location, byte offset or field number |
| char | fldstart | 339 | 3 | Starting position within the field |
| char | len | 342 | 3 | Length of data to extract from the field |
| char | tp_start | 345 | 3 | Starting position to deposit data into identification string |
| char | marker | 358 | 2 | Character to be used for identification string picture |

(Continued on next page)

| (Contd)<br>Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | fld_maxlen | 360 | 5 | 0 if variable length field<br>Maximum size of field if fixed length field |
| char | justify | 365 | 2 | Read from right or left |
| char | filler | 367 | 162 | Reserved for future use |
| char | N/A | 529 | 1 | Newline character |

# appxref.dat/idx

## General information

This table contains general information about *appxref.dat*.

| Information | Details |
|---|---|
| File Type | Trading Partnership cross-reference records. |
| File Name | appxref.dat |
| Record Types | 1 |

## Record lengths

The record length is 511.

## Record layouts

This table contains record layout information for *appxref.dat*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | appname | 0 | 61 | Name of application description or file definition |
| char | str | 61 | 41 | Value of identification string that is cross-referenced to Trading Partnership Code |
| char | tp_code | 102 | 16 | Trading Partnership Code that is cross-referenced to the identification string |
| char | filler | 118 | 393 | Reserved for future use |

# cattype.dat/idx

## General information

This table contains general information about *cattype.dat*.

| Information | Details |
|---|---|
| File Type | Trading Partnership category types |
| File Name | *cattype.dat* |
| Record Types | 1 |

## Record lengths

The record length is 320.

## Record layouts

This table contains record layout information for *cattype.dat*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | prompt1 | 0 | 16 | Trading Partnership category prompt 1 |
| char | descr1 | 16 | 41 | Description/help line 1 |
| char | prompt2 | 57 | 16 | Trading Partnership category prompt 2 |
| char | descr2 | 73 | 41 | Description/help line 2 |
| char | prompt3 | 114 | 16 | Trading Partnership category prompt 3 |
| char | descr3 | 130 | 41 | Description/help line 3 |
| char | prompt4 | 171 | 16 | Trading Partnership category prompt 4 |
| char | descr4 | 187 | 41 | Description/help line 4 |
| char | prompt5 | 228 | 16 | Trading Partnership category prompt 5 |
| char | descr5 | 244 | 41 | Description/help line 5 |
| char | unused | 285 | 35 | Reserved for future use |

# catvalue.dat/idx

## General information

This table contains general information about *catvalue.dat* and *catvalue.idx*.

| Information | Details |
|---|---|
| File Type | Values for Trading Partnership category types |
| File Name | *catvalue.dat, catvalue.idx* |
| Record Types | 1 |

## Record lengths

The record length is 48.

## Record layouts

This table contains record layout information for *catvalue.dat* and *catvalue.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | cat_no | 0 | 2 | Category number; 1 through 5 |
| char | cat_value | 2 | 11 | Category value |
| char | unused | 13 | 37 | Reserved for future use |

# contact.dat/idx

## General information

This table contains general information about *contact.dat* and *contact.idx*.

| Information | Details |
|---|---|
| File Type | contact |
| File Name | *contact.dat* (*contact.idx*) |
| Record Types | 1 |

## Record lengths

The record length is 640.

## Record layouts

This table contains record layout information for *contact.dat* and *contact.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | type | 0 | 2 | Contact related to:<br>◗ Interchange<br>◗ Group<br>◗ Trading Partnership |
| char | code | 2 | 16 | Contact code |
| char | name | 18 | 36 | Contact's name |
| char | addr1 | 54 | 36 | Address line 1 |
| char | addr2 | 90 | 36 | Address line 2 |
| char | city | 126 | 20 | City |
| char | state | 146 | 3 | State abbreviation |
| char | zipcode | 149 | 11 | Zip code |
| char | country | 160 | 16 | Country |
| char | phone1 | 176 | 16 | Phone number |

(Continued on next page)

| (Contd)<br>Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | ext1 | 192 | 7 | Extension |
| char | phone2 | 199 | 16 | Alternate phone number |
| char | ext2 | 215 | 7 | Alternate phone extension |
| char | fax | 222 | 16 | Fax number |
| char | internet | 238 | 81 | Internet address |
| char | comments | 319 | 256 | Free-form comments |
| char | unused | 575 | 65 | Unused filler for future use |

# dm.dat/idx, pat.dat/idx

## Introduction

This topic describes the Data Manager, pattern, configuration, and Data Manager Archive ISAM files.

## Data Manager ISAM File Layout

This table contains general information about *dm.dat, dm.idx, pat.dat,* and *pat.idx* files.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | pat_nam | 0 | 16 | Pattern Name |
| char | dm_name | 16 | 7 | Data Manager Name |
| char | tp_code | 23 | 32 | Trading Partner Code |
| char | tp_mod | 55 | 2 | Trading Partnership Modifier |
| char | dest_dir | 57 | 65 | Destination Directory |
| char | dest_file | 122 | 65 | Destination Filename |
| char | scr_name | 187 | 22 | Script Name |
| char | unused | 209 | 3 | Unused Filler (was doc_loc) |
| char | arc_swtch | 212 | 2 | Archive switch [y/n] |
| char | arc_loc | 214 | 21 | Archive Location |
| char | arc_hnd | 235 | 7 | Archive Handler |
| char | org_mbx | 242 | 16 | Origin Mailbox |
| char | dst_mbx1 | 258 | 16 | Destination Mailbox |
| char | dst_mbx2 | 274 | 16 | Destination Mailbox 2 |
| char | dst_mbx3 | 290 | 16 | Destination Mailbox 3 |
| char | dst_typ1 | 306 | 2 | Dest Type 1<br>◗ M = Mailbox<br>◗ L = Dist List |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | dst_typ2 | 308 | 2 | Dest Type 2<br>◗ M = Mailbox<br>◗ L = Dist List |
| char | dst_typ3 | 310 | 2 | Dest Type 3<br>◗ M = Mailbox<br>◗ L = Distribution List |
| char | errh_class | 312 | 9 | Class |
| char | dst_que_nm | 321 | 9 | Destination Queue Name - Future Usage |
| char | dst_que_grp | 330 | 9 | Destination Queue Group - Future Usage |
| char | dst_que_pr | 339 | 2 | Destination Queue Priority - Future Usage |
| char | description | 341 | 41 | Description information |
| char | dm_sequence | 382 | 3 | Agent’s position within a flow |
| char | mailboxxref | 385 | 2 | Perform Mailbox cross reference |
| char | filler | 387 | 99 | Unused filler |

## Archive ISAM File Layout

This table contains general information about <dm_name>_arch.dat and <dm_name>_arch.idx, where dm_name represents the name of a data manager.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | rev | 0 | 3 | Archive handler revision code |
| char | tp_code | 3 | 15 | Excel/data managers Trading Partnership code |
| char | mod | 18 | 2 | Data manager modifier |
| char | doc_ref_no | 20 | 41 | Unique doc reference number this transaction. |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | t_date | 61 | 9 | Data manager drop-off date (CCYYMMDD) |
| char | t_time | 70 | 9 | Data manager drop-off time |
| char | a_date | 79 | 9 | Archive date (CCYYMMDD) |
| char | a_time | 88 | 9 | Archive time |
| char | char_cnt | 97 | 10 | Number of bytes in this transaction |
| char | ar_name | 107 | 20 | Archive file name inbound |
| char | ar_offset | 127 | 10 | Archive offset |
| char | iox | 137 | 2 | Direction of data manager<br>i = inbound<br>o = outbound<br>x = other |
| char | status | 139 | 2 | Record the result of data manager handling<br>◗ A = Application Error<br>◗ C = No config<br>◗ D = for duplicates<br>◗ E = this transaction had errors<br>◗ N = for good transaction<br>◗ P = for no Trading Partnership code<br>◗ Q = Destination Queue Error<br>◗ R = Routing error<br>◗ S = Destination Queue Error<br>◗ T = Translation failed |
| char | retv_dt | 141 | 9 | Date when this data was last accessed (CCYYMMDD) |
| char | retv_tm | 150 | 9 | Record of this data last access (HHMMSS)<br>(Continued on next page) |

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | retv_type | 159 | 2 | Record of this data last access<br>◗ C for copy no transaction registrar access<br>◗ R for Remove transaction registrar entry |
| char | nl | 161 | 1 | End of file |

# .dmcfg.dat/idx

## General information

This table contains general information about the *.dmcfg.dat* and *.dmcfg.idx*. These files are used by *fmgr*.

| Information | Details |
|---|---|
| File Type | data manager |
| File Name | *.dmcfg.dat (.dmcfg.idx)* |
| Record Types | 1 |

## Record lengths

The record length is 400.

## Record layouts

This table contains record layout information for *.dmcfg*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | name | 0 | 5 | Agent name |
| char | actv | 5 | 2 | Activity<br>◗ n = not active<br>◗ y = active<br>◗ a = automatic. Agent is forked when **fmgr** starts. |
| char | pid | 7 | 11 | UNIX Process ID |

(Continued on next page)

| (Contd)<br>Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | type | 18 | 2 | Record type<br>◗ a = archive<br>◗ d = download/udf<br>◗ f = file<br>◗ h = host command card<br>◗ i = inbd<br>◗ l = lm<br>◗ m = app<br>◗ u = base dm model<br>◗ x = translate |
| char | argv | 20 | 48 | Description for agent. The default is inbound data manager for inbd type. |
| char | buffer | 68 | 332 | Unused |

# edistat.i/o, edihist.dat/idx

## General information

This table contains general information about *edistat.i* and *edistat.o*, and *edihist.dat* and *edihist.idx*.

| Information | Details |
|---|---|
| File Type | audit/archive |
| File Name | *edistat.i*, *edistat.o*, *edihist.dat* (*edihist.idx*) |
| Record Types | 1 |

## Record lengths

The record length is 1024.

## Record layouts

This table contains record layout information for *edistat.i* and *edistat.o*, and *edihist.dat* and *edihist.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | ic_send | 0 | 36 | Interchange ID of sender of set |
| char | ic_recv | 36 | 36 | Interchange ID of receiver of set |
| char | gs_send | 72 | 36 | Group ID of sender of set |
| char | gs_recv | 108 | 36 | Group ID of receiver of set |
| char | ic_ctl_no | 144 | 15 | Interchange control number |
| char | grp_ctl_no | 159 | 15 | Functional group control number |
| char | set_ctl_no | 174 | 15 | Transaction set control number |
| char | rec_type | 189 | 4 | Record type |
| char | tp_code | 193 | 16 | Trading Partner Code |
| char | set_id | 209 | 7 | Transaction set ID |
| char | t_date | 216 | 9 | Date of translation |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | t_time | 225 | 9 | Time of translation |
| char | ic_date | 234 | 9 | Interchange date |
| char | ic_time | 243 | 9 | Interchange time |
| char | gs_date | 252 | 9 | Group date |
| char | gs_time | 261 | 9 | Group time |
| char | fa_date | 270 | 9 | Date functional acknowledgment expected/arrived |
| char | fa_time | 279 | 9 | Time functional acknowledgment expected/arrived |
| char | purge_date | 288 | 9 | Date after which record may be purged |
| char | gp_id | 297 | 7 | Func group ID of sets acknowledged |
| char | seg_term | 304 | 2 | Segment terminator |
| char | seg_cnt | 306 | 10 | Number of segments translated |
| char | char_cnt | 316 | 10 | Number of characters translated |
| char | doc_ref_no | 326 | 41 | Document reference number |
| char | env_flag | 367 | 2 | Envelope this file? (Y/N) |
| char | fa_req | 369 | 2 | Acknowledgment expected? Y/N |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | fa_stat | 371 | 2 | FA Status:<br>◗ A = Acknowledgment received - OK<br>◗ D = Acknowledgement delivered<br>◗ E = Acknowledgment received - ERRORS, accepted<br>◗ N = Acknowledgement not expected<br>◗ P = Acknowledgment received - ERRORS, partly accepted<br>◗ R = Acknowledgment arrived - ERRORS, rejected<br>◗ Y = Acknowledgment expected<br>◗ V = Verbal acknowledgement OK<br>◗ Z =Verbal acknowledgement with errors |
| char | trans_infile | 373 | 128 | Input file to translator |
| char | trans_outfile | 501 | 128 | Output file to translator |
| char | trans_offset | 629 | 10 | Offset of set to be archived; rec_type = "IN" offset of input rec_type = "OUT" offset of output |
| char | arch_dir | 639 | 116 | Directory containing arch_no file |
| char | arc_file | 755 | 13 | Archive file name (YYMMDDHH.arc) |
| char | arc_offset | 768 | 10 | Archive file offset to set |
| char | fa_type | 778 | 2 | Acknowledgement rcvd type; 1 = 997, 2 = 999, 3 = CONTRL |
| char | fa_dir | 780 | 116 | Directory containing FA file |
| char | fa_file | 896 | 13 | FA file name (YYMMDDHH.arc) |
| char | fa_offset | 909 | 10 | FA file offset to set |
| char | usrbuf | 919 | 53 | Area for user specified data. Fields are delimited with tildes (~). |

(Continued on next page)

| (Contd)<br>Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | result | 972 | 2 | Translation error code;<br>1 = error, 0 = OK |
| char | unused | 974 | 50 | Reserved for future use |

# envaux.cfg

## General information

This table contains general information about *envaux.cfg.*

| Information | Details |
|---|---|
| File Type | User directory list |
| File Name | *envaux.cfg* |
| Record Types | 3 |

## Required string

The first line of the *envaux.cfg* file must contain this string:

EDI-CfgClient

## Record layouts

This table contains record layout information for *envaux.cfg*.

| Description | Record Length |
|---|---|
| Report directory | 128 |
| Help/text files directory | 128 |
| Local temporary directory | 128 |

# envprim.cfg

## General information

This table contains general information about *envprim.cfg*

| Information | Details |
|---|---|
| File Type | User directory list |
| File Name | *envprim.cfg* |
| Record Types | 10 |

This file is used in the Application Integration subsystem and in the Visual Mapper subsystem.

## Record layouts

This table contains record lengths for *envprim.cfg*.

| Record | Record Length |
|---|---|
| Standards directory | 127 |
| IG directory | 127 |
| App/File Definitions directory | 127 |
| Map files directory | 127 |
| TP files directory | 127 |
| EDI Status Archive directory | 127 |
| EDI History Archive directory | 127 |
| Temporary directory (User) | 127 |
| Data files directory | 127 |

**Notes**

The record length does not include the mandatory trailing null character.

Records (for example, the IG directory) used only with the Visual Mapper subsystem are blank for the Application Integration subsystem.

# generic.dat/idx

## General information

This table contains general information about the *generic.dat* file used with the Visual Mapper.

| Information | Details |
|---|---|
| File Type | Generic synonym file |
| File Name | *generic.dat* (*generic.idx*) |
| Record Types | 1 |

## Record lengths

The record length is 208.

## Record layouts

This table contains record layout information for *generic.dat* and *generic.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | uname | 0 | 20 | User-defined list name (or source label for map-specific files) |
| char | inval | 20 | 80 | Value of source data item to be replaced |
| char | outval | 100 | 80 | Replacement value for data item |
| char | unused | 180 | 28 | Unused filler |

# Life Cycle Table: Informix

## Introduction

This topic describes the Informix Life Cycle Table.

## The lc221 table

This table describes the columns in the lc2211 Life Cycle table.

| Name | Type | Description |
|---|---|---|
| TP | VARCHAR(31) | Trading Partnership identification code |
| DOC | VARCHAR(40) | Document reference number |
| IOX | VARCHAR(1) | Direction of data manager<br>◗ i = inbound<br>◗ o = outbound<br>◗ x = other |
| SEQ | VARCHAR(2) | Sequence number |
| ISUNIQ | VARCHAR(9) | Unique identifier assigned in the data manager |
| MYISID | VARCHAR(35) | Interchange sender code |
| MYGSID | VARCHAR(35) | Application/group sender code |
| TPISID | VARCHAR(35) | Interchange receiver code |
| TPGSID | VARCHAR(35) | Application receiver code |
| GSVERS | VARCHAR(15) | Version number |
| STSETID | VARCHAR(6) | Set identifier |
| PGM | VARCHAR(6) | Data manager name |
| RSLT | VARCHAR(1) | Data manager result code (0=Success) |
| DT | DATE | Data manager process date (CCYYMMDD) |
| TM | VARCHAR(9) | Data manager process time (HHMMSS) |
| SDIR | VARCHAR(60) | Source directory name (work directory) |

(Continued on next page)

| (Contd)<br>Name | Type | Description |
|---|---|---|
| SFIL | VARCHAR(60) | Source file name (drop-off name) |
| DDIR | VARCHAR(60) | Destination directory name |
| DFIL | VARCHAR(60) | Destination file name |
| ADIR | VARCHAR(60) | Archive directory |
| ISCTL | VARCHAR(15) | Interchange control number |
| GSCTL | VARCHAR(15) | Group control number |
| STCTL | VARCHAR(15) | Set control number |
| TDT | DATE | Translation date (CCYYMMDD) |
| TTM | VARCHAR(9) | Translation time (HHMMSS) |
| FADT | DATE | Functional acknowledgment date (CCYYMMDD) |
| FATM | VARCHAR(9) | Functional acknowledgment time (HHMMSS) |
| FAREQ | VARCHAR(1) | Functional acknowledgment request flag<br>Values:<br>◗ A = Accept<br>◗ P = Partially accepted<br>◗ R = Reject<br>◗ E = Accepted with errors<br>◗ N = Inbound: no acknowledgment generated outbound: no acknowledgment expected<br>◗ Y = Inbound: acknowledgment to be generated, outbound: acknowledgment expected |
| SEGCNT | VARCHAR(9) | Segment count |
| CHARCNT | VARCHAR(9) | Character count |
| MAILGROUP | VARCHAR(9) | (Reserved for Advanced Data Distribution mailbox operations) |

(Continued on next page)

## The LCDestInfo table

This table describes the columns in the LCDestInfo secondary Life Cycle table. This table joins the parent table based on the **tp**, **doc**, and **iox** fields.

| Name | Type | Description |
|---|---|---|
| ActivitySysKey | VARCHAR(16) | Reserved for future use |
| tp | VARCHAR(36) | Trading Partnership code expanded |
| doc | VARCHAR(81) | Document Reference |
| IOX | VARCHAR(2) | Direction of data manager<br>◗ i = inbound<br>◗ o = outbound<br>◗ x = other |
| mailbagid | VARCHAR(10) | Run number (Mail bag ID) |
| DestType | VARCHAR(4) | Reserved for future use<br>(Destination Type) |
| DestHost | VARCHAR(128) | Destination host name |
| DestID | VARCHAR(16) | Destination File Unique Key |
| DestUser | VARCHAR(128) | Reserved for future use<br>(Destination User) |
| DestDir | VARCHAR(60) | Destination Directory Name |
| DestFile | VARCHAR(128) | Destination File Name |
| DestCharCnt | VARCHAR(16) | Destination Character Count |
| DestRecCnt | VARCHAR(16) | Destination Record Count |
| DestResult | VARCHAR(3) | Translation Result |
| DestContentType | VARCHAR(125) | File Type (Binary, EDI, etc.) |
| DestContentSubType | VARCHAR(125) | Reserved for future use |
| DestDesc | VARCHAR(80) | Description of subject |

(Continued on next page)

| (Contd)<br>Name | Type | Description |
|---|---|---|
| DestEncrypt | VARCHAR(2) | Reserved for future use<br>(Encryption Flag) |
| DestCmp | VARCHAR(2) | Reserved for future use<br>(Compression Flag) |

# Life Cycle Table: Oracle

## Introduction

This topic describes the Oracle Life Cycle Table.

## The lc221 table

This table describes the columns in the lc221 Life Cycle table.

| Name | Type | Description |
|---|---|---|
| TP | VARCHAR2(31) | Trading Partnership code |
| DOC | VARCHAR2(40) | Document reference number |
| IOX | VARCHAR2(1) | Direction of data manager<br>◗ i = inbound<br>◗ o = outbound<br>◗ x = other |
| SEQ | VARCHAR2(2) | Sequence number |
| ISUNIQ | VARCHAR2(9) | Unique identifier assigned in the data manager |
| MYISID | VARCHAR2(35) | Interchange sender code |
| MYGSID | VARCHAR2(35) | Application/group sender code |
| TPISID | VARCHAR2(35) | Interchange receiver code |
| TPGSID | VARCHAR2(35) | Application receiver code |
| GSVERS | VARCHAR2(15) | Version number |
| STSETID | VARCHAR2(6) | Set identifier |
| PGM | VARCHAR2(6) | Data manager name |
| RSLT | VARCHAR2(3) | Data manager result code (0=Success) |
| DT | DATE | Data manager process date (CCYYMMDD) |
| TM | VARCHAR2(9) | Data manager process time (HHMMSS) |
| SDIR | VARCHAR2(60) | Source directory name (work directory)<br>(Continued on next page) |

| (Contd)<br>Name | Type | Description |
|---|---|---|
| SFIL | VARCHAR2(60) | Source file name (drop-off name) |
| DDIR | VARCHAR2(60) | Destination directory name |
| DFIL | VARCHAR2(60) | Destination file name |
| ADIR | VARCHAR2(60) | Archive directory |
| ISCTL | VARCHAR2(15) | Interchange control number |
| GSCTL | VARCHAR2(15) | Group control number |
| STCTL | VARCHAR2(15) | Set control number |
| TDT | DATE | Translation date (CCYYMMDD) |
| TTM | VARCHAR2(9) | Translation time (HHMMSS) |
| FADT | DATE | Functional acknowledgment date (CCYYMMDD) |
| FATM | VARCHAR2(9) | Functional acknowledgment time (HHMMSS) |
| FAREQ | VARCHAR2(1) | Functional acknowledgment request flag<br>◗ A = Accept<br>◗ P = Partially accepted<br>◗ R = Reject<br>◗ E = Accepted with errors<br>◗ N = inbound: No acknowledgment generated, outbound: No acknowledgment expected<br>◗ Y = inbound: Acknowledgment to be generated, outbound: Acknowledgment expected |
| SEGCNT | VARCHAR2(9) | Segment count |
| CHARCNT | VARCHAR2(9) | Character count |
| MAILGROUP | VARCHAR2(9) | (Reserved for Advanced Data Distribution mailbox operations) |

(Continued on next page)

## The LCDestInfo table

This table describes the columns in the LCDestInfo secondary Life Cycle table. This table joins the parent table based on the **tp**, **doc**, and **iox** fields.

| Name | Type | Description |
|---|---|---|
| ActivitySysKey | VARCHAR2(16) | Reserved for future use |
| tp | VARCHAR2(36) | Trading Partnership code expanded |
| doc | VARCHAR2(81) | Document Reference |
| IOX | VARCHAR2(2) | Direction of data manager<br>◗ i = inbound<br>◗ o = outbound<br>◗ x = other |
| mailbagid | VARCHAR2(10) | Run number (Mail bag ID) |
| DestType | VARCHAR2(4) | Reserved for future use<br>(Destination Type) |
| DestHost | VARCHAR2(128) | Destination host name |
| DestID | VARCHAR2(16) | Destination File Unique Key |
| DestUser | VARCHAR2(128) | Reserved for future use<br>(Destination User) |
| DestDir | VARCHAR2(60) | Destination Directory Name |
| DestFile | VARCHAR2(128) | Destination File Name |
| DestCharCnt | VARCHAR2(16) | Destination Character Count |
| DestRecCnt | VARCHAR2(16) | Destination Record Count |
| DestResult | VARCHAR2(3) | Translation Result |
| DestContentType | VARCHAR2(125) | File Type (Binary, EDI, etc.) |
| DestContentSubType | VARCHAR2(125) | Reserved for future use |
| DestDesc | VARCHAR2(80) | Description of subject |

(Continued on next page)

| (Contd)<br>Name | Type | Description |
|---|---|---|
| DestEncrypt | VARCHAR2(2) | Reserved for future use<br>(Encryption Flag) |
| DestCmp | VARCHAR2(2) | Reserved for future use<br>(Compression Flag) |

# Life Cycle Table: Sybase

## Introduction

This topic describes the Sybase Life Cycle Table.

## The lc221 table

This table describes the columns in the lc221 Life Cycle table.

| Name | Type | Description |
|---|---|---|
| TP | CHAR(40) | Trading Partnership identification code |
| DOC | CHAR(40) | Document reference number |
| IOX | CHAR(1) | Direction of data manager<br>◗ i = inbound<br>◗ o = outbound<br>◗ x = other |
| SEQ | CHAR(2) | Sequence number |
| ISUNIQ | CHAR(9) | Unique identifier assigned in the data manager |
| MYISID | CHAR(35) | Interchange sender code |
| MYGSID | CHAR(35) | Application/group sender code |
| TPISID | CHAR(35) | Interchange receiver code |
| TPGSID | CHAR(35) | Application receiver code |
| GSVERS | CHAR(15) | Version number |
| STSETID | CHAR(6) | Set identifier |
| PGM | CHAR(6) | Data manager name |
| RSLT | CHAR(3) | Data manager result code (0=Success) |
| DT | DATE | Data manager process date (CCYYMMDD) |
| TM | CHAR(9) | Data manager process time (HHMMSS) |
| SDIR | CHAR(60) | Source directory name (work directory) |

(Continued on next page)

| (Contd)<br>Name | Type | Description |
|---|---|---|
| SFIL | CHAR(60) | Source file name (drop-off name) |
| DDIR | CHAR(60) | Destination directory name |
| DFIL | CHAR(60) | Destination file name |
| ADIR | CHAR(60) | Archive directory |
| ISCTL | CHAR(15) | Interchange control number |
| GSCTL | CHAR(15) | Group control number |
| STCTL | CHAR(15) | Set control number |
| TDT | DATE | Translation date (CCYYMMDD) |
| TTM | CHAR(9) | Translation time (HHMMSS) |
| FADT | DATE | Functional Acknowledgment Date (CCYYMMDD) |
| FATM | CHAR(9) | Functional Acknowledgment Time (HHMMSS) |
| FAREQ | CHAR(1) | Functional acknowledgment request flag<br>◗ A = accept<br>◗ P = Partially accepted<br>◗ R = Reject<br>◗ E = Accepted with errors<br>◗ N = inbound: No acknowledgment generated outbound: No acknowledgment expected<br>◗ Y = inbound: Acknowledgment to be generated, outbound: Acknowledgment expected |
| SEGCNT | CHAR(9) | Segment count |
| CHARCNT | CHAR(9) | Character count |
| MAILGROUP | CHAR(9) | (Reserved for Advanced Data Distribution mailbox operations) |

(Continued on next page)

## The LCDestInfo table

This table describes the columns in the LCDestInfo secondary Life Cycle table. This table joins the parent table based on the **tp**, **doc**, and **iox** fields.

| Name | Type | Description |
|---|---|---|
| ActivitySysKey | CHAR(16) | Reserved for future use |
| tp | CHAR(36) | Trading Partnership code expanded |
| doc | CHAR(81) | Document Reference |
| IOX | CHAR(2) | Direction of data manager<br>◗ i = inbound<br>◗ o = outbound<br>◗ x = other |
| mailbagid | CHAR(10) | Run number (Mail bag ID) |
| DestType | CHAR(4) | Reserved for future use<br>(Destination Type) |
| DestHost | CHAR(128) | Destination host name |
| DestID | CHAR(16) | Destination File Unique Key |
| DestUser | CHAR(128) | Reserved for future use<br>(Destination User) |
| DestDir | CHAR(60) | Destination Directory Name |
| DestFile | CHAR(128) | Destination File Name |
| DestCharCnt | CHAR(16) | Destination Character Count |
| DestRecCnt | CHAR(16) | Destination Record Count |
| DestResult | CHAR(3) | Translation Result |
| DestContentType | CHAR(125) | File Type (Binary, EDI, etc.) |
| DestContentSubType | CHAR(125) | Reserved for future use |
| DestDesc | CHAR(80) | Description of subject<br>(Continued on next page) |

| (Contd)<br>Name | Type | Description |
|---|---|---|
| DestEncrypt | CHAR(2) | Reserved for future use<br>(Encryption Flag) |
| DestCmp | CHAR(2) | Reserved for future use<br>(Compression Flag) |

# org.dat/idx

## General information

This table contains general information about *org.dat* and *org.idx*. This table is used by *smgr*.

| Information | Details |
|---|---|
| File Type | Organization |
| File Name | *org.dat* (*org.idx*) |
| Record Types | 1 (interchange level with interchange IDs)<br>2 (Group/application level with interchange and group IDs) |

## Record lengths

The record length is 321.

## Record layouts

This table contains record layout information for *org.dat* and *org.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | u_ic_id | 0 | 36 | User's interchange ID |
| char | p_ic_id | 36 | 36 | Partner's interchange ID |
| char | u_ap_id | 72 | 36 | User's group/application ID |
| char | p_ap_id | 108 | 36 | Partner's group/application ID |
| char | rec_type | 144 | 2 | Record type:<br>◗ 1=Interchange<br>◗ 2=Group |
| char | org_code | 146 | 16 | Organization code |
| char | descr | 162 | 31 | Organization description/name |
| char | base_rec | 193 | 16 | Base Trading Partner Code |
| char | last_ctl_rcv | 209 | 15 | Last control or batch number received |

(Continued on next page)

| (Contd | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | last_ctl_snt | 224 | 15 | Last control or batch number sent |
| char | unused | 239 | 82 | Unused filler for future use |

# .scrcfg.dat/idx

## General information

This table contains general information about *.scrcfg.dat* and *.scrcfg.idx* and their use by *smgr*.

| Information | Details |
|---|---|
| File Type | Scripts |
| File Name | *.scrcfg.dat (.scrcfg.idx)* |
| Record Types | 1 |

## Record lengths

The record length is 53.

## Record layouts

This table contains record layout information for *.scrcfg.dat* and *.scrcfg.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | scr | 0 | 15 | Script name |
| char | stat | 15 | 7 | Status of script (active or inactive) |
| char | desc | 22 | 31 | Description of script |

# <standard version>.std

## General information

This table contains general information about the <*standard version*>*.std* file used with the Visual Mapper.

| Information | Details |
|---|---|
| File Type | Standard description |
| File Name | standard version (3 chars) + standard release (3 characters) + ".std" |
| Record Types | 10 |

## Record lengths

This table contains record lengths for <*standard version*>*.std*.

| Record | Record Length |
|---|---|
| Standard title | 64 |
| Transaction set index | 64 |
| Component index | 80 |
| Set description | 80 |
| Segment description | 80 |
| Element block | 12 |
| Element description | 80 |
| Component description | 160 |
| Header segment description | 80 |
| Trailer segment description | 80 |

(Continued on next page)

## Record layouts

This table contains record layout information for *<standard version>.std*.

| Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Standard Title | char | title | 0 | 64 | Standard version and release |
| Transaction Set Index | char | id | 0 | 7 | Transaction/message ID |
| | char | pos | 7 | 9 | File offset to start of set |
| | char | title | 16 | 41 | Transaction/message title |
| | char | unused | 57 | 7 | Filler |
| Component Index | char | position | 0 | 8 | File pointer to block beginning |
| | char | ele_count | 8 | 8 | Number of records in block |
| | char | cmp_posn | 16 | 8 | c-component block beginning |
| | char | cmp_count | 24 | 8 | Number of records in block |
| | char | scmp_posn | 32 | 8 | s-component block beginning |
| | char | scmp_count | 40 | 8 | Number of records in block |
| | char | hdr_posn | 48 | 8 | Header segment description block beginning |
| | char | hdr_count | 56 | 8 | Number of records in block |
| | char | trl_posn | 64 | 8 | Trailer segment description block beginning |
| | char | trl_count | 72 | 8 | Number of records in block |
| | char | id | 0 | 7 | Transaction set ID |
| | char | title | 7 | 41 | Transaction set description<br>(Continued on next page) |

| (Contd) Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Component Index (cont.) | char | nu_of_segs | 48 | 4 | Number of segments in this set |
| | char | unused | 52 | 28 | Filler |
| Segment Description | char | id | 0 | 4 | Segment ID |
| | char | title | 4 | 41 | Segment description |
| | char | req_des | 45 | 2 | Segment requirement |
| | char | max_usage | 47 | 7 | Maximum usage |
| | char | repeat_count | 54 | 8 | Maximum loop count |
| | char | num_mand | 62 | 3 | Number of mandatory elements |
| | char | ssloop | 65 | 2 | Single segment loop flag |
| | char | unused | 67 | 13 | Filler |
| Element Block | char | seq | 0 | 4 | Sequence number |
| | char | ele_num | 4 | 6 | Element dictionary number |
| | char | ele_rqt | 10 | 2 | Element requirement |
| Element Description | char | ref_nu | 0 | 5 | Data element number |
| | char | title | 5 | 26 | Element description |
| | char | type | 31 | 3 | Element data type |
| | char | max_len | 34 | 4 | Element maximum length |
| | char | min_len | 38 | 3 | Element minimum length |
| | char | qualflag | 41 | 2 | 1 if qualifying previous |
| | char | unused | 43 | 37 | Filler<br>(Continued on next page) |

| **(Contd) Record** | **Type** | **Name** | **Offset** | **Length** | **Description** |
|---|---|---|---|---|---|
| Component Description | char | refno | 0 | 5 | Composite element number |
| | char | title | 5 | 26 | Composite element title |
| | char | count | 31 | 3 | Number of components |
| | char | comp_refno*24 | 34 | 5 | Component reference number/requirement |
| | char | req | 154 | 2 | Component requirement |
| | char | seq | 156 | 4 | Sequence number |
| Header Segment Description | char | id | 0 | 4 | Segment ID |
| | char | title | 4 | 41 | Segment description |
| | char | req_des | 45 | 2 | Segment requirement |
| | char | max_usage | 47 | 7 | Maximum usage |
| | char | repeat_count | 54 | 8 | Maximum loop count |
| | char | num_mand | 62 | 3 | Number of mandatory elements |
| | char | ssloop | 65 | 2 | Single segment loop flag |
| | char | num_elm | 67 | 3 | Number of elements |
| | char | unused | 70 | 10 | Filler |
| Trailer Segment Description | char | id | 0 | 4 | Segment ID |
| | char | title | 4 | 41 | Segment description |
| | char | req_des | 45 | 2 | Segment requirement |
| | char | max_usage | 47 | 7 | Maximum usage |
| | char | repeat_count | 54 | 8 | Maximum loop count |
| | char | num_mand | 62 | 3 | Number of mandatory elements |
| | char | ssloop | 65 | 2 | Single segment loop flag |
| | char | num_elm | 67 | 3 | Number of elements |
| | char | unused | 70 | 10 | Filler |

# <standard version>.CDS

## General information

This table contains general information about the *<standard version>.CDS* file used with the Visual Mapper.

| Information | Details |
|---|---|
| File Type | ID code validation |
| File Name | <version>.CDS (for example, 003020.CDS) |
| Record Types | 1 |

## Record lengths

Record lengths are variable length, newline terminated.

## Record layouts

This table contains record layout information for *<standard version>.CDS*.

| Type | Name | Length | Description |
|---|---|---|---|
| char | ddno | 4 | Element data dictionary number |
| char | idcode | 1-6 | Valid element code |
| char | newline | 1 | Carriage return/line feed |

# tp.dat/idx

## General information

This table contains general information about *tp.dat* and *tp.idx*.

| Information | Details |
|---|---|
| File Type | trading partner |
| File Name | *tp.dat* (*tp.idx*) |
| Record Types | 1 |

## Record lengths

The record length is 1230.

## Record layouts

This tables contain record layouts for *tp.dat* and *tp.idx*.

| Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Primary Key | char | tp_code | 0 | 16 | Trading Partner Code |
| Secondary Key | char | u_ic_id | 16 | 36 | User's interchange ID |
| | char | p_ic_id | 52 | 36 | Partner's interchange ID |
| | char | u_ap_id | 88 | 36 | User's application ID |
| | char | p_ap_id | 124 | 36 | Partner's application ID |
| | char | doc_ver | 160 | 13 | Document/message version ID |
| | char | doc_type | 173 | 7 | Document/message type (set ID) |
| Inbound and Outbound | char | descr | 180 | 31 | Partnership description/ name |
| | char | category | 211 | 5*11 | User-defined partner categories |

(Continued on next page)

| (Contd) Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Inbound and Outbound (cont'd) | char | ic_hdr | 266 | 4 | Interchange header |
| | char | ctl_no_sw | 270 | 2 | Control number switch<br>◗ G=Global<br>◗ L=Local |
| Inbound Only | char | table_in | 272 | 61 | Inbound translation table |
| | char | in_ropath | 333 | 65 | Output file directory path |
| | char | in_route | 398 | 65 | Output file name |
| | char | in_options | 463 | 11 | Inbound translation options |
| | char | ic_ctl_no | 474 | 15 | Interchange control number |
| | char | ic_ctl_sw | 489 | 2 | Sequence check flag |
| | char | gs_ctl_no | 491 | 15 | Group control number |
| | char | gs_ctl_sw | 506 | 2 | Sequence check flag |
| | char | st_ctl_no | 508 | 15 | Set/Message control number |
| | char | st_ctl_sw | 523 | 2 | Sequence check flag |
| | char | accept_err | 525 | 2 | Accept sets in error switch |
| | char | func_ack_sw | 527 | 2 | Generate acknowledgment flag |
| | char | edi_out_tp | 529 | 16 | Outbound EDI Trading Partnership Code - standard to standard |
| Outbound Only | char | table_out | 545 | 61 | Outbound translation table |
| | char | out_ropath | 606 | 65 | Output file directory path<br>(Continued on next page) |

| (Contd) Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Outbound Only (cont'd) | char | out_route | 671 | 65 | Output file name |
| | char | out_options | 736 | 11 | Outbound translation options |
| | char | func_ack_rq | 747 | 2 | Acknowledgment required |
| | char | func_ack_pr | 749 | 11 | FA waiting period (in seconds) |
| | char | env_flag | 760 | 2 | One interchange per file? |
| | char | elem_sep | 762 | 3 | Element separator |
| | char | comp_sep | 765 | 3 | Component/ sub-element separator |
| | char | seg_term | 768 | 3 | Segment terminator |
| | char | dec_mark | 771 | 3 | Decimal mark - EDIFACT |
| | char | rel_char | 774 | 3 | Release character - EDIFACT |
| | char | syn_level | 777 | 3 | Syntax level - EDIFACT |
| | char | gen_una | 780 | 2 | Generate UNA segment (y/n) EDIFACT |
| | char | gen_group_ hdr | 782 | 2 | Generate group header EDIFACT/TRADACOM |
| Inbound Only | char | file_gen_no | 784 | 5 | File generation number |
| | char | file_gen_sw | 789 | 2 | Sequence check flag |
| | char | arch_in_sw | 791 | 2 | Archive inbound file (y/n) |
| | char | arch_in_pr | 793 | 4 | Number of days to archive inbound file |
| | char | arch_out_sw | 797 | 2 | Archive outbound file (y/n)<br>(Continued on next page) |

| (Contd) Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Inbound Only (cont'd) | char | arch_out_pr | 799 | 4 | Number of days to archive outbound file |
| | char | trans_type | 803 | 1 | Translation type<br>◗ 1=Std2App<br>◗ 2=App2Std<br>◗ 3=Std2Std<br>◗ 4=App2App<br>◗ 5=Std2XML<br>◗ 6=App2XML<br>◗ 7=XML2Std<br>◗ 8=XML2App<br>◗ 9=XML2XML |
| | char | recon_ids | 804 | 1 | Use reconciliation IDs (y/n) |
| | char | UNH_asso_ code | 805 | 7 | Assigned association code (EDIFACT only) |
| | char | unused | 812 | 18 | For future use |
| Outbound Only | char | func_gp | 830 | 160 | Group structure |
| | char | interchange | 990 | 240 | Interchange structure |

## func_gp

The contents of the func_gp (Group Structure) record of tp.dat/.idx vary according to which Group Header Segment (GS, UNG, BAT) or NCPDP transaction header is used.

### GS Group Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | func_id_cd | 830 | 3 | Functional group ID code |
| uchar | gs_send_id | 833 | 16 | Application sender's ID |
| uchar | gs_recv_id | 849 | 16 | Application receiver's ID |
| uchar | dt_in_date | 865 | 9 | Data interchange date |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | dt_in_time | 874 | 9 | Data interchange time |
| uchar | gs_ctl_no | 883 | 10 | Group control number |
| uchar | resp_agency_cd | 893 | 3 | Responsible agency code |
| uchar | doc_ver | 896 | 13 | Version/release indicator |
| uchar | char_in_st_ctl | 909 | 2 | Number of characters in the set control number’s prefix |

### UNG Group Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | func_id | 830 | 7 | Functional group ID |
| uchar | snd_id | 837 | 36 | Application sender ID |
| uchar | snd_id_qual | 873 | 5 | Application sender qualifier |
| uchar | rec_id | 878 | 36 | Application recipient ID |
| uchar | rec_id_qual | 914 | 5 | Application recipient qualifier |
| uchar | prep_date | 919 | 9 | Date of preparation |
| uchar | prep_time | 928 | 5 | Time of preparation |
| uchar | grp_ref | 933 | 15 | Functional group reference number |
| uchar | ctl_agcy | 948 | 3 | Controlling agency |
| uchar | msg_ver | 951 | 4 | Message type version number |
| uchar | msg_rel | 955 | 4 | Message type release number |
| uchar | asn_code | 959 | 7 | Association assigned code |
| uchar | password | 966 | 15 | Application password |

(Continued on next page)

### BAT Group Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | bat_ref | 830 | 15 | Functional group ID |

### NCPDP transaction header (request and response)

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| **Request structure** | | | | |
| uchar | bin_no | 830 | 7 | BIN number |
| uchar | ver_rel | 837 | 3 | NCPDP version/release number |
| uchar | trans_code | 840 | 3 | Transaction code (E1, E1, E3, etc.) |
| uchar | ctl_no | 843 | 11 | Processor control number |
| uchar | trans_cnt | 854 | 2 | Transaction count (1 - 4) |
| uchar | prod_id_qual | 856 | 3 | Service provider ID qualifier |
| uchar | prod_id | 859 | 16 | Service provider ID |
| uchar | date_serv | 875 | 9 | Date of service |
| uchar | vendor_id | 884 | 11 | Software vendor/certification ID |
| **Response structure** | | | | |
| uchar | ver_rel | 895 | 3 | NCPDP version/release number |
| uchar | trans_code | 898 | 3 | Transaction code (E1, E1, E3, etc.) |
| uchar | trans_cnt | 900 | 2 | Transaction count (1 - 4) |
| uchar | resp_stat | 902 | 2 | Header response status |
| uchar | prod_id_qual | 904 | 3 | Service provider ID qualifier |
| uchar | prod_id | 907 | 16 | Service provider ID |
| uchar | date_serv | 923 | 9 | Date of service |

(Continued on next page)

## interchange

The contents of the interchange (Interchange Structure) record vary according to which Interchange Header Segment (BG, ICS, ISA, STX, UNB) or the NCPDP batch structure.

### BG Interchange Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | comm_id | 990 | 11 | Communication ID |
| uchar | comm_psw | 1001 | 11 | Communication password |
| uchar | in_send_id | 1012 | 13 | Application sender ID |
| uchar | in_recv_id | 1025 | 13 | Application receiver ID |
| uchar | dt_int_date | 1038 | 7 | Data interchange date |
| uchar | dt_int_time | 1045 | 5 | Data interchange time |
| uchar | in_ctl_no | 1050 | 6 | Transmission control number |
| uchar | u_ic_id | 1056 | 36 | Your interchange ID |
| uchar | p_ic_id | 1092 | 36 | Partner's interchange ID |
| uchar | u_ap_id | 1128 | 36 | Your group/application ID |
| uchar | p_ap_id | 1164 | 36 | Partner's group/application ID |

### ICS Interchange Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | el_sep | 990 | 3 | Data element separator |
| uchar | sub_el_sep | 993 | 3 | Sub-element separator |
| uchar | in_ctrl_st_id | 996 | 5 | Interchange standard ID |
| uchar | in_ctrl_vers | 1001 | 6 | Interchange version ID |
| uchar | in_send_qual | 1007 | 3 | Interchange sender qualifier |
| uchar | in_send_id | 1010 | 16 | Interchange sender ID |

(Continued on next page)

| (Contd | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | in_recv_qual | 1026 | 3 | Interchange receiver qualifier |
| uchar | in_recv_id | 1029 | 16 | Interchange receiver ID |
| uchar | in_sub_date | 1045 | 7 | Interchange submit date |
| uchar | in_sub_time | 1052 | 5 | Interchange submit time |
| uchar | in_ctl_no | 1057 | 10 | Interchange control number |
| uchar | seg_term | 1067 | 3 | Segment terminator |

### ISA Interchange Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | auth_info_qual | 990 | 3 | Authorization information qualifier |
| uchar | auth_info | 993 | 11 | Authorization information |
| uchar | sec_info_qual | 1004 | 3 | Security information qualifier |
| uchar | sec_info | 1007 | 11 | Security information |
| uchar | in_send_qual | 1018 | 3 | Interchange sender ID qualifier |
| uchar | in_send_id | 1021 | 16 | Interchange sender ID |
| uchar | in_recv_qual | 1037 | 3 | Interchange receiver ID qualifier |
| uchar | in_recv_id | 1040 | 16 | Interchange receiver ID |
| uchar | in_date | 1056 | 7 | Interchange date |
| uchar | in_time | 1063 | 5 | Interchange time |
| uchar | in_std_id | 1068 | 2 | Interchange standards ID |
| uchar | in_vers_id | 1070 | 6 | Interchange version ID |
| uchar | in_ctl_no | 1076 | 10 | Interchange control number |
| uchar | fnc_ack_req | 1086 | 2 | Acknowledgment requested |
| uchar | test_ind | 1088 | 2 | Test/production indicator |
| uchar | sub_el_sep | 1090 | 3 | Sub-element separator |

(Continued on next page)

### STX Interchange Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | syn_id | 990 | 5 | Syntax rules identifier |
| uchar | syn_ver | 995 | 2 | Syntax rules version |
| uchar | sender_code | 997 | 15 | Sender ID code |
| uchar | sender_name | 1012 | 36 | Sender name |
| uchar | recver_code | 1048 | 15 | Receiver ID code |
| uchar | recver_name | 1063 | 36 | Receiver name |
| uchar | trans_date | 1099 | 7 | Transmission date |
| uchar | trans_time | 1106 | 7 | Transmission time |
| uchar | sender_ref | 1113 | 15 | Sender reference |
| uchar | recver_ref | 1128 | 15 | Receiver reference |
| uchar | app_ref | 1143 | 15 | Application reference |
| uchar | priority | 1158 | 2 | Priority code |
| uchar | gen_cdt_seg | 1160 | 2 | Generate CDT segment |
| uchar | gen_sdt_seg | 1162 | 2 | Generate SDT segment |
| uchar | gen_typ_seg | 1164 | 2 | Generate TYP segment |
| uchar | gen_fil_seg | 1166 | 2 | Generate FIL segment |

### UNB Interchange Header Segment

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | syn_id | 990 | 5 | Syntax identifier |
| uchar | syn_ver | 995 | 2 | Syntax version |
| uchar | snd_id | 997 | 36 | Interchange sender ID |
| uchar | snd_id_qual | 1033 | 5 | Interchange sender ID qualifier |
| uchar | rv_rte_adr | 1038 | 15 | Reverse routing address |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | rec_id | 1053 | 36 | Interchange recipient ID |
| uchar | rec_id_qual | 1089 | 5 | Interchange recipient ID qualifier |
| uchar | rte_adr | 1094 | 15 | Routing address |
| uchar | prep_date | 1109 | 9 | Date of preparation |
| uchar | prep_time | 1118 | 5 | Time of preparation |
| uchar | int_ref | 1123 | 15 | Interchange control reference |
| uchar | pwd | 1138 | 15 | Recipient reference/password |
| uchar | pwd_qual | 1153 | 3 | Recipient reference qualifier |
| uchar | app_ref | 1156 | 15 | Application reference |
| uchar | prio_cd | 1171 | 2 | Processing priority code |
| uchar | ack_req | 1173 | 2 | Acknowledgment request |
| uchar | comm_agr | 1175 | 36 | Communications agreement |
| uchar | test_ind | 1211 | 2 | Test indicator |

### NCPDP batch transaction

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| Batch header structure | | | | |
| uchar | seg_id | 990 | 3 | Segment identifier: 00 = header |
| uchar | trans_type | 993 | 2 | Transmission type:<br>◗ T = Transaction<br>◗ R = Response<br>◗ E = Error |
| uchar | snd_id | 995 | 25 | Sender ID |
| uchar | batch_no | 1020 | 8 | Batch number |
| uchar | creat_date | 1028 | 9 | Creation date |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| uchar | creat_time | 1037 | 5 | Creation time |
| uchar | file_type | 1042 | 2 | File type:<br>◗ P = Production<br>◗ T = Test |
| uchar | ver_rel | 1044 | 3 | Batch version/release number |
| uchar | rec_id | 1047 | 25 | Receiver ID |
| **Batch transaction detail structure** | | | | |
| uchar | seg_id | 1072 | 3 | Segment identifier: G1 = Detail |
| uchar | ref_no | 1075 | 11 | Transaction reference number |
| **Batch trailer structure** | | | | |
| uchar | seg_id | 1086 | 3 | Segment identifier: 99 = trailer |
| uchar | batch_no | 1089 | 8 | Batch number |
| uchar | rec_cnt | 1097 | 11 | Service provider ID qualifier |
| uchar | message | 1108 | 36 | Message |

# tpmisc.dat/idx

## General information

This table contains general information about *tpmisc.dat* and *tpmisc.idx*.

| Information | Details |
|---|---|
| File Type | Miscellaneous Trading Partnership information |
| File Name | *tpmisc.dat, tpmisc.idx* |
| Record Types | 1 |

## Record lengths

The record length is 497.

## Record layouts

This table contains record layout information for *tpmisc.dat* and *tpmisc.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | tp_code | 0 | 16 | Trading Partnership Code |
| char | gateway_in | 16 | 16 | Inbound gateway |
| char | gateway_out | 32 | 16 | Outbound gateway |
| char | appout_msgtbl | 48 | 16 | Outbound application error message table |

### Server Process Flow Fields

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | mbox_id | 64 | 16 | Mailbox ID |
| char | ds_tbl | 80 | 11 | Document specifier table |
| char | pattern | 91 | 16 | Pattern |

(Continued on next page)

### Application Data Fields

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | app_in_ctl_no | 107 | 15 | Inbound application control number |
| char | app_out_ctl_no | 122 | 15 | Outbound application control number |

### Test Mode Fields

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | test_flag | 137 | 2 | Trading Partnership test flag (y/n) |
| char | u_test_ic_id | 139 | 36 | Your test interchange ID |
| char | p_test_ic_id | 175 | 36 | Trading partner’s test interchange ID |
| char | u_test_ap_id | 211 | 36 | Your test group ID |
| char | p_test_ap_id | 247 | 36 | Trading partner’s test group ID |
| char | u_test_ic_qual | 283 | 3 | Your interchange ID qualifier |
| char | p_test_ic_qual | 286 | 3 | Trading partner’s interchange ID qualifier |
| char | test_fname | 289 | 65 | Test file name |
| char | test_fpath | 354 | 65 | Path to test file name |

(Continued on next page)

### Creation of Historical Records

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | crt_input_hist | 419 | 2 | Create input historical records (y/n) |
| char | crt_output_hist | 421 | 2 | Create output historical records (y/n) |
| char | appSourceFileName | 423 | 65 | Application Source File Name |
| char | ack_requested | 488 | 2 | Acknowledgment request flag. Use UNB:09 as acknowledgment trigger. |
| char | Unused filler | 490 | 7 | Reserved for future use |

# tprecon.dat/idx

## General information

This table contains general information about *tprecon.dat* and *tprecon.idx*.

| Information | Details |
|---|---|
| File Type | Reconciliation IDs trading partner file |
| File Name | *tprecon.dat* (*tprecon.idx*) |
| Record Types | 1 |

## Record lengths

The record length is 320.

## Primary key

This table contains primary key information for *tprecon.dat* and *tprecon.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | tp_code | 0 | 16 | Trading Partnership Code |

## Record layouts

This table contains record layout information for *tprecon.dat* and *tprecon.idx*.

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | tp_code | 0 | 16 | Trading Partnership Code |
| char | u_ic_id | 16 | 36 | Sender's reconciliation interchange ID |
| char | p_ic_id | 52 | 36 | Receiver's reconciliation interchange ID |
| chap | u_ap_id | 88 | 36 | Sender's reconciliation application ID |

(Continued on next page)

| (Contd)<br>Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| chap | p_ap_id | 124 | 36 | Receiver's reconciliation application ID |
| char | unused | 160 | 160 | Unused filler to pad to even word boundary |

# tradacom.dat/ids

## General information

This table contains general information about *tradacom.dat* and *tradacom.idx*.

| Information | Details |
|---|---|
| File Type | TRADACOMS supplementary Trading Partnership |
| File Name | *tradacom.dat* (*tradacom.idx*) |
| Record Types | 1 |

## Record lengths

The record length is 1012.

## Record layouts

These tables contain record layout information for *tradacom.dat* and *tradacom.idx*.

### CDT Segment (Customer's Details)

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | tp_code | 0 | 16 | Trading Partner Code |
| char | cus_loc_num | 16 | 14 | Customer's location number |
| char | cus_id | 30 | 18 | Customer's ID |
| char | cus_name | 48 | 41 | Customer's name |
| char | cus_adr1 | 89 | 36 | Customer's address 1 |
| char | cus_adr2 | 125 | 36 | Customer's address 2 |
| char | cus_adr3 | 161 | 36 | Customer's address 3 |
| char | cus_adr4 | 197 | 36 | Customer's address 4 |
| char | cus_post_code | 233 | 9 | Customer's postal code |

(Continued on next page)

| (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | cus_n_vat | 242 | 10 | Customer's numeric VAT number |
| char | cus_a_vat | 252 | 18 | Customer's alphanumeric VAT number |

### SDT Segment (Supplier's Details)

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | sup_loc_num | 270 | 14 | Supplier's location number |
| char | sup_id | 284 | 18 | Supplier's ID |
| char | sup_name | 302 | 41 | Supplier's name |
| char | sup_adr1 | 343 | 36 | Supplier's address 1 |
| char | sup_adr2 | 379 | 36 | Supplier's address 2 |
| char | sup_adr3 | 415 | 36 | Supplier's address 3 |
| char | sup_adr4 | 451 | 36 | Supplier's address 4 |
| char | sup_post_code | 487 | 9 | Supplier's postal code |
| char | sup_n_vat | 496 | 10 | Supplier's numeric VAT number |
| char | sup_a_vat | 506 | 18 | Supplier's alphanumeric VAT number |

### TYP Segment (Transaction Type)

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | trans_code | 524 | 5 | Transaction code |
| char | trans_type | 529 | 13 | Transaction type |

(Continued on next page)

### FIL Segment (File Details)

| Type | Name | Offset | Length | Description |
|---|---|---|---|---|
| char | file_gen_num | 542 | 5 | File generation number |
| char | unused | 547 | 465 | For future use |

# trans.ord

## General information

This table contains general information about *trans.ord*.

| Information | Details |
|---|---|
| File Type | Diagnostics Report |
| File Name | trans.ord |
| Record Types | 5 |

## Record lengths

The record length is:

- Headers = Variable
- Detail Lines = 77

## Record layouts

These tables contain record layout information for *trans.ord*.

| Type | Offset | Length | Description |
|---|---|---|---|
| char | 0 | Variable | Input File = <Input File Name> |
| char | 0 | Variable | Table File = <Map Table Path and File Name> |

### Section Header

| Type | Offset | Length | Description |
|---|---|---|---|
| char | 0 | Variable | File Line = <Input file line described in this section> |
| char | 0 | Variable | Record ID = <Source record ID for this section> |

(Continued on next page)

### Section Information

| Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Field | char | srce_field | 0 | 3 | Number of source field/ element |
| | char | blanks | 3 | 5 | Blank spaces for formatting |
| Input | char | input | 8 | 18 | Current value of the source field/element unless the value in the next column is a system variable; if the value in the next column is a system variable, the current value of the system variable |
| | char | blanks | 26 | 2 | Blank spaces for formatting<br>(Continued on next page) |

| Record | (Contd) Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Source Destination | char | map_labels | 28 | 18 | Label of the source field/element.<br>Name of a system variable that is mapped to the destination field/ element; or current value of the system variable in the destination field/ element followed by a variable number of blanks, followed by a map to sign (->) followed, by a variable number of blanks, followed by the:<br>◗ Label of the destination<br>◗ Field/Element receiving the data<br>◗ Mapped from the source<br>◗ Field/element or name of a system<br>◗ Variable. |
| | char | blanks | 46 | 3 | Blank spaces for formatting |
| Record | char | dest_record | 49 | 7 | Destination record ID |
| | char | blanks | 56 | 2 | Blank spaces for formatting |

(Continued on next page)

### Report Header

| Record | Type | Name | Offset | Length | Description |
|---|---|---|---|---|---|
| Field | char | dest_field | 58 | 3 | Number of the destination field/element receiving the data mapped from the source field/element or system variable |
| | char | blanks | 61 | 5 | Blank spaces for formatting |
| Comment | char | comment | 66 | 11 | One of the following:<br>◗ Blanks = Data mapped to destination field<br>◗ NEW RECORD = New record created from source item<br>◗ Conditional = Conditional mapping with condition not satisfied<br>◗ Part <N1> of <N2> = Value has <N2> parts and this is part <N1> |

# userpref.cfg

## General information

This table contains general information about the *userpref.cfg* file. This file is used only by the visual mapper.

| Information | Details |
|---|---|
| File Type | User preferences list |
| File Name | *userpref.cfg* |
| Record Types | 1 User identifier (EDI-cfgusr, 12 positions)<br>2 The 18 records listed below. |

## Record layouts

This table contains record layout information for *userpref.cfg*.

| Description | Type | Offset | Length |
|---|---|---|---|
| Expert mode (y/n) | char | 0 | 1 |
| Show Navigator start screen (y/n) | char | 1 | 1 |
| Hide to-do (y/n) | char | 2 | 1 |
| Database driver preference | char | 3 | 129 |
| Prompt for database password (y/n) | char | 132 | 1 |
| AN format for IGs | char | 133 | 10 |
| ID format for IGs | char | 143 | 10 |
| N format for IGs | char | 153 | 10 |
| R format for IGs | char | 163 | 10 |
| DT format for IGs | char | 173 | 10 |
| TM format for IGs | char | 183 | 10 |
| IG collapsed on open (y/n) | char | 193 | 1 |
| Use envelope segments (y/n) | char | 194 | 1 |

(Continued on next page)

| (Contd)<br>Description | Type | Offset | Length |
|---|---|---|---|
| AN format for APPs | char | 195 | 10 |
| ID format for APPs | char | 205 | 10 |
| N format for APPs | char | 215 | 10 |
| R format for APPs | char | 225 | 10 |
| DT format for APPs | char | 235 | 10 |
| TM format for APPs | char | 245 | 10 |
| Fixed length (y/n) | char | 255 | 1 |
| Variable length (y/n) | char | 256 | 2 |
| String delimiter | char | 258 | 4 |
| Use string delimiter (y/n) | char | 262 | 1 |
| APP collapsed on open (y/n) | char | 263 | 1 |
| Platform for compile | char | 264 | 15 |
| File transfer program name | char | 279 | 13 |
| Maximum combo box controls | char | 292 | 5 |
| Unused | char | 297 | 215 |

# xmlspl<n>.dat/idx

## General information

The tables listed in this topic are used by the XML option. This table contains general information about *xmlsbl.dat*.

| Information | Details |
|---|---|
| File Type | XML Splitter Configuration tab |
| File Name | *xmlspl<n>.dat*, where <n> indicates the element level |
| Record Types | 1 |

## Record layout xmlspl1.dat

This table contains record layout information for *xmlspl1.dat/idx.* This table is used by the XML option.

| Name | Length | Offset | Description |
|---|---|---|---|
| idname | 65 | 0 | ID |
| elem | 31 | 65 | Element Name |
| namesp | 129 | 96 | Namespace, optional |

## Record layout xmlspl2.dat

This table contains record layout information for *xmlsp2.dat/idx.* This table is used by the XML option.

| Name | Length | Offset | Description |
|---|---|---|---|
| xmltbl2_idname | 65 | 0 | ID (from xmlspl1) |
| idname | 65 | 65 | ID |
| elem | 31 | 130 | Element Name |
| namesp | 129 | 161 | Namespace, optional |

(Continued on next page)

## Record layout xmlspl3.dat

This table contains record layout information for *xmlsp3.dat/idx*. This table is used by the XML option.

| Name | Length | Offset | Description |
|---|---|---|---|
| xmltbl1_idname | 65 | 0 | ID (from xmlspl2) |
| idname1 | 65 | 65 | ID |
| elem | 31 | 130 | Element Name |
| namesp | 129 | 161 | Namespace, optional |

# xmltptbl.dat/idx

## General information

This table contains general information about *xmltptbl.dat.* This table is used by the XML option.

| Information | Details |
|---|---|
| File Type | XML Trading Partnership Rule |
| File Name | *xmltptbl.dat* |
| Record Types | 1 |

## Record layouts for

This table contains record layout information for *xmltptbl.dat*.

| Name | Length | Offset | Description |
|---|---|---|---|
| idname1 | 65 | 0 | Element ID 1 |
| idname2 | 65 | 65 | Element ID 2 |
| idname3 | 65 | 130 | Element ID 3 |
| appname | 65 | 195 | App file name (ddf or dtd) |
| rule1 | 129 | 260 | Rule 1 |
| rule2 | 129 | 389 | Rule 2 |
| rule3 | 129 | 518 | Rule 3 |
| rule4 | 129 | 647 | Rule 4 |
| rule5 | 129 | 776 | Rule 5 |
| rule6 | 129 | 905 | Rule 6 |

# xmlxref.dat/idx

## General information

This table contains general information about *xmltptbl.dat.* This table is used by the XML option.

| Information | Details |
|---|---|
| File Type | XML Trading Partnership Cross Reference |
| File Name | *xmlxref.dat* |
| Record Types | 1 |

## Record layouts

This table contains record layout information for *xmlxref.dat*.

| Name | Length | Offset | Description |
|---|---|---|---|
| str1 | 25 | 0 | Unique string 1 |
| str2 | 25 | 25 | Unique string 2 |
| str3 | 25 | 50 | Unique string 3 |
| str4 | 25 | 75 | Unique string 4 |
| str5 | 25 | 100 | Unique string 5 |
| str6 | 25 | 125 | Unique string 6 |
| tp_code | 16 | 150 | Trading Partnership Code |

# xtable.dat/idx

## General information

This table contains general information about *xtable.dat* and *xtable.idx*.

| Information | Details |
|---|---|
| File Type | Standard Cross Reference Table |
| File Name | *xtable.dat* (*xtable.idx*) |
| Record Types | 1 |

## Record lengths

The record length is 185.

## Record layouts

This table contains record layout information for *xtable.dat* and *xtable.idx*.

| Name | Offset | Length | Description |
|---|---|---|---|
| u_ic_id | 0 | 36 | User's Interchange ID |
| p_ic_id | 36 | 36 | Trading Partner's Interchange ID |
| u_ap_id | 72 | 36 | User's Group/Application ID |
| p_ap_id | 108 | 36 | Trading Partner's Group/Application ID |
| refver1 | 144 | 13 | Reference Number, 1st Part |
| refver2 | 157 | 13 | Reference Number, 2nd Part |
| stdname | 170 | 13 | Standard Name |
| stdtype | 183 | 2 | Standard Type<br>◗ 1 = X12<br>◗ 2 = EDIFACT<br>◗ 3 = TRADACOM |

CHAPTER

5

# Data Type Formats

**Contents**

**Visual Mapper Formats**

# Visual Mapper Formats

## Alphanumeric Formats

**AN, ID**

### Default format

AN: left-justified field with unused positions deleted, code LK

ID: left-justified, blank-filled field, code LB

### Format descriptions

This table describes the field formats for the AN and ID data types.

| Field Format | Description |
|---|---|
| B | Use blanks to fill unused positions. |
| K | Delete unused positions and truncate the field. |
| L | Left-justify the characters. |
| N | Use nulls (for example, hexadecimal 00) to fill unused positions. |
| R | Right-justify the character. |
| V | Delete extra zeros, leaving only one leading zero. |
| X | Remove all leading zeros. |
| Z | Use zeros to fill unused positions. |

# Numeric Formats

## N

### Default format

Right-justified, zero-filled, unsigned integer, code MZRA

**Right-justified fields cannot be null-filled; they can only be blank-filled or zero-filled. If you specify that a field is to be both right-justified and null-filled, Gentran:Server right-justifies and blank-fills the field instead.**

### Format descriptions

This table describes the field formats for the N data type.

| Field Format | Description |
|---|---|
| 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 | Use # of decimals.<br>**Note**<br>The # of Decimals format is available only if you select the User-Defined format (U) for the Decimal Format. |
| A | Do not modify decimal format (as is). |
| B | Use blanks to fill unused positions. |
| C | TCMD (Military usage only). |
| E | Encode value last digit over punch. |
| I | Use implied decimals 0 to 99 decimal places. |
| L | Left-justify. |
| M | Precede negative value with minus sign (-); use no sign for positive value. |
| N | Use nulls (for example, hexadecimal 00) to fill unused positions. |
| P | Precede value with plus (+) or minus (-). |
| R | Right-justify. |
| U | User-defined decimal format. |
| V | Delete extra leading zeros, leaving only one leading zero. |

(Continued on next page)

| (Contd)<br>Field Format | Description |
|---|---|
| W | Use whole number; value is an integer without a decimal point. |
| X | Remove all leading zeros. If value is zero, leave a null. |
| Z | Use zeros to fill unused positions. |

## N0-N9

### Default format

Both the Application Editor and the Standard/IG Editor support field formats N0-N9.

| | |
|---|---|
| Application Editor: | Right-justified, zero-filled, unsigned, implied decimal number (MZRI.*n*, where *n* is 0 for N0; 1 for N1; and 9 for N9) |
| Standard/IG Editor: | Right-justified, zero-filled, unsigned, implied decimal number (MZRI.*n*, where *n* is 0 for N0; 1 for N1; and 9 for N9). |

**Right-justified fields cannot be null-filled; they can only be blank-filled or zero-filled. If you specify that a field is to be both right-justified and null-filled, Gentran:Server right-justifies and blank-fills the field instead.**

### Format descriptions

This table describes the field formats for the N0-N9 data types.

| Field Format | Description |
|---|---|
| 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 | Use # of decimals (0 for N0; 0 or 1 for N1; 0, 1, or 2 for N2; 0, 1, 2, or 3 for N3; and 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9 for N9)<br>**Note**<br>The # of Decimals format is available only if you select the User Defined format (U) for the Decimal Format. For N0-N9 data types, you should use only the Implied Decimal data format. |
| A | Do not modify decimal format (as is). |
| B | Use blanks to fill unused positions. |
| C | TCMD (Military usage only). |

(Continued on next page)

| (Contd)<br>Field Format | Description |
|---|---|
| E | Encode value last digit over punch.<br>**Note**<br>If the value for data with data format E (encoded) ends in a digit instead of an alphabetic character, Gentran:Server assumes it is a positive value and that the digit is a literal part of the value. |
| I | Use implied decimals 0 to 99 decimal places. |
| L | Left-justify. |
| M | Precede negative value with minus sign (-); use nothing for positive value. |
| N | Use nulls (for example, hexadecimal 00) to fill unused positions. |
| P | Precede value with plus (+) or minus (-). |
| R | Right-justify. |
| U | Do not use U (User specified decimal) for N data types. |
| V | Delete extra leading zeros, leaving only one leading zero. |
| W | Use whole number; value is an integer without a decimal point. |
| X | Remove all leading zeros. If value is zero, leave a null. |
| Z | Use zeros to fill unused positions. |

**R** **Default format**

Right-justified, zero-filled, unsigned integer, code MZRA

**Right-justified fields cannot be null-filled; they can only be blank-filled or zero-filled. If you specify that a field is to be both right-justified and null-filled, Gentran:Server will right-justify and blank-fill the field instead.**

(Continued on next page)

### Format descriptions

This table describes the field formats for the R data type.

| Field Format | Description |
|---|---|
| 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 | Use # of decimals.<br>**Note**<br>This format is available only if you select the User Defined format (U) for the Decimal Format. |
| A | Do not modify decimal format (as is). |
| B | Use blanks to fill unused positions. |
| C | TCMD (Military use only). |
| E | Do not use E (encoded) with real data types. |
| I | Do not use I (implied decimal) with real data types. |
| L | Left- justify. |
| M | Precede negative value with minus sign (-); use nothing for positive value. |
| N | Use nulls (for example, hexadecimal 00) to fill unused positions. |
| P | Precede value with plus (+) or minus (-). |
| R | Right-justify. |
| U | User-defined decimal format. |
| V | Delete extra leading zeros, leaving only one leading zero. |
| W | Use whole number; value is an integer without a decimal point. |
| X | Remove all leading zeros. If value is zero, leave a null. |
| Z | Use zeros to fill unused positions. |

(Continued on next page)

## R0-R9

### Default format

Field formats R0-R4 are available in both the Application Editor and the Standard/ IG Editor. Field formats R5-R9 are available in only the Application Editor.

Application Editor: Default numeric format for R0-R9 is a right-justified, zero-filled, unsigned, user-defined decimal number (MZRU.*n*, where *n* is 0 for R0; 1 for R1; and 9 for R9).

Standard/IG Editor: Default numeric format for R0-R4 is a right-justified, zero-filled, unsigned, user-defined decimal number (MZRU.*n*, where *n* is 0 for R0; 1 for R1; and 4 for R4)

**Right-justified fields cannot be null-filled; they can only be blank-filled. Therefore, if you inadvertently specify that a field is to be both right-justified and null-filled, Gentran:Server right-justifies and blank-fills the field instead.**

### Format descriptions

This table describes the field formats for the R0-R9 data types.

| Field format | Description |
|---|---|
| 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 | Use # of decimals (0 for R0; 0 or 1 for R1; 0, 1, or 2 for R2; 0, 1, 2, or 3 for R3; and 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9 for R9).<br>**Note**<br>The # of Decimals format is available only if you select the User Defined format (U) for the Decimal Format. |
| A | Do not modify decimal format (as is). |
| B | Use blanks to fill unused positions. |
| C | TCMD (military use only). |
| E | Do not use format E (encoded) with real data types. |
| I | Contains implied decimals (0 to 99 decimal places). |
| L | Left- justify. |
| M | Precede negative value with minus sign (-); use nothing for positive value. |
| N | Use nulls (for example, hexadecimal 00) to fill unused positions. |
| P | Precede value with plus (+) or minus (-). |
| R | Right-justify. |

(Continued on next page)

| (Contd)<br>Field format | Description |
|---|---|
| U | User specifies the decimal format. |
| V | Delete extra leading zeros, leaving only one leading zero. |
| W | Use whole number; value is an integer without a decimal point. |
| X | Remove all leading zeros. If value is zero, leave a null. |
| Z | Use zeros to fill unused positions. |

## Operating system limitations

Gentran:Server supports all numeric activity, regardless of the number of digits in the value. However, calculations are limited by the number of significant digits that your operating system supports.

The UNIX and Windows operating systems handle with precision only the first 16 digits of a numeric value. This means that these operating systems view only the first 16 digits as significant and consider additional digits to be place holders.

Initially, the Gentran:Server translator sees numeric values as ASCII character strings. A conversion program converts the values to a numeric format. Because of the operating system limitations, the program that converts character strings to numeric formats replaces the digits in the 17th position and higher with zeros.

### Example

If you map from a numeric source and the input value is this 18-character value:

123456789012345678

then the output value is:

123456789012345600

### Suggestion

If the number you map is not used in a calculation, we recommend that you use an alphanumeric data type (AN, for example) instead of a numeric data type. The alphanumeric data type retains the input value of each digit or character.

If the number you map is used in a calculation and you must use a numeric data type, then take into account that the system replaces digits beyond the 16th with zeros. Therefore, the results of calculations are precise only to 16 digits.

# Binary Formats

## Binary data formats

When data in an outbound translation is binary, a path and file name are associated with the field/element containing the data. During translation, Gentran:Server reads the binary file and places the data in the outbound document for transmission.

When data in an inbound translation is binary, **edifrmat** creates a file for that data in the current directory and puts the data into that file, leaving the file name in place of the binary data in the output file. There is a separate file with a unique name for each instance of binary data found in the inbound data.

# Date and Time Formats

Date and time fields take the formats in the following tables. The sample formats are applied to the date and time of 8:30:15 p.m. on Friday, December 31, 1999.

## Key

Letters in the Field Format column represent the following:

- C=Century
- D=Day
- H=Hour
- J=Julian date
- M=Month
- m=Minute
- S=Second
- W=Week
- Y=Year
- Z=Time Zone

## Standard date

This table describes the Standard Date field formats. The examples use the date and time of 8:30:15 p.m. on Friday, December 31, 1999.

| Field Format | Description | Example |
|---|---|---|
| DD | D1. Day (2-digit field) | 31 |
| DD/MM/CCYY | D2. Day/Month/Century year (10-digit field) | 31/12/1999 |
| DD-MM-YY | D3. Day-Month-Year (8-digit field) | 31-12-99 |
| DDMMYY | D4. Day Month Year (6-digit field) | 311299 |
| MM | D5. Month (2-digit field) | 12 |
| MM/DD/YY | D6. Month/Day/Year (8-digit field) | 12/31/99 |
| MMDD | D7. Month Day (4-digit field) | 1231 |
| MMDDYY | D8. Month Day Year (6-digit field) | 123199 |
| WW | D9. Week of the year (2-digit field) | 52 |
| YY | D10. Year (2-digit field) | 99 |

(Continued on next page)

| (Contd)<br>Field Format | Description | Example |
|---|---|---|
| YYMM | D11. Year Month (4-digit field) | 9912 |
| YYMMDD | D12. Year Month Day (6-digit field) | 991231 |
| YYWW | D13. Year Week (4-digit format) | 9952 |
| YYWWD | D14. Year Week Day of week (5-digit format) | 99526 |
| CCYY-MM-DD | D15. Standard database format | 1999-12-3120:30:15 |
| | **Gentran:Server converts to or from the default date format for the database used as the destination or source, using the D15 date format of CCYY/MM/DD as the standard for conversion. Therefore, always use the D15 date format (with 10 positions) for all database applications.** | |
| DD-MMM-YY | D16. Day-Month-Year (9-digit field) | 31-DEC-99 |

### Date and time

This table describes the Date and Time field formats. The examples use the date and time of 8:30:15 p.m. on Friday, December 31, 1999.

| Field Format | Description | Example |
|---|---|---|
| HH-DD-MM-YY | DT1. Hour-Day-Month-Year | 20-31-12-99 |
| mm-HH-DD-MM-YY | DT2. Minute-Hour-Day-Month-Year | 30-20 -31-12-99 |
| YYMMDDHHmm | DT3. Year Month Day Hour Minute | 9912312030 |
| YYMMDDHHmmSS | DT4. Year Month Day Hour Minute Second | 991231203015 |
| YYMMDDHHmmSSZZZ | DT5. Year Month Day Hour Minute Second Time zone | 991231203015EST |
| YYMMDDHHmmZZZ | DT6. Year Month Day Hour Minute Time zone | 9912312030EST |

(Continued on next page)

## Date, time, and century

This table describes the Date, Time, and Century field formats. The examples use the date and time of 8:30:15 p.m. on Friday, December 31, 1999.

| Field Format | Description | Example |
|---|---|---|
| CCYYMMDDHHmm | DTC1. Century year Month Day Hour Minute | 199912312030 |
| CCYYMMDDHHmmSS | DTC2. Century year Month Day Hour Minute Second | 19991231203015 |
| CCYYMMDDHHmmSSZZZ | DTC3. Century year Month Day Hour Minute Second Time zone | 19991231203015EST |
| CCYYMMDDHHmmZZZ | DTC4. Century year Month Day Hour Minute Time zone | 1999123120301530EST |
| HH/DD/MM/CCYY | DTC5. Hour/Day/ Month/ Century year | 20/31/12/1999 |
| mm/HH/DD/MM/CCYY | DTC6. Minute/Hour/ Day/ Month/Century year | 30/20/31/12/1999 |

## Date with century

This table describes the Date with Century field formats. The examples use the date and time of 8:30:15 p.m. on Friday, December 31, 1999.

| Field format | Description | Example |
|---|---|---|
| CC | DC1. Century | 19 |
| CCYY | DC2. Century year | 1999 |
| CCYYMM | DC3. Century year Month | 199912 |
| CCYYMMDD | DC4. Century year Month Day | 19991231 |
| CCYYWW | DC5. Century year Week | 199952 |
| MMDDCCYY | DC6. Month Day Century year | 12311999 |

(Continued on next page)

| (Contd)<br>Field format | Description | Example |
|---|---|---|
| MM/DD/CCYY | DC7. Month/Day/Century year | 12/31/1999 |
| DD-MMM-CCYY | DC8. Day-Month-Century year | 31-DEC-1999 |
| DD.MM.CCYY | DC9. Day.Month.Century year | 31.12.1999 |
| CCYY/MM/DD | DC10. Century year/Month/Day | 1999/12/31 |
| CCYY-MM-DD | DC11. Century year-Month-Day | 1999-12-31 |

## Julian date

This table describes the Julian Date field formats. The examples use the date and time of 8:30:15 p.m. on Friday, December 31, 1999.

| Field Format | Description | Example |
|---|---|---|
| HJJ | DJ1. Hour preceding day as a two-digit integer<br>Where: The letters A-Z, excluding I and O, represent hours on a twenty-four hour clock. | W65 |
| JJJ | DJ2. Day as an integer | 365 |
| YJJJ | DJ3. Last digit of year preceding day as an integer | 9365 |
| YYJJJ | DJ4. Year preceding day as an integer | 99365 |
| JJJY | DJ5. Day as an integer preceding last digit of year | 3659 |
| CCYYJJJ | DJ6. Century and year preceding day as an integer | 1999365 |

## Time only

This table describes the Time Only field formats. The examples use the date and time of 8:30:15 p.m. on Friday, December 31, 1999.

| Field Format | Description | Example |
|---|---|---|
| HHmm | T1. Hour on a 24-hour clock (default) and minute | 2030 |
| HHmmSS | T2. Hour Minute Second | 203015 |
| HHmmSShh | T3. Hour Minute Second Decimal Second | 20301500 |

(Continued on next page)

| (Contd)<br>Field Format | Description | Example |
|---|---|---|
| HHmmSSZZZ | T4. Hour Minute Second Time zone | 203015EDT |
| SS-mm-HH | T5. Second-Minute-Hour | 15-30-20 |
| SS/mm/HH | T6. Second/Minute/Hour | 15/30/20 |
| HH:mm | T7. Hour:Minute | 20:30 |

# Database Data Types

This table correlates the data types from each of the supported databases to the equivalent Gentran:Server data type.

**Note**
The scale and precision of Gentran:Server values is determined by the native database for data types N or R.

| Database | Database Data Type | Gentran:Server Data Type |
|---|---|---|
| Oracle<br>**Note**: Support for Oracle databases does not include support for the Oracle Exadata platform. | Char | ID |
| | Date | DT |
| | Long Raw | B |
| | Number | N, R |
| | Varchar2 | AN |
| Sybase | Char | ID |
| | DateTime | DT |
| | Image | B |
| | Int | N |
| | Numeric | N, R |
| | Varchar | AN |
| TextFile | Date | DT |
| | Numeric | N,R |
| | Varchar | AN |

CHAPTER
6

# Example Record Layout Files

**Contents**

# Overview

## Definition of record layout file

A **record layout file** is a flat file you create outside of Gentran:Server to contain the attributes for the records and fields in your application data. The application data described by this record layout file is also in the form of a flat file.

## Purpose of a record layout file

You can use a record layout file to create an application description. An application description describes the format of data in the documents you exchange with your trading partner. You use an application description as a source or destination document when you create your translation maps.

## In this chapter

This chapter describes only those record layout files used to create an application description for the Visual Mapper.

### References

You can also create an application description manually in Gentran:Server. See the Gentran:Server *for UNIX and Workstation Mapping and Translation Guide* for information.

This chapter has four main sections.

| Section | Description |
|---|---|
| Layout File Rules | The rules for creating record layout files |
| Example Data: Overview | An overview of the data used in the examples contained in this chapter |

(Continued on next page)

| (Contd)<br>Section | Description |
|---|---|
| Fixed-Length Data Example | An example of the record layout file used to describe fixed-length application data<br>This example displays:<br>◗ The fixed-length application data<br>◗ A table showing the attributes of the records and fields in the application data<br>◗ The record layout file<br>◗ The dialog boxes you complete to read in the record layout file<br>◗ The Application Editor screen displaying the application description created from the file |
| Variable Length Data Example | An example of the record file layout file used to describe variable-length application data<br>This example displays:<br>◗ The variable-length application data,<br>◗ A table showing the attributes of the records and fields in the application data<br>◗ The record layout file<br>◗ The dialog boxes you complete to read in the record layout file<br>◗ The Application Editor screen displaying the application description created from the file |

# Layout File Rules

## How delimiters are used

Record layout files use **delimiters** to separate field attributes.

Any flat file containing application data has lines of data. The use of delimiters depends on whether the data is fixed-length or variable-length.

### Fixed-length data

In a flat file containing fixed-length application data fields, spaces or zeros fill out fields that are not full of data. The fields are <u>not</u> separated by delimiters.

### Variable-length data

In a flat file containing variable-length application data fields, delimiters indicate field boundaries. Space padding and zeros padding are <u>not</u> used.

## Delimiter rules

The following rules for delimiters apply to record layout files.

- You must use a delimiter between attributes in a record layout file. This is because there is no file to describe the length of fields in the record layout file.
- You can use any one of the following delimiters to separate field attributes in your record layout file:
  - A comma (,)
  - A blank
  - An asterisk (*).

  Do not select a delimiter that appears in any of the field attributes. This means you cannot use the default delimiter (a blank) if you have spaces in the field names.
- You must specify the delimiter for the application data in the User Setup dialog box available under the Settings menu, not in the record layout file.
- You can use only one delimiter character in a record layout file, as in your application data.

(Continued on next page)

## Fixed-length field attributes

The following attributes apply to fixed-length fields:

- Fixed-length application data has a fixed-length attribute. No minimum or maximum length attributes apply.
- The attributes you can specify for fields in fixed-length application data are as follows:
  - Data type
  - Field name
  - Field length
  - Other attribute.
  - Required flag
  - Retain flag

  You can specify the attributes in any order. Use the Read Application Data Description Layout dialog box to specify the order.

## Variable-length field attributes

The following attributes apply to variable-length fields:

- Variable-length application data has a minimum length attribute and a maximum length attribute, but no fixed-length attribute.
- The attributes you can specify for fields in variable-length application data are as follows:
  - Data type
  - Field name
  - Minimum length
  - Maximum length
  - Other attribute.
  - Required flag
  - Retain flag

  You can specify the attributes in any order. Use the Read Application Data Description Layout dialog box to specify the order.

(Continued on next page)

### Record attributes for variable and fixed-length files

- The attributes you can enter for a record and their required order are as follows:
  - — RECORD
  - — Record name
  - — Record ID
  - — Required flag
  - — Maximum occurrence.
- When entering records and their attributes, you can stop at any point after ‘RECORD,’ but you cannot include attributes that appear after skipped attributes. For example, any of the following are allowable when you use a comma as the delimiter:

  RECORD,<record name>,<record ID>,<req flag>,<max occurrence>

  RECORD,<record name>,<record ID>,<req flag>

  RECORD,<record name>,<record ID>

  RECORD,<record name>

  RECORD

# Example Data: Overview

## Introduction

Both examples on the following pages use the same basic data, but the first example displays the data in fixed-length form and the second example displays the data in variable-length form.

The sample data is based on an invoice application.

## Invoice information

This table shows the information for the invoice.

| **Invoice Information** | |
|---|---|
| Invoice Sender’s Information: | |
| Sender’s name | Toyz Galore Incorporated |
| Address | 57 Eggplant Avenue |
| | Gotham City, MI 48999 |
| Dun’s number | 840223591429 |
| Department number | 3025 |
| Invoice number | BD000000095000034287 |
| Invoice date | To be provided |
| EDI standard | X12 2040 |
| EDI transaction set | 810 |
| **Invoice receiver’s information:** | |
| Receiver’s name | Big Deal Enterprises |
| Receiver’s address | 4444 Shamrock Place |
| | Green Valley, MI 48999 |
| Receiver’s Dun’s number | 468129450384 |
| Department number | 15 |
| Receiver’s Trading Partnership Code | BigDealinvoice |

(Continued on next page)

| **(Contd)**<br>**Invoice Information** | |
|---|---|
| Purchase order number | 12486395 |
| Purchase order date | 7/25/95 |
| Purchase order items | |
| ◗ Baby Goes with You (item # X2I5448399), $4.53 per each doll, 1500 ordered | |
| ◗ Swim and Sun Pool in blue with white fish (item # Z9L8234771), $3.97 per each pool, 2000 ordered | |
| ◗ Boys 24-inch Bike in red with blue trim (item # N4F8263046), $24.00 per bike, 900 ordered | |
| **Other Important Information**<br>Big Deal Enterprises is a special priority customer of Toyz Galore Incorporated, and their orders are shipped the fastest way possible. | |

### Field and record layouts

The field attribute layouts for the examples are shown below.

- Fixed-length field attribute layout:

  ```
  <field name>,<field length>,<data type>,<req
  flag>,<retain flag>,<other attrib>
  ```

- Variable-length field attributes layout:

  ```
  <field name>,<min length>,<max length>,<data type>,<req
  flag>,<retain flag>,<other attrib>
  ```

  **Note**
  Both field layout files use a comma (,) as the delimiter.

- Record attribute layout for both fixed and variable examples:

  ```
  RECORD,<record name>, <record ID>, <req flag>, <max
  occurrence>
  ```

  **Note**
  The record attribute layout also uses a comma (,) as the delimiter.

# Fixed-Length Data Example

## Overview

This example illustrates application data with fixed-length fields. Each fixed-length field has one length, and the value cannot exceed that length. If the value is less than that length, spaces must fill out the rest of the field.

This example contains:

- Sample fixed-length application data.
- Attributes of the records and fields in the fixed-length application data.
- Example of the record layout file.
- Procedure for reading in the fixed-length record layout file and the application description created from the record layout file.

## Application data

This is the example fixed-length application data:

```
BigDealinvoice  HDR BD00000009500003428725-07-950000000000
0000000012486395
BigDealinvoice NTE Special priority customer  Ship fastest way possible
BigDealinvoice  NAM Toyz Galore Incorporated     440-555-4 4550000
BigDealinvoice  ADR 57 Eggplant Avenue
BigDealinvoice  AD2 Gotham City         MI 48999
BigDealinvoice  REF 840223591429
BigDealinvoice  REF 000000003025
BigDealinvoice  NAM Big Deal Enterprises         899-555-2 0000000
BigDealinvoice  ADR 4444 Shamrock Place
BigDealinvoice  AD2 Green Valley        MI 48999
BigDealinvoice  REF 468129450384
BigDealinvoice  REF 000000000015
BigDealinvoice  DTL X2L5448399                 1500each  0453Catalog
BigDealinvoice  DTL Z9L8234771                 2000each  0397Catalog
BigDealinvoice  DTL N4F8263046                 0900each  2400Catalog
BigDealinvoice  DES Baby Goes with You
BigDealinvoice  DES Swim and Sun Pool in blue with white fish
BigDealinvoice  DES Boys 24-inch Bike in red with blue trim
BigDealinvoice  DAT 06/23/95
BigDealinvoice  TRL 0000000000000000003
BigDealinvoice  TRL 0000000000003633500
```

(Continued on next page)

## Record and field attributes

The attributes for the records and fields in the fixed-length application data are shown in the following table:

| Name | Rec ID | Max Occ | Fixed Length | Data Type | Req | Retain |
|---|---|---|---|---|---|---|
| Header | HDR | 1 | | | M | |
| TP | | | 16 | AN | O | N |
| Rec ID | | | 4 | AN | O | N |
| Inv Num | | | 20 | AN | O | N |
| PO Date | | | 8 | DT | O | N |
| PO Num | | | 20 | AN | O | N |
| Note | NTE | 500 | | | O | |
| TP | | | 16 | AN | O | N |
| Rec ID | | | 4 | AN | O | N |
| Note 1 | | | 28 | AN | O | N |
| Note 2 | | | 28 | AN | O | N |
| Name | NAM | 2 | | | O | |
| TP | | | 16 | AN | O | N |
| Rec ID | | | 4 | AN | O | N |
| Name | | | 28 | AN | O | N |
| ID Code | | | 16 | N | O | N |
| Address | ADR | 2 | | | O | |
| TP | | | 16 | AN | O | N |
| Rec ID | | | 4 | AN | O | N |
| Address | | | 32 | AN | O | N |
| Address2 | AD2 | 2 | | | O | |
| TP | | | 16 | AN | O | N |
| Rec ID | | | 4 | AN | O | N |
| City | | | 20 | AN | O | N |

(Continued on next page)

| (Contd) Name | Rec ID | Max Occ | Fixed Length | Data Type | Req | Retain |
|---|---|---|---|---|---|---|
| State | | | 3 | AN | O | N |
| Zip | | | 9 | AN | O | N |
| Reference | REF | 2 | | | O | |
| TP | | | 16 | AN | O | N |
| Rec ID | | | 4 | AN | O | N |
| Dun’s No | | | 12 | AN | O | N |
| Detail | DTL | 1000 | | | O | |
| TP | | | 16 | AN | O | N |
| Rec ID | | | 4 | AN | O | N |
| Part No | | | 30 | AN | O | N |
| Quantity | | | 4 | N | O | N |
| Units | | | 6 | AN | O | N |
| Price | | | 6 | R | O | N |
| Price Cd | | | 12 | AN | O | N |
| Discount | | | 4 | R | O | N |
| Discount | | | 4 | R | O | N |

## Record layout file

The delimited record layout file for the example application data is shown in the following illustration:

```
RECORD,Header,HDR,0,0001
TP,16,AN,O,N
Rec ID,4,AN,O,N
Inv Num,20,AN,O,N
PO Date,8,DT,O,N
PO Num,20,AN,O,N
RECORD,Note,NTE,0,0500
TP,16,AN,O,N
Rec ID,4,AN,O,N
Note 1,28,AN,O,N
Note 2,28,AN,O,N
```

(Continued on next page)

```
RECORD,Name,NAM,O,0002
TP,16,AN,O,N
Rec ID,4,AN,O,N
Name,28,AN,O,N
ID Code,16,N,O,N
RECORD,Address,ADR,O,0002
TP,16,AN,O,N
Rec ID,4,AN,O,N
Address,32,AN,O,N
RECORD,Address2,AD2,O,0002
TP,16,AN,O,N
Rec ID,4,AN,O,N
City,20,AN,O,N
State,3,AN,O,N
Zip,9,AN,O,N
RECORD,Reference,REF,O,0002
TP,16,AN,O,N
Rec ID,4,AN,O,N
Dun's No,12,AN,O,N
RECORD,Detail,DTL,O,1000
TP,16,AN,O,N
Rec ID,4,AN,O,N
Part No,30,AN,O,N
Quantity,4,N,O,N
Units,6,AN,O,N
Price,6,R,O,N
Price Cd,12,AN,O,N
Discount,4,R,O,N
```

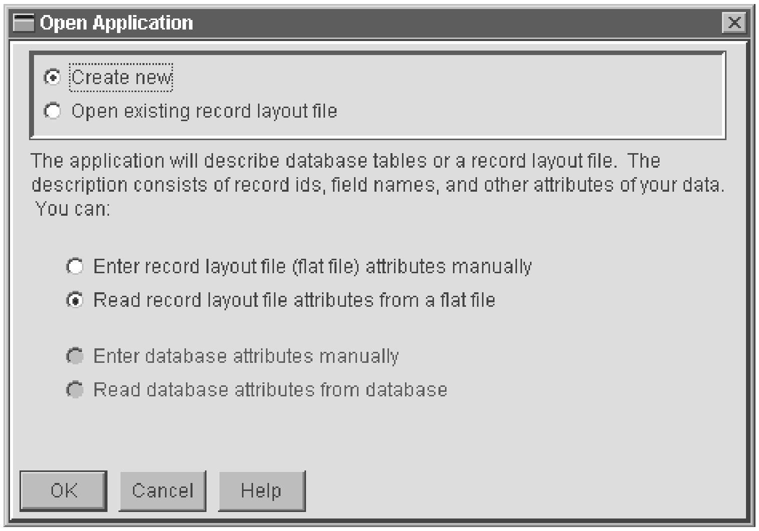
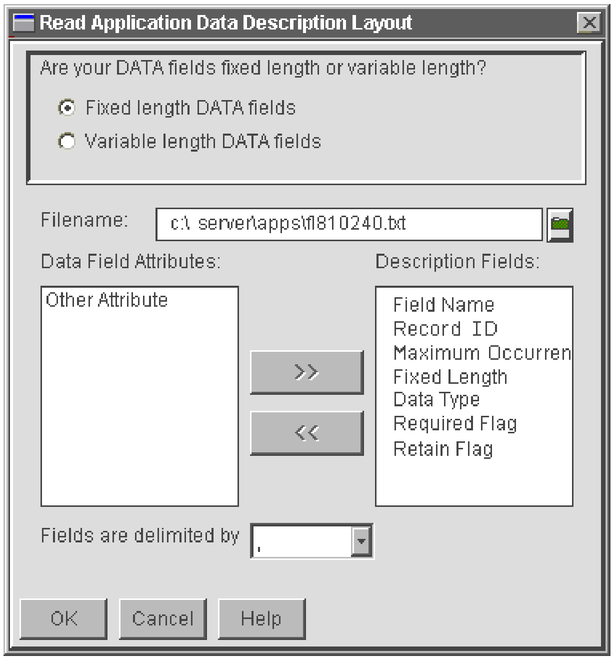
## Reading the file into Gentran:Server

The file in this example is named *fl810240.txt*. Reading this file into Gentran:Server creates the application description for the fixed-length application data in Gentran:Server.

Use this procedure to read in the record layout file.

| Step | Action |
|---|---|
| 1 | Open Gentran:Server and then open the Application Editor by clicking on the **App** button.<br>**System Response**<br>The system displays the Open Application dialog box. |

(Continued on next page)

| (Contd) | Action |
|---|---|
| 2 | Complete the fields as shown in the following illustration. These are the selections for reading in record layout files.<br><br> <br>● Create new<br>○ Open existing record layout file<br>The application will describe database tables or a record layout file. The description consists of record ids, field names, and other attributes of your data. You can:<br>○ Enter record layout file (flat file) attributes manually<br>● Read record layout file attributes from a flat file<br>○ Enter database attributes manually<br>○ Read database attributes from database<br> |
| 3 | Click **OK**.<br>**System Response**<br>The system displays the Read Application Data Description Layout dialog box.<br><br> <br>Are your DATA fields fixed length or variable length?<br>● Fixed length DATA fields<br>○ Variable length DATA fields<br>Filename: c:\ server\apps\fl810240.txt<br>Data Field Attributes: Other Attribute<br>Description Fields: Field Name, Record ID, Maximum Occurren, Fixed Length, Data Type, Required Flag, Retain Flag<br>Fields are delimited by ,<br> <br>(Continued on next page) |

<table>
<tr><th>(Contd)</th><th>Action</th></tr>
<tr><td>4</td><td>

Complete the Read Application Data Description Layout dialog box as shown in the following illustration. These are the selections for reading in the fixed-length record layout file.

**Important**

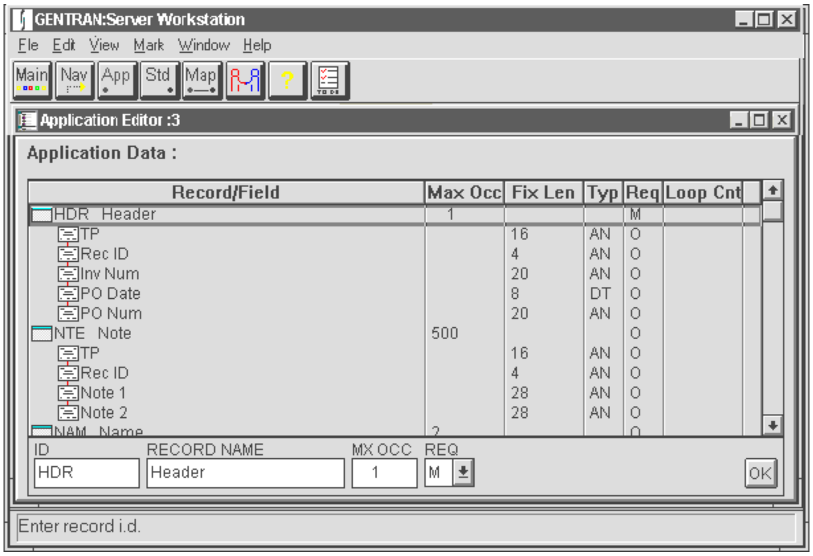
You must select the attributes in the same order as they appear in the record layout file.

</td></tr>
<tr><td>5</td><td>

Click **OK**.

**System Response**<br>
Gentran:Server creates an application description from the fixed-length record layout file.

**Example**<br>
This is an application description created from the fixed-length record layout file that has been read into the Application Editor.



Application Data :

| Record/Field | Max Occ | Fix Len | Typ | Req | Loop Cnt |
|---|---|---|---|---|---|
| HDR Header | 1 | | | M | |
| TP | | 16 | AN | O | |
| Rec ID | | 4 | AN | O | |
| Inv Num | | 20 | AN | O | |
| PO Date | | 8 | DT | O | |
| PO Num | | 20 | AN | O | |
| NTE Note | 500 | | | O | |
| TP | | 16 | AN | O | |
| Rec ID | | 4 | AN | O | |
| Note 1 | | 28 | AN | O | |
| Note 2 | | 28 | AN | O | |
| NAM Name | 2 | | | O | |

| ID | RECORD NAME | MX OCC | REQ |
|---|---|---|---|
| HDR | Header | 1 | M |

Enter record i.d.

</td></tr>
</table>

# Variable Length Data Example

## Overview

This example displays application data with variable-length fields. The length of each field varies according to the length of the value of the field; however, every field has a minimum and a maximum length.

This example contains:

- Example variable-length application data.
- A table displaying the attributes of the records and fields in the variable-length application data.
- Example of the record layout file.
- Procedure for reading in the variable-length record layout file and the application description created from the file.

## Application data

This example application data has an asterisk (*) for a field delimiter and a new line for a record terminator.

```
BigDealinvoice*HDR*BD000000095000034287*25-07-95*00000000
000000000012486395
BigDealinvoice*NTE*Special priority customer*Ship fastest way
possible
BigDealinvoice*NAM*Toyz Galore Incorporated*440-555-44550 000
BigDealinvoice*ADR*57 Eggplant Avenue
BigDealinvoice*AD2*Gotham City*MI*48999
BigDealinvoice*REF*840223591429
BigDealinvoice*REF*000000003025
BigDealinvoice*NAM*Big Deal Enterprises*899-555-20000000
BigDealinvoice*ADR*4444 Shamrock Place
BigDealinvoice*AD2*Green Valley*MI*48999
BigDealinvoice*REF*468129450384
BigDealinvoice*REF*000000000015
BigDealinvoice*PER*John Jay Doe*899-555-2039
BigDealinvoice*DTL*X2L5448399*1500*each*0453*Catalog
BigDealinvoice*DTL*Z9L8234771*2000*each*0397*Catalog
BigDealinvoice*DTL*N4F8263046*0900*each*2400*Catalog
BigDealinvoice*DES*Baby Goes with You
BigDealinvoice*DES*Swim and Sun Pool in blue with white fish
BigDealinvoice*DES*Boys 24-inch Bike in red with blue trim
BigDealinvoice*DAT*06/23/95
BigDealinvoice*TRL*0000000000000000003
BigDealinvoice*TRL*0000000000003633500
```

(Continued on next page)

## Record and field attributes

This table shows the attributes for the records and fields in the above variable-length application data:

| Name | Rec ID | Max Occ | Min Len | Max Len | Data Type | Req | Retain |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| Header | HDR | 1 | | | | M | |
| TP | | | 1 | 16 | AN | O | N |
| Rec ID | | | 2 | 4 | AN | O | N |
| Inv Num | | | 1 | 20 | AN | O | N |
| PO Date | | | 2 | 8 | DT | O | N |
| PO Num | | | 1 | 20 | AN | O | N |
| Note | NTE | 500 | | | | O | |
| TP | | | 1 | 16 | AN | O | N |
| Rec ID | | | 2 | 4 | AN | O | N |
| Note 1 | | | 0 | 28 | AN | O | N |
| Note 2 | | | 0 | 28 | AN | O | N |
| Name | NAM | 2 | | | | O | |
| TP | | | 1 | 16 | AN | O | N |
| Rec ID | | | 2 | 4 | AN | O | N |
| Name | | | 1 | 28 | AN | O | N |
| ID Code | | | 1 | 16 | N | O | N |
| Address | ADR | 2 | | | | O | |
| TP | | | 1 | 16 | AN | O | N |
| Rec ID | | | 2 | 4 | AN | O | N |
| Address | | | 2 | 32 | AN | O | N |
| Address2 | AD2 | 2 | | | | O | |
| TP | | | 1 | 16 | AN | O | N |
| Rec ID | | | 2 | 4 | AN | O | N |

(Continued on next page)

| (Contd) Name | Rec ID | Max Occ | Min Len | Max Len | Data Type | Req | Retain |
|---|---|---|---|---|---|---|---|
| City | | | 1 | 20 | AN | O | N |
| State | | | 1 | 3 | AN | O | N |
| Zip | | | 5 | 9 | AN | O | N |
| Reference | REF | 2 | | | | O | |
| TP | | | 1 | 16 | AN | O | N |
| Rec ID | | | 2 | 4 | AN | O | N |
| Dun’s No | | | 12 | 12 | AN | O | N |
| Detail | DTL | 1000 | | | | O | |
| TP | | | 1 | 16 | AN | O | N |
| Rec ID | | | 2 | 4 | AN | O | N |
| Part No | | | 2 | 30 | AN | O | N |
| Quantity | | | 1 | 4 | N | O | N |
| Units | | | 1 | 6 | AN | O | N |
| Price | | | 6 | 6 | R | O | N |
| Price Cd | | | 1 | 12 | AN | O | |
| Discount | | | 4 | 4 | R | O | |

## Record layout file

This is the delimited record layout file for the example application data:

```
RECORD,Header,HDR,0,1
TP,1,16,AN,O,N
Rec ID,2,4,AN,O,N
Inv Num,1,20,AN,O,N
PO Date,2,8,DT,O,N
PO Num,1,20,AN,O,N
RECORD,Note,NTE,0,500
TP,1,16,AN,O,N
Rec ID,4,4,AN,O,N
Note 1,0,28,AN,O,N
Note 2,0,28,AN,O,N
RECORD,Name,NAM,0,2
TP,1,16,AN,O,N
Rec ID,2,4,AN,O,N
```

(Continued on next page)

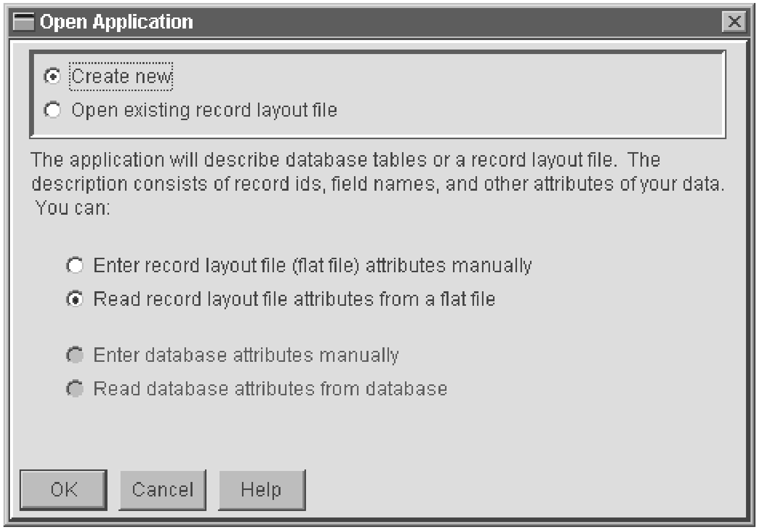
```
Name,1,28,AN,O,N
ID Code,1,16,N,O,N
RECORD,Address,ADR,O,2
TP,1,16,AN,O,N
Rec ID,2,4,AN,O,N
Address,2,32,AN,O,N
RECORD,Address2,AD2,O,2
TP,1,16,AN,O,N
Rec ID,2,4,AN,O,N
City,1,20,AN,O,N
State,1,3,AN,O,N
Zip,5,9,AN,O,N
RECORD,Reference,REF,O,2
TP,1,16,AN,O,N
Rec ID,2,4,AN,O,N
Dun's No,12,12,AN,O,N
RECORD,Detail,DTL,O,1000
TP,1,16,AN,O,N
Rec ID,2,4,AN,O,N
Part No,2,30,AN,O,N
Quantity,1,4,N,O,N
Units,1,6,AN,O,N
Price,6,6,R,O,N
Price Cd,1,12,AN,O,N
Discount,4,4,R,O,N
```

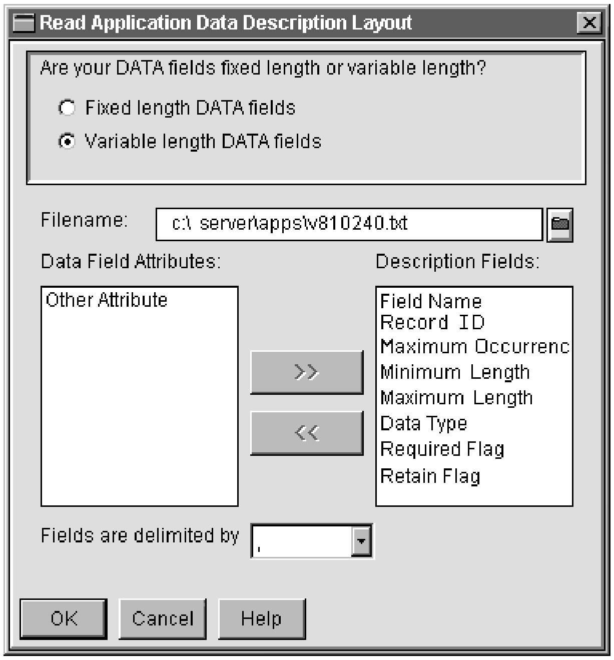
## Reading the file into Gentran:Server

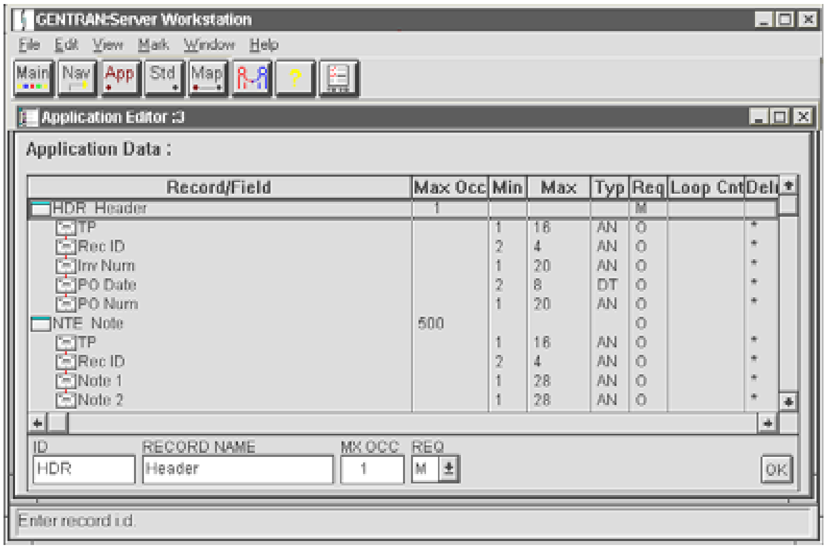
This file in this example is named *v810240.txt*. Reading this file into Gentran:Server creates the application description for the variable-length application data in Gentran:Server.

Use this procedure to read in the record layout file.

| Step | Action |
|---|---|
| 1 | Open Gentran:Server and then open the Application Editor by clicking on the **App** button.<br>**System Response**<br>The system displays the Open Application dialog box. |
| 2 | Complete the fields as shown in the following illustration.<br>(Continued on next page) |

| (Contd) | Action |
|---|---|
| | These are the selections for reading in record layout files.<br><br>Open Application<br>● Create new<br>○ Open existing record layout file<br>The application will describe database tables or a record layout file. The description consists of record ids, field names, and other attributes of your data. You can:<br>○ Enter record layout file (flat file) attributes manually<br>● Read record layout file attributes from a flat file<br>○ Enter database attributes manually<br>○ Read database attributes from database<br> <br>**Comment**<br>The selections on the Open Application dialog box are the same for fixed-length applications and variable-length applications. |
| 3 | Click **OK**.<br>**System Response**<br>The system displays the Read Application Data Description Layout dialog box.<br>(Continued on next page) |

| (Contd) | Action |
|---|---|
| 4 | Complete the Read Application Data Description Layout dialog box as shown below to read in the variable-length record layout file. Select the attributes in the same order as they appear in the application data.<br><br>Read Application Data Description Layout<br>Are your DATA fields fixed length or variable length?<br>○ Fixed length DATA fields<br>● Variable length DATA fields<br>Filename: c:\ server\apps\v810240.txt<br>Data Field Attributes: Other Attribute<br>Description Fields: Field Name, Record ID, Maximum Occurrenc, Minimum Length, Maximum Length, Data Type, Required Flag, Retain Flag<br>Fields are delimited by ,<br> |
| 5 | Click **OK**.<br>**System Response**<br>Gentran:Server creates an application description from the variable-length record layout file.<br>(Continued on next page) |

| (Contd) | Action |
|---|---|
| | **Example**<br>This is the application description created from the variable-length record layout file that has been read into the Application Editor.<br> |

| Record/Field | Max Occ | Min | Max | Typ | Req | Loop Cnt | Deli |
|---|---|---|---|---|---|---|---|
| HDR Header | 1 | | | | M | | |
| TP | | 1 | 16 | AN | O | | * |
| Rec ID | | 2 | 4 | AN | O | | * |
| Inv Num | | 1 | 20 | AN | O | | * |
| PO Date | | 2 | 8 | DT | O | | * |
| PO Num | | 1 | 20 | AN | O | | * |
| NTE Note | 500 | | | | O | | |
| TP | | 1 | 16 | AN | O | | * |
| Rec ID | | 2 | 4 | AN | O | | * |
| Note 1 | | 1 | 28 | AN | O | | * |
| Note 2 | | 1 | 28 | AN | O | | * |

ID: HDR
RECORD NAME: Header
MX OCC: 1
REQ: M

**Note**

The delimiter is the asterisk (set in the Preferences dialog box).

# Index