ORACLE
INSURANCE

Using Documerge

Version 3.2.2
MVS VSE

Revised 11/17/09

Using Documerge MVS VSE

Version 3.22

Part Number: E16444-01

November 2009

Contents

Preface

The *Using Documerge* reference presents detailed, technical information about Documerge. This reference contains information such as descriptions of Documerge programs, JCL examples, control cards and syntax, and dataset requirements. *Using Documerge* is for readers who understand Documerge and are familiar with the other Documerge documentation.

If you have no experience with Documerge, study the *Documerge Concepts guide* to understand the main ideas before you proceed with the technical details. Experienced Documerge users can also refer to *Documerge Concepts* to reinforce their knowledge.

Documerge uses many programs and utilities that are shared among Oracle products. These programs and utilities are documented in this reference as their use pertains to Documerge processing. This reference does not include documentation of non-Oracle products. Please refer to vendor-supplied documentation for products not supplied by Oracle.

The Documerge 3.2 documentation suite consists of this reference and the following books:

- *Composing Forms for Documerge*
- *Documerge Concepts*
- *Documerge Error Messages*
- *Implementing Documerge*
- *Installing Documerge*
- *Migrating Documerge*
- *Troubleshooting Documerge*

Also part of the Documerge 3.2 library, but not dedicated solely to Documerge, are the following books:

- *DCF/PLUS Support for Xerox Highlight Color Printers*
- *Using VLAM reference*

Documerge Highlights

Documerge 3.2 Compatibility with Earlier Versions

Documerge 3.2 is fully compatible with previous 3.x and 2.1 versions of Documerge. No file conversions are required. You can use your existing EDL and Rulebase. However, some new features require VDR or JCL changes (for example, VDRs that called DMGVLAM directly must be changed to call VLMSRVR). Refer to the *Migrating Documerge* for more information.

IMPORTANT!

We no longer support Documerge 1.7 or 2.0.

We do provide aids for migrating from version 1.7 to 2.0; and for migrating from 2.0 to 3.x. There is no migration directly from 1.7 to 3.x, so you must at least upgrade to 2.0 (upgrading to Documerge 2.1 is highly recommended) before installing a 3.x version of Documerge.

Dynacomp Version 1.0.3 and earlier are *not* supported for Documerge 3.x. Dynacomp 1.0.4, however, supports both Documerge 3.x and 2.1.

New Features for Documerge 3.2 Level 2

- **New VDR EXEC PARM: RBOPEN=**

  The format of the parameter is:

  RBOPEN=*value*

  Valid values are:

Value	Explanation
OPENLIB	This is the default.
OPENLIBA	If your VDR calls VLMSRVR with the OPENLIBA command, then specify **RBLIB=OPENLIBA** in your VDR EXEC PARM. Example: `//VDRSTEP EXEC PGM=MYVDR,PARM='RBOPEN=OPENLIBA'` You may specify this with other VDR EXEC PARMS. Example:

```
//VDRSTEP  EXEC PGM=MYVDR,PARM='VLMACCESS=RO,VLMCONTROL=KEEP,RBOPEN=OPENLIBA'
```

- **New DMGMERGE EXEC PARM: EDTBDT=**

  This parameter affects AFP output. The format for this parameter is:

  EDTBDT=*value*

  Valid values are:

Value	Explanation
Y or YES	Normally DMGMERGE will issue an IMM to force a new sheet of paper. A value of **Y** or **YES** causes DMGMERGE to issue an EDT/BDT sequence instead. Use this for post processors that do not recognize an IMM as forcing a new sheet of paper.
N or No	This is the default. It causes DMGMERGE to issue an IMM to force a new sheet of paper.

New Features for Documerge 3.2 Level 1

- **Keep any INVERT DJDE parameters that exists in EDL forms with the new STRIPINVERT parameter for the GLOBAL command.**

  This parameter is for Metacode users only. The format of this parameter is

  STRIPINVERT=*value*

  where ***value*** is the placeholder for one of the following values:

Value	Explanation
Y or YES	Strip (remove / ignore) any INVERT parameters in DJDE records in EDL forms. *This is the default.*
N or NO	Keep any INVERT parameters in DJDE records in EDL forms.

- **Perform right-justification of the DMG.SET.NUMBER tag with the new RTJUSTIFY parameter for the MERGE command.**

  The format of this parameter is

  RTJUSTIFY=DMG.SET.NUMBER

  Specify this on any MERGE command to cause BPSDs that use the DMG.SET.NUMBER tag to be right-justified with this BPSD area. Remember that right-justification is on a character basis and generally requires a fixed-pitch font for desired results.

- **Use bit (dash code) processing with the DMG.SET.NUMBER tag.**

  Use tag names “DMG.SET.NUMBER.BIT.x” where “x” is a value from 1 to 99. See “Reserved Tags and Dash Codes” in the *Documerge Reference Guide* for more information on bit processing.

- **List reserved tag names in TAG parameters in the DMGMERGE SYSIN.**

  To allow this, use the following new GLOBAL parameter:

  ```
  TAGPARMRN=Y
  -or-
  TAGPARMRN=YES
  ```

  The default value for this new TAGPARMRN option is “N” (or No), meaning reserved tag names are ignored if placed in DMGMERGE SYSIN TAG parameters (as is the case with Documerge 3.2.0). The following reserved tags are affect by this:

  - DMG.MISSING.FORMS
  - DMG.VDR.ERRORS
  - DMG.DD.groupname
  - DMG.ERDD.groupname
  - DMG.ERROR.VRF
  - DMG.GCOPY.groupname
  - DMG.MDEF.groupname
  - DMG.FDEF.groupname

  Specifically, this feature was added to allow dynamic building of DMG.DD.groupname tag values using Command Tag processing.

New Features for Documerge 3.2 Level 0

Documerge 3.2 Level 0 offers the following new features and enhancements:

- Speed improvements for use with IBM zSeries machines. Documerge 3.2 programs have been modified to run faster on the IBM zSeries machines.
- Optimization of Metacode output (most effective when used with portrait imposition and logical pages features). You can optimize DMGMERGE Metacode output — combining records into one larger record when possible. This can result in faster print, eliminating the Xerox message "Output has caught up with input."

  For details, see "OPTIMIZE=" on page 413.
- Selection of input trays for banner and trailer forms. DMGMERGE allows input tray selection for user banner and trailer forms.

  For details, see "BANNERFEED=" on page 408, and "TRAILERFEED=" on page 441.
- Implementation of the TAG= parameter for the GLOBAL and FILEDEF commands. You can now code TAG= paramameters for the GLOBAL and FILEDEF commands, similar to the current TAG= command for the MERGE command.

  For details, see "TAG=" on page 400, and "TAG=" on page 416.
- Online distribution (email) of Trouble Ticket fixes. When possible, Oracle will email Trouble Ticket fixes, reducing the time needed to distribute them.

  For details, see "Installing Documerge Trouble Ticket Fixes" on page 481.

Overview of Documerge

Documerge is a mainframe-based software product that assembles complete Document Packages. This is based upon:

- Variable data from an application system.
- User defined printing and collation rules.
- User defined printer hardware and software specifications.

This chapter defines Documerge terminology and discusses 31-bit addressing in Documerge. This chapter also describes briefly the main components of Documerge.

- Electronic Document Library (EDL)
- Printer Environment Library (PELIB)
- Composition Preprocessors
- Normalizers
- Rulebase Library
- Variable Data Reformatter (VDR)
- Variable Replacement File (VRF)
- Merging Process (DMGMERGE)

Documerge Terminology

Term	Meaning
Banner Sheet	The first form in a Documerge output file.
Boilerplate	The print-ready data stream for a form. Can contain locations reserved for variable data (Boilerplate Spaces).
Boilerplate Index Packet	A group of records placed at the beginning of a print-ready form (Boilerplate). Used by the DMGMERGE program for mapping variable data to their assigned locations (Boilerplate Spaces) in the form.
Boilerplate Space	A location reserved for variable data in a print-ready form (Boilerplate). Can be defined both within and outside fixed text.
Normalization	The preparation of a print-ready data stream (except line printers) for use by Documerge. Does not affect the appearance of the printed document. Documerge includes two Normalization programs: ▪ DMGAFPP for forms printed on AFP printers ▪ DMGMETP for forms printed on Metacode printers.
BPSD	**Boilerplate Space Definition**. Reserves a location for variable data (Boilerplate Space) in a print-ready form (Boilerplate). The BPSD is a composition command developed by Oracle. It is not a part of any composition system; rather, you can use the BPSD with different composition systems. (Refer to "Documerge Composition Preprocessors" on page 31 for more information.) BPSD syntax varies with each composition system, but the abbreviation BPSD remains the same. Using the appropriate BPSD syntax, you define the characteristics of each Boilerplate Space, such as the name, length, and optional parameters. Refer to *Composing Forms for Documerge* for more information.
Chain	An independent set of data that is related to a form. Various types of chains can exist for the same form, such as a chain containing storage data and one containing the composition source file. Also, a chain can be a version of the form in a unique format, such as a Metacode version or an AFP version. Refer to *Using VLAM* for more information.
Common Font List	A list of fonts used by all forms in a Document Package.
Document Package	A set of print-ready forms, completed and collated by the DMGMERGE program, for one recipient (Group). Sent to a Documerge output file for printing.
DTN	**Document Type Number**. A user-defined number, from 0 to 99999, assigned to a form. ▪ Indicates the category of a form ▪ Determines Print Options for all forms in the category ▪ Determines the collation order of forms within each DMGRFMT input parameter for a Document Package.
EDL	**Electronic Document Library**. A VLAM library used to store normalized forms. You can also use it to store composition source files.
EDL member	A directory that lists storage information for a form in an EDL. Also points to the chains related to the form. The maximum number of chains for an EDL member is 32. Refer to *Using VLAM* for more information.

Term	Meaning
Form	An electronic representation of a paper document. Created with your composition system. The basic component in a Documerge Document Package. A form can consist of ▪ Part of a page or multiple pages ▪ Fixed text (Boilerplate) only ▪ Locations reserved for variable data (Boilerplate Spaces) only ▪ A combination of Boilerplate and Boilerplate Spaces Documerge classifies forms into the following four types: **Implicit Forms** — Forms that are specified in a Rulebase Library Forms Table. Every Merge Set that calls a particular Rulebase Library uses the FormsTables of that Rulebase Library. Implicit forms must be stored in an EDL. **Explicit Forms** — Forms that are specified in an Explicit Forms Listin the VDR. Every Merge Set that has an Explicit Forms List uses the forms in that list. Explicit forms must be stored in an EDL. **Overlay Forms** — A form that is superimposed on another form. Specified with the xVL Print Option in a Rulebase Structure Rule. Overlays are typically stored in an EDL. **Inline Forms** — Line-printer data from an application system. Specified in an Inline Forms List in the VDR. Passed in print-ready format directly to DMGMERGE. Inline forms *are not* stored in the EDL.
Group	One or more recipients of a Document Package; for example, a customer, a regional office, or both.
Merge Set	The complete set of VRF tags and data necessary to produce one Document Package. The VDR writes the following Document Package components to tags in the VRF: ▪ Variable data from an application system ▪ Names of the forms in the Document Package ▪ Rulebase Library specifications for the forms, such as Print Options and collation order A VDR call to the DMGRFMT sub-program completes one Merge Set. In the VRF, the end of each Merge Set is marked by a high value (x'FF').
PEDEF	**Publishing Environment Definition**. Custom options for printer hardware, software, and fonts. For AFP and Metacode environments.
PELIB	**Publishing Environment Library**. Stores compiled object PEDEFs.
Replacement Character	A special keyboard character that fills a Boilerplate Space until replaced by variable data. Generated by the Documerge preprocessor. The space that Replacement Characters occupy varies with the font, whether fixed or proportional. Default Replacement Characters are: @ (commercial "at" sign) # (pound sign) **!** (exclamation mark) **"** (double quotation mark)

Term	Meaning
Reserved Tag	A predefined tag that begins with the prefix **DMG.** Has one or more of the following special purposes: ▪ Define elements of a Merge Set ▪ Convey special processing instructions to the DMGMERGE program ▪ Produce audit statistics for a DMGMERGE output file ▪ Produce dash codes There are two types of Reserved Tags: ▪ User-selected You can code these Reserved Tags in forms, Rulebase Library Tag Tables, or VDRs. The values of some user-selected Reserved Tags are set by the VDR or its DMGRFMT sub-program; these values are written to the VRF. DMGMERGE sets the values of other user-selected Reserved Tags; these values are not written to the VRF, but exist only in memory during the current DMGMERGE run. ▪ Program-generated Generated automatically by the DMGRFMT sub-program. These values are written to the VRF.
Rulebase Library	A VLAM library containing a Rulebase Table and its subordinate tables. Supplements the VDR input to the VRF. You use the Rulebase Maintenance Utility (DMGRBMUT) to build and maintain a Rulebase Library.
Rulebase Table	The parent table in a Rulebase Library. Contains or refers to subordinate tables that specify rules for assembling Document Packages.
Structure Rule	A subordinate Rulebase Library table that specifies the collation order and printing characteristics of forms in a Document Package.
Tag	Maps variable data from your application system to the VRF, and to a Boilerplate Space. Identifies a string of variable data by defining ▪ A name for the data string ▪ The starting position of the data string in the VRF ▪ The maximum length of the data string. You can code a tag in a VDR or a Rulebase Library Tag Table. The DMGRFMT sub-program writes the data string to the VRF according to the tag's starting position and length. When you code the tag name in a BPSD, you assign the data string to that Boilerplate Space.
Trailer Sheet	The last form in a Documerge output file.
VLAM	**Virtual Library Access Method**. An Oracle product. Manages VSAM libraries of electronic forms and related information.
Variable Data	Information, such as names, addresses, and dollar amounts, that changes with each Document Package and is unique to each Merge Set. Collected by your application system.
VDR	**Variable Data Reformatter**. A Documerge program that ▪ Reads variable data from your application system ▪ Determines the components of a Document Package, using supplemental information from the Rulebase Library ▪ Coordinates variable data with their appropriate Document Packages ▪ Writes this reorganized information to the Variable Replacement File (VRF). The VDR is written to your company's specifications.
VRF	**Variable Replacement File**. Contains the reorganized information that the DMGMERGE program uses to assemble Document Packages. Consists of tags and their corresponding data, arranged in Merge Sets.

Composition Terminology

Term	Meaning
Composition System	Software that you use to create forms and format them for printing.
DCF/PLUS	An Oracle product that converts IBM's DCF AFP print stream to print on selected non-IBM printers.
Document	A single- or multi-page unit containing letters, numbers or graphic characters.
Form	A single- or multi-page unit containing letters, numbers or graphic characters.
Image	The composed print-ready representation of a document.
Logical Bottom	The position where the form logically ends on a printed page (i.e. a form may have a logical bottom of one inch). Allows multiple forms to print on one page.
Logical Page Size	The coded form's intended page size.
Page	The physical output from a print device.
Pel Size	A unit of horizontal measure. It is the number of dots per inch pertaining to an output device. For AFP the pel size can be 1/240 inch or 1/300 inch. For Metacode, pel size can be 1/300 inch or 1/600 inch.

Electronic Publishing Terminology

Term	Meaning
Concatenation	The placement of two or more forms on the same side of a physical sheet of paper.
Data Stream	Information composed and ready to be sent to the printer. It is dependent upon the printer in use. For example: AFP is the data stream format required for IBM printers and Metacode is the data stream format required for Xerox centralized printers.
Duplex	Printing on both sides of a sheet of paper.
Font	The style of type used within a printed document. Each font can print in various sizes, weights and orientations. Documerge permits up to 128 fonts per single-sided image. This number may be further restricted by the memory capacity of the printer hardware and software or type of composition system.
Imposition	Printing two logical pages of a document side by side on the same side of a physical sheet of paper. This is used for booklet printing. The result is a document that can be bound in the center, or folded and distributed as a booklet. This is accomplished by printing the first and last logical pages of the document on the same side of one sheet, the second and next-to-last logical pages on the reverse side of the same sheet and so on.
Inverse Portrait	Orientation that is rotated 180 degrees from the standard portrait orientation.
Inverse Landscape	Orientation that is rotated 180 degrees from the standard landscape orientation.
Landscape	Text printed across the long axis of the page.
Orientation	The placement of the type on the page. Orientation can be portrait or landscape or the inverses of these.
Page Parity	A function which permits you to specify the start side of a page or document.
Portrait	Text that is printed across the short axis of the page.
Simplex	Printing only on one side of a sheet.
Single-sided Image	One side (or a portion of one side) of a sheet.
Template	Fixed text that can be stored at a remote location for distributed printing. Variable data is transmitted to that location to be merged with the template.
Tumble Printing	A form of duplex printing which enables the binding of a document along the shorter axis (or top) of the paper. These documents are produced in head-to-toe format for portrait orientations and head-to-head format for landscape orientations.

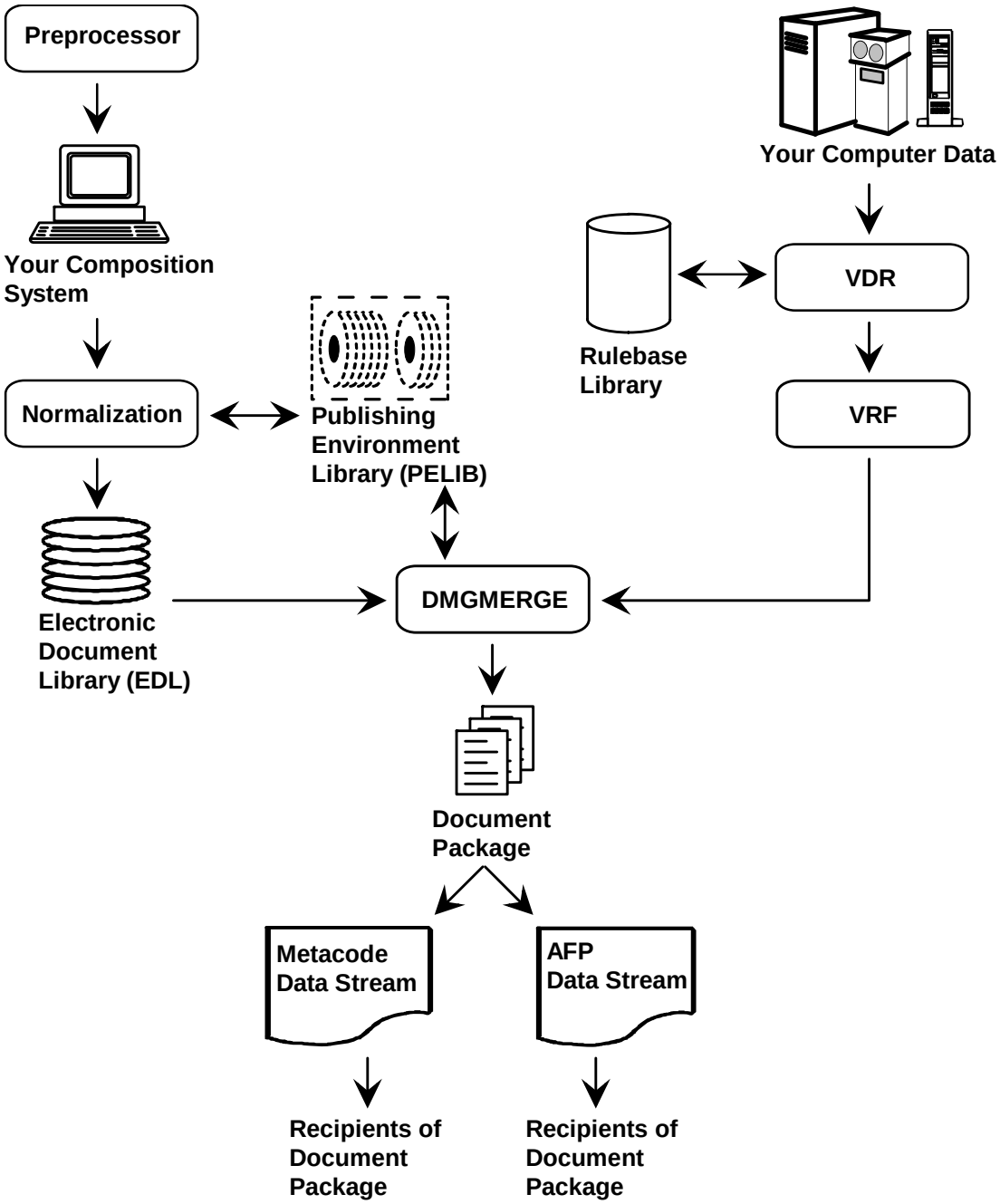
31-Bit Addressing in Documerge 3.x

Documerge 3.x uses the extra storage available through the 31-bit address mode.

With 31-bit addressing, Documerge runs in the storage above the line (above the 16 MB limit of the 24-bit address mode). Therefore, you can use more storage than the REGION allows.

Refer to "31-Bit Addressing" on page 192 for more information.

Documerge Processing Flow



Composition Preprocessors

Several mainframe-based composition systems require an interface program, or preprocessor, to generate source files for printing with Documerge.

Oracle provides a preprocessor for each of the following composition systems:

Preprocessor	Composition System
DMGOPRN	IBM Overlay Generation Language (OGL)
DMGHPRN	Xerox Host Forms Description Language (HFDL)
DMGXPRN	Xerox CompuSet (formerly XICS)

Preprocessors support the Documerge variable-data tagging method — the Boilerplate Space Definition (BPSD). You code BPSD commands in composition source files where you want Documerge to insert variable data. Documerge places no limit on the number of BPSD Commands in the composition source.

The preprocessors convert the BPSD commands into Replacement Characters and build a Boilerplate Index Packet. The Replacement Characters reserve specific areas on the print-ready form for variable-data merging. The Boilerplate Index Packet lists BPSDs and their parameters, and it maps the variable data to their assigned Boilerplate Spaces.

Refer to "Documerge Composition Preprocessors" on page 31 for more information.

Normalization

Normalization is the reorganization of a print-ready data stream (Boilerplate) into a format that maximizes Documerge efficiency. (Forms in line printer mode are not normalized.)

As part of its processing, the Normalization program appends the Boilerplate Index Packet at the beginning of the Boilerplate. (The Boilerplate Index Packet is generated by the preprocessor — or by your composition system, if it does not require a preprocessor.)

Normalization does not affect the appearance of the printed form in any way.

After you normalize a form, you load it into an EDL with the VLAM utility VLMMAINT.

Refer to "Normalization" on page 51 for more information.

Publishing Environment Library (PELIB)

The PELIB contains Publishing Environment Definitions (PEDEFs). The PEDEFs specify printer hardware characteristics, printer software requirements, font characteristics, translation tables and printer commands that are placed within the print images by Documerge.

Refer to "The Publishing Environment Library (PELIB)" on page 73 for more information.

Electronic Document Library (EDL)

The Document Packages created by Documerge consist of forms which are stored in an Electronic Document Library (EDL). The forms are constructed into Document Packages.

EDL members can be forms, also referred to as chains. Each chain is a composed and normalized for a unique printer. The EDL members are contained within VLAM libraries. VLAM libraries are VSAM Relative Record Data Sets (RRDS) that have been formatted into VLAM libraries for use by Documerge.

The EDL contains forms that have been composed and normalized according to the printer hardware used.

Composition of forms is an independent step of Documerge.

Refer to "The Electronic Document Library (EDL)" on page 105 for more information.

Rulebase Library

The Rulebase Library contains tables of rules for collation of the Document Packages, printing of the Document Packages, and variable data requirements.

The Rulebase Library members are contained within VLAM libraries. VLAM libraries are VSAM Relative Record Data Sets (RRDS) that have been formatted into VLAM libraries for use by Documerge.

The Rulebase Tables consist of the following:

- Tag Table indicates:
  - The name of the BPSD tag assigned to variable data.
  - The maximum length of the variable data.
  - The position of the variable data in relation to all variable data passed from the VDR to DMGRFMT.
- Group Table indicates:
  - All possible recipients of the Document Packages.
  - The Structure Rule name used per recipient.
  - Optional sort fields, per recipient.
- Structure table defines the order in which EDL members are placed within the Document Package and their printing characteristics.
- Forms table specifies EDL members that are mandatory (implicit) to the Document Package.
- Imposition Definition table contains the page formatting rules for Document Packages that are printed as booklets.
- Logical Page Definition table divides a Single-Sided Image (SSI) into smaller areas for creating special formats, such as columns.
- Rulebase table specifies all tables used during the Documerge process.

Rulebase Library tables are added as members to the VLAM Library by the Rulebase Maintenance Utility, DMGRBMUT.

Each of the Rulebase Library tables can be independent external tables allowing you to mix and match the printing and collation rules for a given application. Alternatively, all construction rules can be in one table, the Rulebase Table.

Refer to "The Rulebase Library" on page 125 for more information.

Variable Data Reformatter (VDR)

The VDR is a program that reads variable data files produced by the application system. The VDR acts as a bridge between the application system and Documerge. The VDR formats the variable data into a form that is acceptable for Documerge processing. The VDR obtains the printing, collating and variable data rules from the Rulebase Tables through the use of the Documerge Reformatter (DMGRFMT) sub-program.

DMGRFMT extracts the Rulebase information. DMGRFMT then uses the Variable Replacement File Writer (DMGVRFWR) sub-program to write this information to the Variable Replacement File (VRF). The DMGRFMT program also opens the EDL to verify that the requested EDL members exist and obtains EDL member information that is placed in the VRF.

Not all print images contained within a Document Package must be EDL members. Documerge accepts print images generated by application system print programs. Print images such as these are referred to as inline forms. The print lines generated by the print programs are passed as variable data to the VDR.

Refer to "The Variable Data Reformatter (VDR)" on page 191 for more information.

Variable Replacement File (VRF)

The VRF is a file that contains a series of tagged information. The tagged information consists of:

- The EDL member names for the forms that are used in the Document Package.
- The print options for each of the forms.
- Variable data that is available for merging.
- The order in which the images are printed per recipient.
- Inline form data.
- Documerge Reserved Tags generated by DMGRFMT or the VDR.

Refer to "The Variable Replacement File (VRF)" on page 251 for more information.

Reserved Tags

Reserved Tags identify data that is used by DMGMERGE for construction of Document Packages and generation of variable data. The data contained in reserved tags can be used for statistical purposes, generation of dash codes and processing instructions for DMGMERGE.

Refer to "The Variable Replacement File (VRF)" on page 251 for more information.

Merging Program (DMGMERGE)

DMGMERGE does the following:

- Reads the VRF.
- Obtains the EDL forms as designated in the VRF.
- Collates forms as determined by Print Options in the VRF.
- Merges variable data into EDL forms.
- Creates statistical information.
- Prints the forms to the appropriate clean or error file.

DMGMERGE reads the Boilerplate Index Packet in each EDL form. This is done to determine if variable data merging is required and what variable data is to be placed on the print image.

DMGMERGE obtains the variable data for merging on the forms from the VRF. This is done based on tag names that match those contained in the Boilerplate Index Packet.

NOTE

In Documerge 3.x, DMGMERGE obtains variable data (tag values) when it finds the corresponding Replacement Characters in the form itself. In previous versions, DMGMERGE obtained tag values immediately upon finding the Replacement Characters in the Boilerplate Index Packet.

By obtaining tag values later in its processing, DMGMERGE ensures correct page and sheet counts.

In Documerge 3.x, page and sheet counts that are printed on a Trailer page include the Trailer page in the count.

DMGMERGE then places printer commands in the form as required. This is based on parameters specified in the MERGEDEF and PRINTDEF contained in the PELIB. The forms are then collated into Document Packages based upon recipient requirements. The completed Document Packages are directed by DMGMERGE to the appropriate output dataset or SYSOUT class as defined in the DMGMERGE control cards.

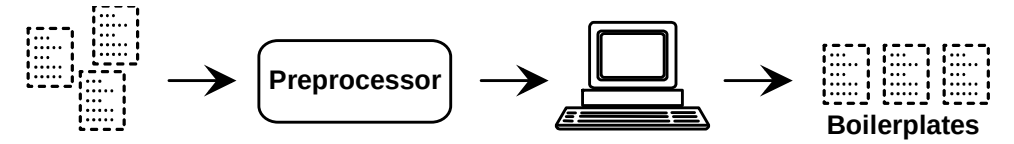
Refer to "The DMGMERGE Program" on page 371 for more information.

Documerge Composition Preprocessors

Several mainframe-based composition systems require an interface program, or **preprocessor**, to generate source files for printing with Documerge. Oracle provides a preprocessor for each of the following composition systems:

Composition System	Documerge Preprocessor
IBM Overlay Generation Language (OGL)	**DMGOPRN**
Xerox CompuSet (formerly XICS)	**DMGXPRN**
Xerox Host Forms Description Language (HFDL)	**DMGHPRN**

A Documerge preprocessor works with its related composition system to generate a data stream with the information that Documerge requires for printing. This print-ready data stream is called a **Boilerplate**.



The DMGMERGE program needs to know:

- the locations (**Boilerplate Spaces**) in a Boilerplate where you want to print variable data
- the specific data that you want to print in each Boilerplate Space.

You give DMGMERGE this information through the **Boilerplate Space Definition (BPSD)**. A Documerge preprocessor supports BPSD processing for its related composition system.

NOTE

Some composition systems do not require a Documerge preprocessor because they support BPSD processing internally. See "Workstation Composition Interfaces" on page 49 for more information.

This chapter discusses the Documerge preprocessors. For detailed information about a composition system, refer to the documentation supplied with that system.

The Boilerplate Space Definition (BPSD)

The Boilerplate Space Definition (BPSD) is a composition command developed by Oracle. You use Documerge BPSDs with your composition system to mark specific areas (Boilerplate Spaces) in a Boilerplate. The DMGMERGE program inserts variable data into these Boilerplate Spaces.

You code BPSDs in composition source files where you want DMGMERGE to insert variable data.

For each BPSD, you code

- The name of the tag whose data you want printed in that Boilerplate Space
- The length of the tag's data
- Optional parameters for processing the data

NOTE

Documerge places no limit on the number of BPSDs you can code in a source file. A BPSD doesn't have to define a contiguous area in a form, and it can span multiple pages.

BPSD coding syntax varies among composition systems. Refer to *Composing Forms for Documerge* for information about coding BPSDs.

When DMGMERGE finds a tag name coded in a Boilerplate Space it searches for the tag's value according to the following hierarchy:

(1) Tags specified with the TAG= control card of the MERGE command

(2) Tags specified with the TAG= control card of the FILEDEF command

(3) Tags specified with the TAG= control card of the GLOBAL command

(4) Tags written to the VRF

(5) Reserved Tags generated internally by DMGMERGE

TIP

Any tag—even tags internally generated DMGMERGE tags—can be overridden if desired. For example, tag DMG.DATE as generated by program DMGMERGE contains today's date in *mm/dd/yy* format (eight characters). However, you could cause a different value by using a TAG parameter or having the VDR add this tag to the VRF. For example, the following forces the value to be “01/02/03”:

```
GLOBAL TAG=(DMG.DATE '01/02/03')
```

You would not even be limited to eight characters for the tag value.

```
GLOBAL TAG=(DMG.DATE 'Monday, December 6, 2004')
```

Any BPSD using this DMG.DATE would need a large LENGTH value to avoid the DMGMERGE truncation error message.

A tag is considered missing If DMGMERGE completes this search sequence and does not find the value. DMGMERGE generates an error message if a tag that is specified as mandatory (with the BPSD parameter TYPE=M) is missing.

You can use the **Y**, **N, R, and 1** values of the DELETE= BPSD parameter to control the use of tag data as DMGMERGE processes the Document Packages in a Merge Set. However, in previous releases, once DMGMERGE deleted the data from the VRF occurrence of a tag, the data was not reset to the first VRF occurrence until processing of the current Document Package ended.

Delete=Values for Re-Setting Tag Values

Documerge offers **1** and **R** DELETE= parameter values that allow tags to be reset to their first values while a Document Package is still being processed. These reset values let you reuse (reprint) a form with the same tag data in the same Document Package.

Here is a summary of the DELETE= parameter values and their actions:

- **DELETE=N** — DMGMERGE inserts the data of the next VRF tag occurrence that matches the BPSD name and does not delete the tag data from working storage.

  N is the default if the DELETE parameter is omitted from the BPSD.

- **DELETE=Y** — DMGMERGE inserts the data of the next VRF tag occurrence that matches the BPSD name and then deletes the tag data from working storage. The data is not used again in processing the current Document Package.

  A DELETE=Y parameter coded for an internal Reserved Tag is ignored.

- **DELETE=1** — DMGMERGE restores the working storage values of all tag data occurrences that match the BPSD name, and inserts the data for the first VRF tag occurrence that matches the BPSD name, and does not delete the tag data.

- **DELETE=R** — DMGMERGE restores the working storage values of all tag data occurrences that match the BPSD name, and inserts the data for the first VRF tag occurrence that matches the BPSD name, and then deletes the tag data.

For more information about the BPSD parameters DELETE= and TYPE=, refer to *Composing Forms for Documerge*.

Zero-Length BPSD Tags

Documerge allows an EDL member to contain only an index packet of BPSDs that all have lengths of zero and thus contain no print data. The DMGZEROL utility creates index-packet-only forms with zero-length BPSD tags, or inserts zero-length tags in existing forms at the beginning of their index packets.

For details about DMGZEROL tag processing, see "DMGZEROL" on page 462.

Use these zero-length tags to

- Reset deleted tags to control the processing of a subsequent form.

  Your EDL can contain a reset-only form that contains BPSDs with LENGTH=0 and DELETE=1. Documerge can use this form to reset tags listed in the index packet of a subsequent form to their first-occurrence values.

- Generate internal statistics or audit records by calling a user-exit program (via a DMG.C.xxx tag). For details, see "DMG.C.xxx Reserved Tag Processing" on page 331.

  Documerge looks up any zero-length tags in the Index Begin Record as it begins processing a Document Package — before it writes any output or updates any page and sheet counts and numbers.

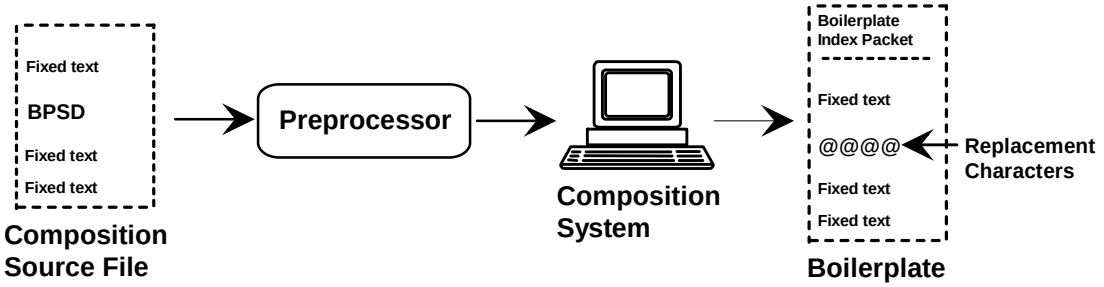
  When Documerge encounters a zero-length DMG.C.xxx command tag or a tag value that starts with a DMGCTRIGGER value, it sets page / sheet counts and numbers to zero. For succeeding Document Packages, page / sheet counts and numbers will be those of the last Document Package processed.

Functions of a Documerge Preprocessor

BPSD processing for the composition systems listed on page 2-31 requires a Documerge preprocessor.

The preprocessor does the following:

- Recognizes BPSDs in a composition source file
- Replaces BPSDs with **Replacement Characters**, which reserve Boilerplate Spaces in the Boilerplate until the DMGMERGE step
- Generates a **Boilerplate Index Packet**, which maps variable data to their assigned Boilerplate Spaces, for the Boilerplate.



NOTE

Some composition systems perform these functions internally. See "Workstation Composition Interfaces" on page 49 for more information.

DCF Processing

The Document Composition Facility (DCF) is a mainframe-based composition system produced by IBM. DCF provides a composition compiler, called SCRIPT. SCRIPT processes documents marked up with its own control words as well as documents marked up with Generalized Markup Language (GML) tags.

DCF Interface: ISIPROF and ISIBPSTG

ISIPROF is a SCRIPT profile (provided by Oracle) which imbeds DSMPROF3 and the macro ISIBPSTG.

The ISIBPSTG macro creates the Boilerplate Index Packet using the SCRIPT Write To File command (.WF), and replaces the BPSD command with Replacement Characters during SCRIPT processing.

The AFP print image produced by SCRIPT and the Boilerplate Index Packet require DMGAFPP Normalization prior to loading to the EDL.

The line-printer data stream that DCF produces is ready for an EDL and DMGMERGE without Normalization.

Replacement characters for DCF-composed documents are generated by the ISIBPSTG macro. The defaults are the "at" sign (@), pound sign (#), exclamation mark (**!**), and double quotation mark (**"**).

To specify the Replacement Characters, locate the set command statement (.se)

```
        .se subchars='@#!"'
```

in ISIBPSTG and remove or add additional Replacement Characters. Up to 16 Replacement Characters may be defined. A minimum of two Replacement Characters must be defined. Characters must be available in the font family being used with the BPSD. **The characters** *cannot* **be contained within the text of the document**.

ISIBPSTG always creates a Boilerplate Index Packet. If there are no BPSD commands present in the composition source, an Index Begin Record and Index End Record are written to the INDEXFL file.

DCF JCL Example

```
//DCFJOB   (Place JOB Card Here)
//*
//JOBLIB   DD DSN=DOCUMERG.V03R02.LOADLIB,DISP=SHR
//         DD DSN=SYS1.DCFLOAD,DISP=SHR
//*
//DCFSTEP EXEC PGM=IKJEFT01,REGION=3072K
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD DUMMY
//SYSTSPRT DD SYSOUT=*,DCB=LRECL=200
//INDEXFL  DD DSN=&&INDEXFL,DISP=(NEW,PASS),
//            UNIT=SYSDA,
//            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209),
//            SPACE=(TRK,(2,2))
//TEXTLIB  DD DISP=SHR,DSN=SYS1.DCFGML3
//         DD DISP=SHR,DSN=DOCUMERG.V03R02.TEXT
//SCRPTLIB DD DISP=SHR,DSN=SYS1.DCFGML3.
//SYSTSIN  DD *
SCRIPT 'DOCUMERG.V03R02.TEXT(DEC)' DEVICE(3820A) +
       PROFILE('DOCUMERG.V03R02.TEXT(ISIPROF)') MESSAGE(ID) +
       FONTLIB('SYS1.AFP.FONTLIB') +
       FILE('DOCUMERG.V03R02.LIST3820(TEMP)') +
       DDUT TWOPASS SYS(X NO D YES) CO
/*
//
```

This JCL file executes TSO and passes TSO the command 'SCRIPT'.

DCF Files

The following are DCF Files found in "DCF JCL Example" on page 35.

SCRIPT Input Files

- **TEXTLIB**
  Datasets that are used during SCRIPT processing that contain any members called for by a .IM (imbed) command. ISIPROF imbeds DSMPROF3 from SYS1.DCFGML3, and ISIBPSTG from DOCUMERG.V03R02.TEXT.
- **SCRPTLIB**
  Files that are used during SCRIPT processing that contain members that are DCF macros.
- **SYSTSIN**
  Contains the SCRIPT command, specifies the files to be processed and designates SCRIPT command options.
  - **'DOCUMERG.V03R02.TEXT(DEC)'**
    A sequential file containing the SCRIPT source member (DEC) to be processed.
  - **PROFILE(DOCUMERG.V03R02.TEXT(ISIPROF))**
    A file containing the SCRIPT PROFILE member ISIPROF.
  - **FONTLIB(SYS1.AFP.FONTLIB)**
    A file containing the AFP font descriptions.
  - **DDUT and TWOPASS (or FPASSES(2))**
    SCRIPT command options required for BPSD processing.

NOTE

To print the index packet when using DCF, the .se PRINTINDEX='NO' can be changed to .se PRINTINDEX='YES' in the ISIBPSTG macro, or a .se PRINTINDEX='YES' command can be added to your DCF source code.

Output Files

- **INDEXFL**
  A sequential file containing the boilerplate index records written by the .WF in ISIBPSTG and directed by the .DD DSMUTWTF DD INDEX as contained in ISIPROF. This file's recommended physical characteristics are:
  - Record format = VBM
  - Record length = 8205
  - Blocksize = 8209

  This file requires DMGAFPP Normalization to concatenate it to the form prior to loading to the EDL.
- **SYSTSIN SCRIPT FILE, DOCUMERG.V03R02.LIST3820(TEMP)**
  The sequential file containing the composed AFP print image. This file requires DMGAFPP Normalization prior to loading to the EDL. This file's recommended physical characteristics are:
  - Record format = VBM
  - Record length = 8205
  - Blocksize = 8209

DCF/PLUS Processing

DCF/PLUS is a post processor which converts DCF (AFP) output to Metacode for printing on non-IBM printers. DCF/PLUS is a Oracle product.

DCF/PLUS Interface: DPLPROF and ISIBPSTG

DPLPROF is a SCRIPT profile (provided by Oracle) which imbeds DSMPROF3, the SCRIPT macro ISIBPSTG (also provided by Oracle), and the profile supplement member DPLPSUPL.

NOTE

DPLPROF is delivered with the Oracle stand-alone product DCF/PLUS. Before using DPLPROF verify the following:

- DCF/PLUS has been fully installed.
- DPLPROF imbeds the ISIBPSTG macro.
- The appropriate Files have been defined according to the instructions contained in DPLPROF.

The ISIBPSTG macro creates the Boilerplate Index Packet using the SCRIPT Write To File command (.WF) and replaces the BPSD command with Replacement Characters during SCRIPT processing. The AFP print image produced by SCRIPT is input to the DCF/PLUS program, DPLMAIN. DPLMAIN converts the composed AFP image into a Metacode print image. The print image from DPLMAIN and the Boilerplate Index Packet require DMGMETP Normalization prior to loading to the EDL.

Replacement Characters for DCF-composed documents are generated by the ISIBPSTG macro. The defaults are the "at" sign (@), pound sign (#), double quotation mark (**"**), and exclamation mark (**!**). Locate the following set command (.se):

```
        .se subchars='@#!"'
```

statement in ISIBPSTG and remove or add additional Replacement Characters. Up to 16 Replacement Characters may be defined. A minimum of two Replacement Characters must be defined. Characters must be available in the font family being used with the BPSD. **The characters** *cannot* **be contained within the text of the document**.

ISIBPSTG always creates a Boilerplate Index Packet. If there are no BPSD commands present in the composition source, an Index Begin Record and an Index End Record are written to the INDEXFL file.

For complete information about DCF/PLUS processing, refer to the *DCF/PLUS 3.1 Reference Guide*.

DCF/PLUS JCL Example

```
//DCFPLUS  JOB (Place JOB Card Here)
//*
//JOBLIB   DD DSN=DOCUMERG.V03R02.LOADLIB,DISP=SHR
//         DD DSN=DCF3PLUS.V03R02.LOADLIB.X9700,DISP=SHR
//         DD DSN=SYS1.DCFLOAD,DISP=SHR
//DCFSTEP EXEC PGM=IKJEFT01,REGION=3072K
//SYSTSPRT DD SYSOUT=*,DCB=LRECL=200
//INDEXFL  DD DSN=&&INDEX,DISP=(NEW,PASS),
//            DCB=(RECFM=VBM,LRECL=155,BLKSIZE=3000),
//            SPACE=(TRK,(2,2)),UNIT=SYSDA
//TEXTLIB  DD DSN=DOCUMERG.V03R02.TEXT,DISP=SHR
//         DD DSN=DCF3PLUS.V03R02.SCRIPT.X9700,DISP=SHR
//         DD DSN=SYS1.DCFGML3,DISP=SHR
//SCRPTLIB DD DSN=SYS1.DCFGML3,DISP=SHR
//SYSTSIN  DD *
  SCRIPT  'DOCUMERG.V03R02.TEXT(DEC)' DEV(9700A) +
  PROF('DCF3PLUS.V03R02.SCRIPT.X9700(DPLPROF)')  MESSAGE(ID) +
  FONTLIB('DCF3PLUS.V03R02.FONTLIB.X9700') +
  FILE('DOCUMERG.V03R02.LIST3820(TEMP)') +
  DDUT SYS(X NO D YES P 9700A03) TWOPASS CO QU
/*
//*
//POSTP   EXEC PGM=DPLMAIN,REGION=2048K
//LIST3820 DD DSN=DOCUMERG.V03R02.LIST3820(TEMP),DISP=SHR
//LIST9700 DD DSN=DOCUMERG.V03R02.METACODE(TEMP),
//            DISP=(NEW,CATLG),SPACE=(CYL,(1,1)),
//            DCB=(RECFM=VB,LRECL=155,BLKSIZE=3000),UNIT=SYSDA
//PEDEF    DD DSN=DCF3PLUS.V03R02.PELIB.X9700,DISP=SHR
//FONT9700 DD DSN=DCF3PLUS.V03R02.FONTLIB.X9700,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
EXECUTE
 INPUT DDNAME=LIST3820
 FONTLIB DDNAME=FONT9700
 OUTPUT -
  PDEV=9700 DUPLEX=NO RSTACK=NONE NUFRONT=NONE -
  FONTS=(UN106E UN107E UN108E UN208E UN110E UN210E UN211E UN214B-
         UN224B PR111E PR211E PR114B UN111E UN105E P07TDC FORMSX) -
  DDNAME=LIST9700 -
  ENVDEF=97ENV
END
/*
//
```

DCF/PLUS Files

The following are DCF/PLUS Files found in "DCF/PLUS JCL Example" on page 38.

SCRIPT Input Files

- **TEXTLIB**
  Files that are used during SCRIPT processing that contain any members called for by the .IM (imbed) command. DPLPROF imbeds DSMPROF3 from the SYS1.DCFGML3 file, ISIBPSTG from the DOCUMERG.V03R02.TEXT file and DPLPSUPL from DCF3PLUS.V03R02.SCRIPT.X9700.
- **SCRPTLIB**
  Files that are used during SCRIPT processing that contain members that are DCF macros.

- **SYSTSIN**
  Issues the SCRIPT command, specifies the files to be processed and designates SCRIPT command options.
  - **'DOCUMERG.V03R02.TEXT(DEC)'**
    A sequential file containing the SCRIPT source member (DEC) to be processed.
  - **PROFILE(DCF3PLUS.V03R02.SCRIPT.X9700(DPLPROF))**
    A file containing the SCRIPT PROFILE member DPLPROF.
  - **FONTLIB(DCF3PLUS.V03R02.FONTLIB.X9700)**
    The file containing the Metacode font descriptions.

SCRIPT Output Files

- **INDEXFL**
  A sequential file containing the boilerplate index records written by the .WF in ISIBPSTG and directed by the .DD DSMUTWTF DD INDEX as contained in DPLPROF.

  This file's recommended physical characteristics are:
  - Record format = VB
  - Record length = 155 (the Metacode LRECL cannot exceed 256)
  - Blocksize = 3000

  This file requires DMGMETP Normalization to concatenate it to the form prior to loading to the EDL.
- **SYSTSIN SCRIPT FILE DOCUMERG.V03R02.LIST3820(TEMP)**
  The sequential file containing the composed AFP print image. This file's recommended physical characteristics are
  - Record format = VBM
  - Record length = 8205
  - Blocksize = 8209

DPLMAIN Input Files

- LIST3820
- **DOCUMERG.V03R02.LIST3820(TEMP)**
  A sequential file containing the composed AFP print image from SCRIPT. This file's recommended physical characteristics are:
  - Record format = VBM
  - Record length = 8205
  - Blocksize = 8209
- **PEDEF DCF3PLUS.V03R02.PELIB.X9700**
  A file containing PEDEF objects.
- **FONT9700 DCF3PLUS.V03R02.FONTLIB.X9700**
  A file containing the Metacode font descriptions. This must be the same font library as used by SCRIPT.

- **SYSIN**
  Contains control cards used during DPLMAIN processing:
  - **INPUT DDNAME=**
    Specifies the filename of the JCL control statement containing SCRIPT composed AFP print image.
  - **FONTLIB DDNAME=**
    Specifies the filename of the JCL control statement containing the Metacode font characteristics.
  - **OUTPUT**
    Contains the parameters used by DPLMAIN. Refer to *DCF/PLUS Reference Guide* for additional information.

DPLMAIN Output Files

- **LIST9700**
  Sequential file containing the Metacode print image. This file requires DMGMETP Normalization prior to loading to the EDL. This file's recommended physical characteristics are:
  - Record format = VB
  - Record length = 155 (the Metacode LRECL cannot exceed 256)
  - Blocksize = 3000

OGL Processing

IBM's Overlay Generation Language (OGL) creates electronic Overlays.

OGL Interface (DMGOPRN)

DMGOPRN is a program that replaces the BPSD command in the OGL composition source with Replacement Characters and creates the Boilerplate Index Packet. The OGL composition source with Replacement Characters is compiled by the OGL program DZIOVRLY. The AFP print image from DZIOVRLY and the Boilerplate Index Packet require DMGAFPP Normalization before loading to the EDL.

The DMGOPRN program generates Replacement Characters for OGL documents. The defaults are the "at" sign (@), pound sign (#), exclamation mark (!), and double quotation mark ("). To specify the Replacement Characters, use the SUBCHARS= control card.

WARNING!

DMGOPRN does not process OGL commands with keywords that span more than one input record. Therefore, the keywords for a command must be coded in only one input record.

OGL/370 Release 1.00

Documerge lets you use IBM's OGL/370 Release 1.00.

To improve printer efficiency, OGL/370 sections the AFP data stream by fonts. This arrangement doesn't affect the way the printed page looks. But, because DMGOPRN creates the Boilerplate Index Packet before OGL/370 sections the data stream, the Replacement Characters and their related data do not match the data stream. Therefore, Documerge has special requirements for Replacement Characters with OGL/370:

- You must tell DMGOPRN to generate a unique Replacement Character for each font defined in the Overlay. DMGOPRN generates these Replacement Characters from a group of default characters. You do this with the **OGL370=** SYSIN parameter.
- You must create the group of default characters for DMGOPRN. This group must have at least as many default characters as there are fonts defined in the Overlay. You do this with the **SUBCHARS=** SYSIN parameter.

DMGOPRN JCL Example

```
//DMGOPRN   ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* **       DOCUMERGE V. 3.2 OGL SOURCE CODE PRE-PROCESSOR          **
//* **                                                                **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.LOADLIB,DISP=SHR
//*
//DMGOPRN  EXEC PGM=DMGOPRN
//*
//INFILE   DD  *
 ogl source code
/*
//SYSIN    DD  *         <= NEW FOR 2.0
 PRINT=NO -
 SUBCHARS='#@"!'
/*
//OUTFILE  DD  DSN=&&OGLINPUT,
//             DISP=(NEW,PASS),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//             UNIT=sysda,
//             SPACE=(TRK,(1,1),RLSE)
//INDEXFL  DD  DSN=&&INDEXFL,
//             DISP=(NEW,PASS),
//             DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209),
//             UNIT=sysda,
//             SPACE=(TRK,(1,1),RLSE)
//MESSAGE  DD  SYSOUT=*,    <= DMGOPRN MESSAGES
//             DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD  SYSOUT=*
//*
//* The OGL (DZIOVRLY) step follows using the OGL source in OUTFILE
//* as input.
//*
//
```

OGL Files

The following are OGL files found in "DMGOPRN JCL Example" on page 42.

DMGOPRN Input Datasets

- **INFILE**

  Sequential files containing OGL source code with BPSD commands, DOCUMERG.V03R02.TEXT(OGL).

- **SYSIN**
  - **OGL370=**SYSIN (continued)

YES or Y DMGOPRN generates a unique Replacement Character for each font defined in the Overlay. DMGOPRN generates these Replacement Characters from a group of default characters.

NOTE: You must create the group of default characters for DMGOPRN. This group must have at least as many default characters as there are fonts defined in the Overlay. You do this with the **SUBCHARS=** SYSIN parameter.

IMPORTANT: When the OGL370= value is YES or Y:

- Be careful when you use the CHAR= parameter in a BPSD command: don't assign a Replacement Character to more than one font ID. A good way to avoid this is to assign the Replacement Character only to the first reference to a font ID. DMGOPRN generates that Replacement Character for subsequent references to the font ID, regardless of the font name associated with the font ID.
- You can use the GEN=NO parameter in a BPSD command. But be sure that you don't assign a Replacement Character that's used by a different, previous font ID.

NO or N DMGOPRN generates Replacement Characters as usual. NO or N is the default value for the OGL370= parameter.

NOTE: When the OGL370= value is NO or N, you can assign any default Replacement Character to multiple fonts.

- **PRINT=**

YES or Y DMGOPRN produces a printable Boilerplate Index Packet.

NO or N DMGOPRN suppresses a printable Boilerplate Index Packet. PRINT=NO is the default.

NOTE: Boilerplate Index Packets produced with PRINT=YES cannot be normalized and loaded to the EDL. Boilerplate indexes to be normalized and loaded to the EDL must be produced with PRINT=NO.

- **SUBCHARS=**

  This overrides the default Replacement Characters. It is recommended that at least two Replacement Characters be specified, with a maximum of 16. Characters must not appear anywhere within the text. Characters should be included in the font. SUBCHARS values must be enclosed in single quotes.

  You can define non-printable characters as Replacement Characters. The SUBCHARS= parameter lets you define Replacement Characters as an EBCDIC hexadecimal string. You can use this option if a form's text includes characters that are usually used as Replacement Characters, or if the form uses a large number of fonts. The syntax is:

```
SUBCHARS=X'hh...'
```

  `'hh...'` represents one or more pairs of hexadecimal digits, each pair defining a Replacement Character.

DMGOPRN Output Files

- **INDEXFL**

  A sequential file containing Boilerplate index records created by DMGOPRN. This file's recommended physical characteristics are:
  - Record format = VBM
  - Record length = 8205
  - Blocksize = 8209

  This dataset requires DMGAFPP Normalizaton to concatenate it to the AFP print

image prior to loading to the EDL.

NOTE

The print image contained in the OVRLIB is normalized, not the print image designated to the SAMPLE file.

- **MESSAGE**
  Contains messages produced by DMGOPRN.
- **OUTFILE**
  Sequential file containing the OGL source with Replacement Characters created by DMGOPRN. This file's recommended physical characteristics are:
  - Record format = FB
  - Record length = 80
  - Blocksize = 80

  The output file from the preprocessor is the input file for the composition program.

XICS Processing

The Xerox Integrated Composition System (XICS) is a composition software system used to format documents for printing. XICS is a Xerox product.

XICS Interface: DMGXPRN

DMGXPRN is a program which replaces the BPSD command in the XICS composition source with Replacement Characters and creates the Boilerplate Index Packet. The XICS composition source with Replacement Characters is compiled by the XICS programs, CompuSet and XRXINT. The Metacode print image from XRXINT and the Boilerplate Index Packet require DMGMETP Normalization prior to loading into the EDL.

Replacement characters for XICS composed documents are generated by the DMGXPRN program. The defaults are the at sign (@), pound sign (#), exclamation point (**!**), and double quote (**"**). To specify the Replacement Characters, use the SUBCHARS= control card.

DMGXPRN JCL Example

```
//*DMGXPRN  ** put your job card here **
//*
//*
*****************************************************************
//* **                                                          **
//* **          DOCUMERGE V. 3.2 XICS SOURCE CODE PRE-PROCESSOR   **
//* **                                                          **
*****************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGXPRN  EXEC PGM=DMGXPRN
//*
//INFILE   DD  *
 XICS SOURCE CODE
/*
//SYSIN    DD  *         <= NEW FOR 2.0
 PRINT=NO -
 SUBCHARS='#@"!'
/*
//OUTFILE  DD  DSN=&&XICSNPUT,
//             DISP=(NEW,PASS),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//             UNIT=sysda,
//             SPACE=(TRK,(1,1),RLSE)
//INDEXFL  DD  DSN=&&INDEXFL,
//             DISP=(NEW,PASS),
//             DCB=(RECFM=VB,LRECL=155,BLKSIZE=3000),
//             UNIT=sysda,
//             SPACE=(TRK,(1,1),RLSE)
//MESSAGE  DD  SYSOUT=*,    <= DMGXPRN MESSAGES
//             DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD  SYSOUT=*
//*
//* The XICS step follows using the XICS source in OUTFILE as input.
//*
//
```

XICS Files

The following are XICS files found in "DMGXPRN JCL Example" on page 45.

DMGXPRN Input Files

- **INFILE**
  A file containing XICS composition source with BPSD commands.
- **SYSIN**
  A file containing optional commands for the DMGXPRN program. The commands are
  - **PRINT=**
    A value of YES produces a printable Boilerplate Index Packet; a value of NO suppresses a printable Boilerplate Index Packet.
  - **SUBCHARS=**
    Overrides the default Replacement Characters. It is recommended that at least two Replacement Characters be specified with a maximum of 16. Characters should not appear anywhere in the text. Characters should be included in the font. Characters must be enclosed in single quotes.

DMGXPRN Output Files

- **INDEXFL**
  A file containing the Boilerplate index records. This file's recommended physical characteristics are:
  - Record format = VB
  - Record length = 155 (the Metacode LRECL cannot exceed 256)
  - Blocksize = 3000

  This file requires DMGMETP Normalization to concatenate it to the Metacode print image prior to loading to the EDL.
- **OUTFILE**
  The input file to CompuSet. This file's recommended physical characteristics are:
  - Record format = FB
  - Record length = 80
  - Blocksize = 80

  The output file for the preprocessor is the input file for the composition program.
- **MESSAGE**
  Contains messages produced by DMGXPRN.

HFDL Processing

The Host Forms Description Language (HFDL) provides the capability for precisely defining forms electronically. HFDL runs on a mainframe and is a Xerox product.

Documerge does not support the use of HFDL optimization. Optimization must be turned off by specifying a 1 as the second digit for the OPMZ control card.

HFDL Interface: DMGHPRN

DMGHPRN is a program which replaces the BPSD Command within the HFDL composition source with Replacement Characters. It also creates the Boilerplate Index Packet. The HFDL composition source with Replacement Characters is compiled by the HFDL program (HFDL). The print image from HFDL and the Boilerplate Index Packet require DMGMETP Normalization prior to loading to the EDL.

Replacement characters for HFDL composed documents are generated by the DMGHPRN program. The defaults are the at sign (@), pound sign (#), exclamation point (!), and double quote (“). To specify the Replacement Characters, use the SUBCHARS= control card.

DMGHPRN JCL Example

```
//DMGHPRN   ** put your job card here **
//*
//*
*****************************************************************
//* **                                                          **
//* **       DOCUMERGE V. 3.2 HFDL SOURCE CODE PRE-PROCESSOR     **
//* **                                                          **
/*****************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGHPRN  EXEC PGM=DMGHPRN
//*
//INFILE   DD  *
 hfdl source code
/*
//SYSIN    DD  *         <= NEW FOR 2.0
 PRINT=NO -
 SUBCHARS='#@"!'
/*
//OUTFILE  DD  DSN=&&HFDLNPUT,
//             DISP=(NEW,PASS),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//             UNIT=sysda,
//             SPACE=(TRK,(1,1),RLSE)
//INDEXFL  DD  DSN=&&INDEXFL,
//             DISP=(NEW,PASS),
//             DCB=(RECFM=VB,LRECL=155,BLKSIZE=3000),
//             UNIT=sysda,
//             SPACE=(TRK,(1,1),RLSE)
//MESSAGE  DD  SYSOUT=*,   <= DMGHPRN MESSAGES
//             DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD  SYSOUT=*
//*
//* The HFDL step follows using the HFDL source in OUTFILE as input.
//*
//
```

HFDL Datasets

The following are HFDL files found in "DMGHPRN JCL Example" on page 47.

DMGHPRN Input Files

- INFILE
  A file containing HFDL composition source with BPSD commands.
- SYSIN
  - **SUBCHARS=**
    This overrides the default Replacement Characters. It is recommended that at least two Replacement Characters be specified with a maximum of 16. Characters should not appear anywhere in the text. Characters should be included in the font. Characters must be enclosed in single quotes.
  - **PRINT=**
    A value of YES produces a printable Boilerplate Index Packet; a value of NO suppresses a printable Boilerplate Index Packet.

DMGHPRN Output Files

- **INDEXFL**
  A file containing boilerplate index records. This file requires DMGMETP Normalization to concatenate it as a Metacode print image prior to loading to the EDL. This file's recommended physical characteristics are:
  - Record format = VB
  - Record length = 155 (the Metacode LRECL cannot exceed 256)
  - Blocksize = 3000
- **MESSAGE**
  contains messages produced by DMGHPRN.
- **OUTFILE**
  A file containing HFDL source with Replacement Characters.

Workstation Composition Interfaces

Documerge uses data streams produced by the following workstation-based composition systems:

- Intran FormBuilder V1.5
- TyForm V5 or later with CDPMerge
- Elixir ElixiForm with DocuTag V1.2.
- Ventura with Elixir VP297 and DocuTag
- XPS732

These composition systems vary in the types of data streams they produce:

- Some require a composition interface program, or preprocessor, to produce print-ready data streams with BPSD tags and Boilerplate Index Packets.
- Others do not require a preprocessor because they produce complete, normalized Boilerplates ready for loading to an EDL.

Refer to the documentation for your composition system to determine if a preprocessor is required to produce print-ready data streams for Documerge.

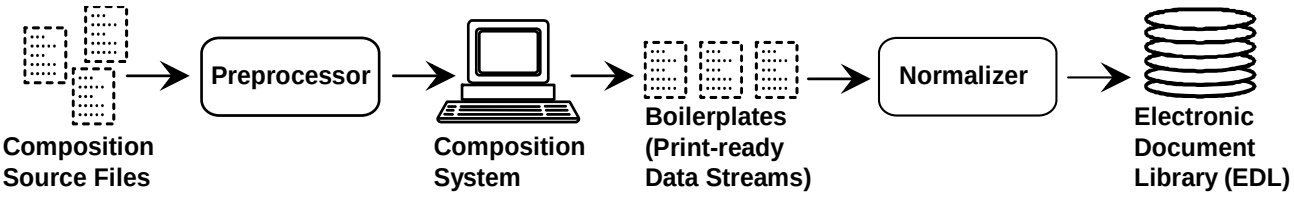
Preprocessors for workstation-based composition systems are developed, supported, and documented by the vendors of the composition systems.

Normalization

The Normalization process is required for all Metacode forms and is recommended for AFP forms. Normalization formats the print-ready data stream (Boilerplate) for use by Documerge. (Forms in line printer mode are not normalized.)

Normalization minimizes the amount of work the DMGMERGE program must do. As part of its processing, the Normalization program appends the Boilerplate Index Packet at the beginning of the Boilerplate. The Boilerplate Index Packet maps variable data to their assigned Boilerplate Spaces.

Normalization maintains the integrity of the data stream. It does not affect the printed appearance of the form.



Oracle provides two Normalization programs:

- DMGMETP for Metacode forms and Boilerplate Index Packets
- DMGAFPP for AFP forms and Boilerplate Index Packets

NOTE

Workstation composition systems output varies by vendor. Refer to the reference manuals specific to your composition system to determine if composition interface and/or Normalization processing is required of the output. Follow the processing steps required based upon the composition source and/or print image produced by the system.

Normalized Boilerplate Index Packets

Boilerplate Index Packets are a series of comment records. These records are used by the DMGMERGE program for placing variable data on the form. Boilerplate Index Packets are created by the following Documerge composition interface programs:

- DMGOPRN (OGL)
- DMGXPRN (XICS)
- DMGHPRN (HFDL)
- ISIBPSTG (DCF or DCF/PLUS)

These composition interface programs are discussed in "Documerge Composition Preprocessors" on page 31 and in *Composing Forms for Documerge*.

PC-based composition systems also create Boilerplate Index Packets.

Boilerplate Index Packets contain at least two index records: Index Begin (DMG2NDXBEG) and Index End (DMG2NDXEND). When BPSD commands are present within the composition source, the composition preprocessor produces:

- Index begin record
- Comment records
- Index records listing parameters for each BPSD command.
- Index end record

When BPSD commands are absent from the composition source, the Boilerplate Index Packet contains just the following:

- Index begin record

```
%%%DMG2INDXBEG%%%
```

- Index end record.

```
%%%DMG2INDXEND%%%
```

For Documerge processing, there is no maximum number of index records per form.

Printing of Boilerplate Index Packets produced by DMGOPRN, DMGXPRN and DMGHPRN is controlled by the SYSIN control card PRINT=. A value of YES produces a printable boilerplate index and a value of NO suppresses printing of the Boilerplate Index Packet.

NOTE

Composition source processed by DMGXPRN and DMGHPRN with PRINT=YES can be normalized and loaded to the EDL. Composition source processed by DMGOPRN with PRINT=YES *cannot* be normalized and loaded to the EDL; its print value must be set at PRINT=NO.

DMGMETP

DMGMETP is the Metacode data stream Normalization program. It adds special record identifiers to the Boilerplate Index Packet, DJDE Packets, font definitions, page definitions and replacement records.

DMGMETP can be used to renormalize forms previously normalized by DMGNMMN (the Documerge version 1 Normalization program) for creating a Common Font List.

If you use the Oracle product Template Technology, refer to the *Tag Table* for additional Normalization information.

DMGMETP Normalization JCL

```
//DMGMETP   ** put your job card here **
//*
//* ****************************************************************
//* **                                                            **
//* ** DOCUMERGE V. 3.2 METACODE NORMALIZER                       **
//* **  (DMGMETP replaces DMGNMMN)                                **
//* **                                                            **
//* ****************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGMETP  EXEC PGM=DMGMETP,REGION=4M,
//  PARM=' / WORKBUFF=500K'
//*
//INDEXFL  DD DUMMY,
//            DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//INFILE   DD DSN=Metacode.input.pds(MEMBER),
//            DISP=SHR
//OUTFILE  DD DSN=dmgmetp.output.pds(MEMBER),
//            DISP=OLD
//PEDEF    DD DSN=documerg.v03r02.pelib,
//            DISP=SHR
//FONTMETA DD DSN=documerg.v03r02.Metacode.fontlib,
//            DISP=SHR
//MESSAGE  DD SYSOUT=*,DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD SYSOUT=*
//WRKFIL   DD DSN=&&WRKFIL,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                 HALF TRACK
//SYSIN    DD *
-
PROCESS -
-        ENVDEF=97ENV
-
-        INFILE  DDNAME=INFILE   -
-        OUTFILE DDNAME=OUTFILE  -
-        FONTLIB DDNAME=FONTMETA -
-        PELIB   DDNAME=PEDEF    -
-        FGRPDEF=9700            -
-        CODEDEF=ASCII9          -
-        BOTTOM=0                -
-        COMMONFONTS=YES         -
-        GRAPHICSONLY=YES
/*
//
```

Documerge Processing of Metacode Normalized Forms

Documerge Forms with DJDE Packets

In Metacode form print streams, there are two types of DJDE packets:

- **Record-oriented packet —** typically contain IMAGE or GRAPHIC parameters. Xerox allows more than one record-oriented parameter or packet of only record-oriented parameters per page (single sided image).
- **Page-oriented packet —** typically contain FONTS, FORMS, JDE, JDL. SIDE, FEED, and/or BTEXT parameters. Xerox allows only one page-oriented packet per page (single sided image), and all page-oriented parameters must be in the same packet.

During concatenation, or anytime Documerge places two or more forms on the same page, Documerge discards any page-oriented DJDE packets in the second and subsequent forms.

Documerge Rules for Processing DJDE Packets in Forms

For Metacode form processing, these rules apply:

- Do not mix page-oriented and record-oriented parameters in the same DJDE packet.
- Bracket page-oriented packets with the D97DJDE comment records. For example:

```
%%%D97DJDEBEG%%%
[page-oriented DJDE record(s) go here]
%%%D97DJDEEND%%%
```

  These comment records can come before or immediately after the **x'8B'** page carriage control. The usual practice is to put them before the carriage control.
- For multi-page forms, there can be a page-oriented DJDE packet for each form-page, but each of these packets must have the D97DJDE comment records described previously.
- Do not bracket a record-oriented DJDE packet with the D97DJDE comment records.
- If you have coded forms with DJDE comment records that contain a *C* and a blank in the first two DJDE-skip positions, you cannot use these comment records to pass information to a post processor because Documerge 3.1 and later releases strip out (delete) any DJDE comment records.
- DMGMERGE can optionally ignore any FEED= DJDE parameters in EDL forms. Because FEED= DJDE parameters force a new sheet of paper, Oracle strongly recommends that you run DMGMERGE with the GLOBAL parameter STRIPFEED=YES. For more information about STRIPFEED, see "STRIPFEED=" on page 398.
- DMGMERGE can optionally ignore any OTEXT= DJDE parameters in EDL forms. Because OTEXT= DJDE parameters force a new sheet of paper, Oracle strongly recommends that you run DMGMERGE with the GLOBAL STRIPOTEXT=YES parameter. For more information about STRIPOTEXT, see "STRIPOTEXT=" on page 399.

Documerge Forms with Graphic DJDEs

In Metacode form print streams, graphics are rendered in two ways:

- **INLINE —** the graphic is part of the composed form. This is referred to as a *Graphic*. The DJDE packet in the form contains a DJDE GRAPHIC=... record.

  According to Xerox rules, the GRAPHIC=... DJDE record must be the only record in its packet. Also, Xerox requires that these packets and their associated raster pattern data be the last records in the data stream for each form-page. (The graphic does not have to print at the bottom of the page — where it prints is not related to where the graphic records occur in the data stream.)

  Documerge requires that each graphic DJDE on a form-page be bracketed by the DMGIMAGBEG/END records. When it normalizes a form, DMGMETP brackets any graphic DJDEs with the DMGIMAGBEG/END comment records:

```
%%%DMGIMAGBEG%%% Beginning of Xerox Image or Graphic
GRAPHIC DJDE record
graphic data (raster pattern) records
%%%DMGIMAGEND%%% End of Xerox Image or Graphic
```

  During concatenation, or anytime Documerge places two or more forms on the same page, Documerge places any graphic DJDEs in temporary storage and writes them back into the form data stream as the last items before the end of the page.

- **.IMG File —** the graphic is not part of the composed form, but resides on the printer. The DJDE packet in the form contains an IMAGE=... parameter.

  When processing image DJDEs, DMGMERGE

  - Inserts a **hex 0101** dummy Metacode record after each IMAGE DJDE to ensure that two DJDE packets don't process back-to-back.
  - Requires that IMAGE parameter must be the *first* parameter in a DJDE record, but allows other parameters to follow the IMAGE parameter in the same record.
  - Allows multiple IMAGE DJDE records in the same packet.
  - Allows other DJDE records to precede or succeed the IMAGE DJDE record in the same packet.
  - No longer requires %%%DMGIMAGBEG%%% and %%%DMGIMAGEND%%% comment records. (They are still a requirement for graphic DJDEs.)

Documerge Guidelines for Metacode Graphics

When preparing graphics for Documerge processing, follow these guidelines:

- If forms from a workstation composition system contain graphics, you must renormalize the forms with DMGMETP using the GRAPHICSONLY= and PRINTDEF= control cards.
- Documerge 3.1 and later releases recognize and shift (if required because of concatenation, logical pages, Overlays, etc.) the **LOGO=(name, vpos units, hpos units...)** format for LOGO DJDEs. However, Documerge does not support the **LOGO=(name,HERE...)** format.

- When creating DJDE statements for graphics, ensure the following:
  - The GRAPHIC parameter must be the *only* parameter in a DJDE packet.
  - Except for the DJDE ID, the text of the DJDE must be coded in ASCII.
  - The word *IMAGE* or *GRAPHIC* must start in the DJDE skip column as specified by the PRINTDEF DJDESKIP value.
  - A GRAPHIC or IMAGE DJDE must not specify the H or HOLD option, or the associated CANCEL or ALTER options.
  - Except for the *,END;* parameter, A graphic or image DJDE must not contain any parameters. Furthermore, the ,END; must be in the same record as the IMAGE or GRAPHIC identifier. (While placement of **,END;** on a separate record is valid Metacode, Documerge requires placement of both on the same record.)

To use normalized Metacode graphics with the Documerge **concatenation** or **Overlay** features, follow these guidelines:

- If a form contains an IMAGE= parameter, the DMGMETP BOTTOM= parameter value must specify where the .IMG will end on the page.
- Xerox, and consequently Documerge, allows only one FRM (FORMS DJDE) per page-face for concatenation, Overlay, or logical page placement. If you need more than one graphical element on a page, consider using an IMG (IMAGE DJDE), although IMGs typically cause slower printing speed.

CAUTION!

If you concatenate or specify logical pages using several small forms that contain one or more IMAGE and/or GRAPHIC DJDE specifications, the page might become unprintable, and Documerge will not report this fact.

- Documerge supports EDL forms that use the BFORM DJDE parameter, which specifies the printing of a FRM only on the back (even) side of the page.

  However, Documerge does not support concatenation or shifting for BFORM, and will not change BFORM to match a different page parity. Therefore, specify the EVN (even) Print Option for forms coded with the BFORM DJDE.

Documerge Forms with Metacode Highlight Color Graphics

To print highlight color graphic DJDEs in black and white on non-color printers, ensure that the GLOBAL parameter STRIPCOLOR= has a value of *YES* (for details, see "STRIPCOLOR=" on page 398), and that the PRINTDEF has the COLOR= parameter set to *NO* (for details, see "COLOR=" on page 100.

DMGMETP EXEC Parameters

Documerge uses the following EXEC parameters in the DMGMETP program.

Parameter	Value
NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. The memory allocated for LM/MM is the number you specify multiplied by the block size of the WRKFIL. (See "WRKFIL" on page 378.) Specify a number from **2** to **2048**. You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. If the memory allocation is smaller than the amount of data, LM/MM uses the WRKFIL for secondary space. (See "WRKFIL" on page 378.) The WORKBUFF= value must be at least twice the value of the WRKFIL block size. The WORKBUFF= value must not exceed the WRKFIL block size multiplied by 2048. Valid values are: **nnnK** nnn units of 1024 bytes **nnnM** nnn units of 1,048,576 bytes (1024x1024)
	You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=.
	If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)

DMGMETP Files

The following are descriptions of DMGMETP Files found in "DMGMETP Normalization JCL" on page 53.

Input Files

- **INFILE**
  A file containing the Metacode form produced by the XICS, HFDL, or DPLMAIN (DCF/PLUS) composition system or the EDL member to be renormalized.
- **INDEXFL**
  A file containing the Boilerplate Index Packet produced by DMGXPRN, DMGHPRN or ISIBPSTG (DCF/PLUS) composition interface.
- **PEDEF**
  A file containing PEDEF objects. PEDEFs reside in the PELIB sub-library.

NOTE

The PEDEF file must be the same one used during the composition process (DCF/PLUS only).

- **FONTMETA**
  A font library file containing Metacode fonts in an AFP font format. The font objects reside in the FONTMETA sub-library.

NOTE

This must be the same font library used during the composition process.

- **WRKFIL**
  An internal work file.
- **SYSIN**, DMGMETP Control Cards

  BOTTOM=

  A numeric value between 0 and 9999, specified in dots, designating the logical bottom of the physical page. Optional parameter defaulting to a value equal to the physical bottom of page. This parameter is used to control the logical bottom of forms to be used in Documerge.

IMPORTANT

When a Metacode form contains graphics that reside on the printer as .IMG files, the BOTTOM= value must indicate where the graphic ends on the form.

To calculate the BOTTOM= values

1. Measure the number of inches from the top of the page to the end of the graphic.
2. Multiply that number by the appropriate PEL size.

If you specify both the CONcatenate and LANdscape Print Options for a Metacode form, you must set BOTTOM= equal to the actual size of the form *including the last line*. A BOTTOM= value that does not include the last line (the artificial bottom) can cause overprinting or an unprintable page.

CODEDEF=
Specifies the name of the CODEDEF used to build translate tables to use for replacement character and variable data translation. A value specified with this parameter overrides the CODEDEF referenced in the PRINTDEF.

COMMONFONTS=
A value of Y (YES) alters the form's font list to match the font list contained in the ENVDEF. A value of N (NO) indicates the form's font list will not be altered. This control card is optional with a default of N (NO). It is recommended that all forms contain a common font list.

When using the Documerge Concatenation or Overlay features, a common font list is mandatory. If a named PDE was used during the HFDL or XICS step, the fonts of the named PDE must be specified in the PDE FONTS= control card. If a PDE is being used, COMMONFONTS= should be set to NO. The document's font list should match the fonts on the printer's PDE.

ENVDEF=
The ENVDEF (Output Environment Definition) in the PEDEF input file. This control card is required if you code COMMONFONTS=YES. The ENVDEF contains the Common Font List. (The number of fonts in the list is limited by:

1) the sizes of the fonts

2) the amount of printer memory allocated for fonts

The printer's memory can handle a maximum of 128 fonts per font list. For Tumble printing, inverse fonts comprise half of the maximum.) Also, the ENVDEF can call a PRINTDEF (Printer Definition).

- **SYSIN**, DMGMETP Control Cards (continued)

FONTLIB DDNAME=
Specifies the file name in the JCL control statement containing the Metacode font descriptions. The default is FONTMETA.

FGRPDEF=
The FGRPDEF (Font Group Definition) in the PEDEF input file. This control card is required if you do not specify a PRINTDEF that calls a FGRPDEF. You code the PRINTDEF either:

1) directly, with the PRINTDEF= control card

or

2) indirectly, with the ENVDEF= control card.

GRAPHICSONLY=

A value of YES normalizes the Interpress graphics only. Normalized forms from the PC composition systems must be renormalized through DMGMETP if the forms contain graphics.

PDE FONTS=(XXXXXX,XXXXXX,XXXXXX)
Is used to specify a font list when normalizing a form. Use with a global PDE (a stand-alone file with a .PDE extension stored on the printer); PDE FONTS= provides DMGMETP with information about fonts and character widths. A maximum of 128 fonts can be specified.

PELIB DDNAME=
Specifies the file name in the JCL control statement containing the PEDEF objects; ENVDEF, TRANSFDEF, PRINTDEF, FGRPDEF and CODEDEF. The default is PEDEF.

PRINTDEF=
The PRINTDEF (Printer Definition) in the PEDEF input file. This control card is required if

1) you do not code the ENVDEF= control card

2) the ENVDEF that you specify in the ENVDEF= control card does not call a PRINTDEF

PROCESS
This command starts DMGMETP and is mandatory.

Output Files

- **OUTFILE**
  A file containing the normalized form and Boilerplate Index Packet ready for EDL loading. This file's recommended physical characteristics are:
  - Record format = VB
  - Record length = 155 (the Metacode LRECL cannot exceed 256)
  - Blocksize = 3000
- **MESSAGE**
  DMGMETP messages

Boilerplate Index Packet Column Definitions: DMGMETP

Example Index Begin Record

```
    ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00001 %%%DMG2NDXBEG%%% XXXX #### #### #### #### #### XXXX CR ##/##/## ##.## XXXXXXXX
```

Line 1 Index Begin Record

If no BPSDs exist in a form, its Boilerplate Index Packet contains only this record and the Index End Record.

Columns 2-17 Index Begin Identifier

Columns 19-22 Composition System Identifier. Valid values are:

DCF3	Produced by DPLPROF/ISIBPSTG
XPRN	Produced by DMGXPRN
HFDL	Produced by DMGHPRN

Columns 24-27 Number of Single-Sided Images (SSIs).

Columns 29-32 Lowest dot address, logical bottom of portrait page.

Columns 34-37 Highest scan address, logical bottom of landscape page.

Columns 39-42 Lowest dot address, physical bottom of portrait page.

Columns 44-47 Highest scan address, physical bottom of landscape page.

Columns 49-52 The page orientation. Valid values are:

LAND	Landscape
INVL	Inverse landscape
PORT	Portrait
INVP	Inverse portrait

Columns 54-70 Creation date and time.

Columns 72-79CODEDEF used to build translate tables for translation of Replacement Characters and variable data.

Sample DMGMETP Normalization Boilerplate

```
     ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8.
00001  %%%DMG2NDXBEG%%% XXXX #### #### #### #### #### XXXX CR ##/##/## ##.## xxxxxxxx
00002   tag name                        length
00003   –                               –     character
00004   –                               –     – mandatory/optional
00005   –                               –     – | delete after use
00006   –                               –     – | – line end
00007   –                               –     – | – | multi-data
00008   –                               –     – | – | – generate
00009   –                               –     – | – | – |
00010   V                               V     V V V V V V
00011  %AGE                             00003 @ O N – N Y
00012  %ADDRESS                         00100 # O Y – N Y
00013  %BIRTHDATE                       00010 – M N – N Y
00014  %%%DMG2NDXEND%%% End of Index Lines -- Text lines follow
00015  %%%D97DJDEBEG%%%
00016  $$XEROX JDL=DPLJDL,JDE=ONLINE,;
00017  $$XEROX FONTS=(XXXXXX,XXXXXX,XXXXXX,XXXXXX,XXXXXX,XXXXXX,FORMSX),;
00018  $$XEROX DUPLEX=NO,END;
00019  %%%D97DJDEEND%%%
00020  %%%ISIPAGEDEF%%% XXXX
00021  %%%ISIFONTDEF%%% 000/XXXXXX  001/XXXXXX  002/XXXXXX  003/XXXXXX
00022  %%%DMGREPLBEG%%% Beginning of Metacode records with Replacement Characters
00023  variable text
00024  variable text
00025  variable text
00026  variable text
00027  variable text
00028  variable text
00029  variable text
00030  %%%DMGREPLEND%%% End of Metacode records with Replacement Characters
00031  Beginning of Metacode records with no Replacement Characters (fixed text)
00032  fixed text
00033  fixed text
00034  fixed text
00035  fixed text
00036  fixed text
00037  REPORT END
```

Sample DMGMETP Normalization Boilerplate Field Descriptions

Lines 2-10 Comment records used as column titles.

Lines 11-13 BPSD command records.

Columns 3-32 Tag name assigned to BPSD command.

Columns 34-38 Length assigned to BPSD command.

Column 40 Replacement Character assigned to BPSD command.

Column 42 Mandatory (M) or Optional (O).

Column 44 Delete after use: Yes (Y) or No (N).

Column 46 Line-end character.

Column 48 Multi-data: No (N) or numeric value of 0-9.

Column 50 Generates Replacement Characters: YES (Y) or NO (N).

NOTE: The order of the tags in the index follows the order in which they were found in the composition source. Refer to *Composing Forms for Documerge* for BPSD command coding restrictions.

Line 14 Index End Record.

If no BPSDs exist in a form, its Boilerplate Index Packet contains only this record and the Index Begin Record.

Columns 2-17 End Index Identifier.

Columns 18-57 Constant comment field.

Lines 15-19 DJDE Packet produced by DCF/PLUS, XICS or HFDL processing.

Line 20 ISIPAGEDEF
The page definition comment record indicating the orientation of the page. This line is placed in the form by DMGMETP. Valid values are PORT or LAND.

Line 21 ISIFONTDEF
The font definition comment record indicating the sequential number of the font in the font list followed by the font name. This line is placed in the form by DMGMETP.

Line 22 DMGREPLBEG
Indicates the beginning of replacement records generated by the BPSD tag replacement characters. This line is placed in the form by DMGMETP.

Lines 23-29 Metacode records with Replacement Characters.
DMGMERGE reads these records to insert the variable data.

Line 30 DMGREPLEND to
Indicates the end of the replacement records. This line is placed in the form by DMGMETP.

Lines 31-36 The fixed text (Metacode records with no Replacement Characters).

Line 37 Report End.

Metacode Example

The Normalization program DMGMETP concatenates the Boilerplate Index Packet to the form during Normalization processing. When the composition source contains BPSD tags of:

```
        BPSD NAME=AGE LENGTH=3
        BPSD NAME=ADDRESS LENGTH='25 4' DELETE=Y
        BPSD NAME=BIRTHDATE LENGTH=10 TYPE=M
```

and is processed by DCF (using DCF/PLUS) using DPLPROF/ISIBPSTG, the Metacode Boilerplate Index Packet in "Boilerplate Index Packet produced by DCF/PLUS" on page 64 is generated.

Boilerplate Index Packet produced by DCF/PLUS

```
     ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00001  %%%DMG2NDXBEG%%% DCF3    1 2264 1717 2264 1717 Created 02/26/91 14:07
00002   tag name                        length
00003   —                               —     character
00004   —                               —     — mandatory/optional
00005   —                               —     — | delete after use
00006   —                               —     — | — line end
00007   —                               —     — | — | multi-data
00008   —                               —     — | — | — generate
00009   —                               —     — | — | — |
00010   V                               V     V V V V V V
00011  %AGE                             00003 @ O N — N Y
00012  %ADDRESS                         00100 # O Y — N Y
00013  %BIRTHDATE                       00010 — M N — N Y
00014  %%%DMG2NDXEND%%% End of Index Lines -- Text lines follow
```

ISIBPSTG, DMGHPRN and DMGXPRN generate an Index Begin Record and Index End Record when the composition source contains no BPSD commands. This index packet is then passed to DMGMETP, which concatenates it to the Metacode data stream, producing the output shown in "Sample DMGMETP Normalization Boilerplate" on page 62.

DMGAFPP

DMGAFPP is the Normalization program for AFP data streams. It reviews the space allocation for blanks in replacement data, redefines the widths of the blanks, and breaks any records exceeding the 8K limit.

DMGAFPP can be used to renormalize forms previously normalized by the Documerge version 1 Normalization program.

If you use Oracle Template Technology, refer to the Tag Table for additional Normalization information.

DMGAFPP JCL

```
//DMGAFPP   ** put your job card here **
//*
//* ****************************************************************
//* **                                                            **
//* **         DOCUMERGE V.3.2 AFP normalizer                     **
//* **                                                            **
//* ****************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGAFPP EXEC PGM=DMGAFPP,REGION=4M,
//  PARM=' / WORKBUFF=500K'
//INDEXFL  DD DUMMY,
//            DCB=(RECFM=VB,LRECL=8205,BLKSIZE=8209)
//DOCUMENT DD DSN=documerg.v03r02.list3820(MEMBER),
//            DISP=SHR
//MASTER   DD DSN=documerg.v03r02.afpnorml(MEMBER),
//            DISP=SHR
//PEDEF    DD DSN=your.documerg.v03r02.pelib,
//            DISP=OLD
//FONTLIB  DD DSN=your.documerg.v03r02.afp.fontlib,
//            DISP=SHR
//MESSAGE  DD SYSOUT=*,DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD SYSOUT=*
//WRKFIL   DD DSN=&&WRKFIL,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                 HALF TRACK
//SYSIN DD *
-
PROCESS -
         INFILE DDNAME=DOCUMENT    -
         ENVDEF=3820
-        FONTLIB=FONTLIB  -
-        INDEXFL=INDEXFL  -
-        OUTFILE DDNAME=MASTER     -
/*
//
```

DMGAFPP EXEC Parameters

Parameter	Value
NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. The memory allocated for LM/MM is the number you specify multiplied by the block size of the WRKFIL. (See "WRKFIL" on page 378.) Specify a number from **2** to **2048**. You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. If the memory allocation is smaller than the amount of data, LM/MM uses the WRKFIL for secondary space. (See "WRKFIL" on page 378.) The WORKBUFF= value must be at least twice the value of the WRKFIL block size. The WORKBUFF= value must not exceed the WRKFIL block size multiplied by 2048. Valid values are: **nnnK** nnn units of 1024 bytes **nnnM** nnn units of 1,048,576 bytes (1024 — 1024)
	You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=.
	If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)

DMGAFPP Files

The following are descriptions of DMGAFPP files found in "DMGAFPP JCL" on page 65.

Input Files

- **DOCUMENT**
  A file containing the AFP form produced by DCF or OGL composition system or the EDL member to be renormalized. This file's recommended physical characteristics are:
  - Record format = VBM
  - Record length = 8205
  - Blocksize = 8209
- **INDEXFL**
  A file containing the Boilerplate Index Packet produced by DMGOPRN or ISIPROF/ISIBPSTG composition interface. This file's recommended physical characteristics are:
  - Record format = VBM
  - Record length = 8205
  - Blocksize = 8209

- **FONTLIB**
  A font library file containing the AFP font descriptions.

NOTE

This must be the same FONTLIB file used during the composition process.

- **PEDEF**
  A file containing PEDEF objects. Required if COMMONFONTS=Y is specified. PEDEFs reside in the PELIB sub-library.
- **SYSIN**, DMGAFPP Control Cards

  BOTTOM=

  Optional parameter defaulting to a value equal to the physical bottom of page. Used to control the logical bottom of forms to be used as Documerge Overlays.

  Contains a numeric value between 0 and 9999 designating the logical bottom of the physical page. To calculate the BOTTOM= value, measure from the top of the page and multiply the number of inches from the top by the PEL size. If this value is omitted, the actual size of the form is used.

  For Overlays, you typically code BOTTOM=0 to indicate the form takes no vertical size (usually done for Overlays).

  For non-Overlay forms, you typically omit BOTTOM=.

TIP

you can code a BOTTOM= value for any form.You can even specify that a form takes more space than it actually does, which is useful for adding extra white space between concatenated forms or after Overlay forms.

  COMMONFONTS=

  A value of Y (YES) alters the form's font list to match the font list contained in the ENVDEF. A value of N (NO) indicates the form's font list will not be altered. This is an optional parameter with a default of N (NO). It is recommended that all forms contain a common font list. The use of a common font list is not required in an AFP environment.

  ENVDEF=

  The environment definition contained in the PEDEF file. The ENVDEF is required when COMMONFONTS=Y is specified.

  DESCENDER

  Optional. Specifies if the computed vertical size of the form is to include descenders in the fixed text. Descenders are characters (typically lowercase letters) that extend below the baseline, such as **y** and **p**. There is no value or equal sign coded with this keyword.

  Code the single keyword **DESCENDER** to include fixed-text descenders; omit the word to not include the descenders. Typically, you don't code this keyword unless you get undesirable overprinting when concatenating a form that contains descenders. In that case, you code **DESCENDER** and re-normalize the form.

  FONTLIB DDNAME=

  Specifies the name of the JCL control statement containing the AFP font descriptions. Optional with default of FONTLIB.

INDEXFL DDNAME=

Optional. Specifies the name of the JCL control statement containing the Boilerplate Index Packet produced by the composition interface. If there is no INDEXFL, specify **DD DUMMY** for the file in the JCL. You can also code just INDEXFL=. (INDEXFL DDNAME=INDFILE and INDEXFL=INDFILE are equivalent.)

- **SYSIN**, DMGAFPP Control Cards (continued)

  INFILE DDNAME=

  Specifies the name of the JCL control statement containing the AFP form produced by the composition system. Optional with default of DOCUMENT. You can also code just INFILE=. (INFILE DDNAME=INPUT and INFILE=INPUT are equivalent.)

  ORIENT=

  Optional. Specifies the orientation of a form. Normally this parameter is not required; if omitted, DMGAFPP determines the orientation based on the most used text orientation found in the form.

  Valid values are

PORT	portrait
LAND	landscape
INVP	invervse portrait
INVL	inverse landscape.

  OUTFILE DDNAME=

  Specifies the name of the JCL control statement containing the normalized form. Optional with default of MASTER. You can also code just OUTFILE=. (OUTFILE DDNAME=MYOUT and OUTFILE=MYOUT are equivalent.)

  PELIB DDNAME=

  Specifies the name of the JCL control statement containing the PEDEFs. Optional with default of PEDEF. Required if COMMONFONTS=Y is specified.

  RELMOVE=

  Optional. Determines whether absolute moves (AMIs and AMBs) are converted to relative moves (RMIs and RMBs). This conversion results in accumulated rounding errors if forms are composed at 240x240 dpi but printed at 300x300 dpi; however, Documerge Concatenation is slightly faster if this conversion is performed.

  Valid values are

Y (YES)	AMIs and AMBs are converted to RMIs and RMBs. **Y** is the default.
N (NO)	AMIs and AMBs are not converted to RMIs and RMBs. Specify **N** (NO) to avoid rounding errors when printing 240x240 dpi forms on a 300x300 dpi printer.

Output Files

MASTER

A file containing the normalized form and Boilerplate Index Packet ready for loading to the EDL. Recommended physical characteristics are

Record format	VBM
Record length	8205
Blocksize	8209

MESSAGE

A file containing messages generated by DMGAFPP.

WRKFIL

An internal work file.

Boilerplate Index Packet Column Definitions: DMGAFPP

Documerge 3.1 and later releases support 1440 pels per inch AFP print streams, and AFP requires a 5-digit field to describe values like the pel specifications for a page. To accommodate these larger specifications, Documerge 3.1 and later releases use 5 columns (instead of 4 columns preceded by a leading space) for the following fields:

- Logical bottom of page in pels (columns **36-40** instead of 37-40)
- Logical right-most pel (columns **41-45** instead of 40-45)
- Physical bottom of page in pels (columns **46-50** instead of 47-50)
- Physical right-most pel (columns **51-55** instead of 50-55)

You code each value right-justified in its field. Documerge will interpret leading blanks as zeros, but the last position of any field coded must contain a digit (0 – 9). For example, the value `'00012'` is the same as `'12'`.

Example Index Begin Record

```
....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
        %%%DMG2NDXBEG%%% XXXX aaa bbbbbccccddddeeeee XXXX Cr ##/##/## ##:##
```

Line 1	Index Begin Record
Columns 10-25	Begin Index Identifier.
Columns 27-30	Composition System Identifier; valid values are DCFA produced by ISIPROF/ ISIBPSTG and OGL produced by DMGOPRN.
Columns 32-34	Number of single sided images.
Columns 36-40	Logical bottom of page in pels.
Columns 41-45	Logical right-most pel.
Columns 46-50	Physical bottom of page in pels.
Columns 51-55	Physical right-most pel.
Columns 57-60	Orientation of form: Landscape (LAND); Portrait (PORT), Inverse Landscape (INVL); Inverse Portrait (INVP).
Columns 62-78	Creation date and time.

Sample AFPP Boilerplate Index Packet

```
    ...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
00001         %%%DMG2NDXBEG%%% XXXX aaa bbbbbccccddddeeeee XXXX Cr ##/##/## ##:##
00002         tag name                          length
00003         —                                 —     character
00004         —                                 —     — mandatory/optional
00005         —                                 —     — | delete after use
00006         —                                 —     — | — line end
00007         —                                 —     — | — | multi-data
00008         —                                 —     — | — | — generate
00009         —                                 —     — | — | — |
00010         V                                 V     V V V V V V
00011         %AGE                              00003 @ O N — N Y
00012         %ADDRESS                          00100 # O Y — N Y
00013         %BIRTHDATE                        00010 — M N — N Y
00014         %%%DMG2NDXEND%%% End of Index Lines -- Text lines follow
```

Lines 2 - 10 Comment records used as column titles for index records.

Lines 11 - 13 BPSD index records

NOTE: The order of the tags in the index follow the order in which they were found in the composition source. Refer to *Composing Forms for Documerge* for BPSD command coding restrictions.

Columns 11-40 Tag name assigned to BPSD command.

Columns 42-46 Length assigned to BPSD command.

Column 48 Replacement Character assigned to BPSD command.

Column 50 Mandatory (M) or Optional (O).

Column 52 Delete after use: Yes (Y) or No (N).

Column 54 Line end character.

Column 56 Multi-data: No (N) or numeric value of 0-9.

Column 58 Generates replacement characters: YES (Y) or NO (N).

Line 14 Index End Record

Columns 10-25 Index End Identifier.

Columns 27-65 Constant comment field.

AFP Example

The Normalization program DMGAFPP concatenates the Boilerplate Index Packet to the form during Normalization processing.

When source code contains the following BPSD commands:

```
        BPSD NAME=AGE LENGTH=3
        BPSD NAME=ADDRESS LENGTH='25 4' DELETE=Y
        BPSD NAME=BIRTHDATE LENGTH=10 TYPE=M
```

and is processed by:

- DCF using ISIPROF/ISIBPSTG
- OGL using DMGOPRN

and is normalized with DMGAFPP, the AFP Boilerplate Index Packet in "AFPP Boilerplate Index Packet" on page 71 is generated.

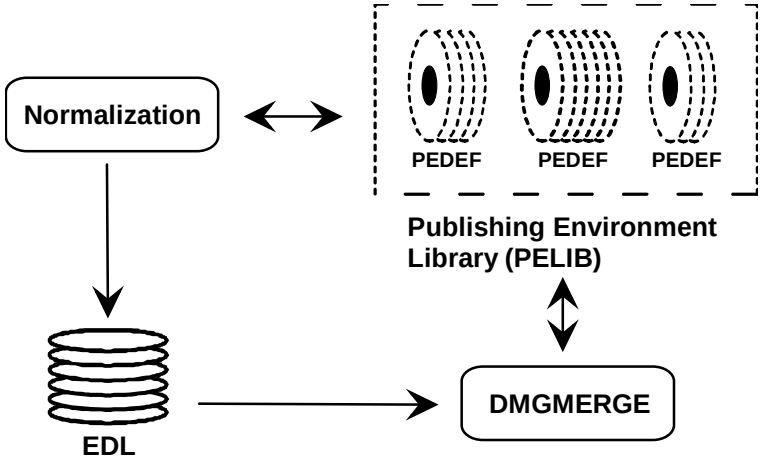
AFPP Boilerplate Index Packet

```
      ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
00001           %%%DMG2NDXBEG%%% AFPP   1  2081    0 2081    0 PORT Cr 01/20/91 21:32
00002            tag name                          length
00003            —                                 —     character
00004            —                                 —     — mandatory/optional
00005            —                                 —     — | delete after use
00006            —                                 —     — | — line end
00007            —                                 —     — | — | multi-data
00008            —                                 —     — | — | — generate
00009            —                                 —     — | — | — |
00010            V                                 V     V V V V V V
00011           %AGE                               00003 @ O N — N Y
00012           %ADDRESS                           00100 # O Y — N Y
00013           %BIRTHDATE                         00010 — M N — N Y
00014          %%%DMG2NDXEND%%% End of Index Lines -- Text lines follow
```

The programs ISIBPSTG or DMGOPRN generate an Index begin and end record when the composition source does not contain BPSD commands.

The Publishing Environment Library (PELIB)

The Publishing Environment Library (PELIB) stores your **Publishing Environment Definitions (PEDEFs)**. PEDEFs are tables that tell the DMGMERGE program about your printer's characteristics. PEDEFs contain information such as the printer type, its software parameters, and the print-job specifications.



Documerge comes with a starter set of PEDEFs, in their source format. You must compile these PEDEFs into object format. When you store these starter PEDEFs in object format, they set up a default publishing environment.

You store PEDEF object formats in two Documerge libraries:

- The **Publishing Environment Library** (**PELIB**) contains compiled PEDEFs
- The **Font Library** (**FONTLIB**) contains AFP-like fonts for Metacode printers.

You can modify the starter PEDEFs to meet your specific needs, or you can create new PEDEFs. You can create as many of each PEDEF as you need.

IMPORTANT!

Do not change a delivered source file.

Copy the source and modify the copy.

Creating and Maintaining Your PELIB

The Documerge Publishing Environment Definitions (PEDEFs) and their Publishing Environment Library (PELIB) are created and maintained through two Oracle utilities:

- DPLDUTL
- DFXPHBE

The number of PELIBs your company creates and maintains depends on your company's Documerge implementation plans. You may have a PELIB already, depending on whether your company has installed the Oracle product DCF/PLUS.

A PELIB can be created as part of the Documerge product installation. Your company may have to allocate DASD space for this operation. Contact your technical support group for more information.

The initial creation of a Documerge PELIB is a multi-step process.

(1) Allocate the PELIB disk storage using the procedures at your company.

(2) This step only has to be done once for each PELIB when it is first created.

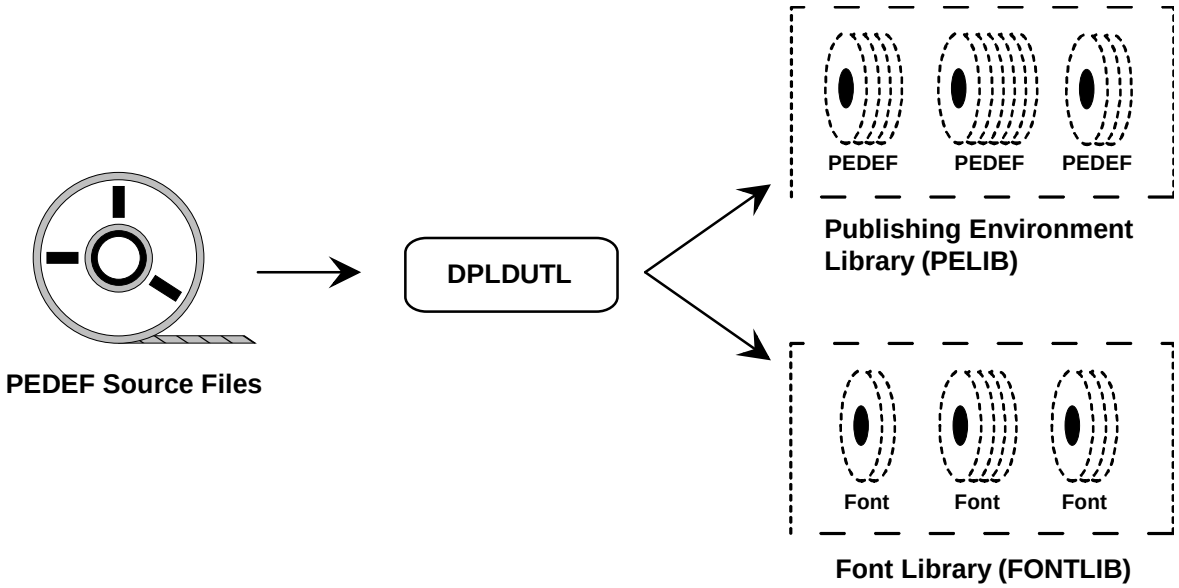
(3) Add new PEDEF objects to the newly created PELIB, using the DPLDUTL utility.

Refer to Sections 4 and 5 of the *PEDEFs reference* for complete details of creating and modifying PEDEFs.

Refer to *Installing Documerge* for more information and sample JCL of how this PELIB was created and initially loaded with the demonstration PRINTDEF and MERGEDEF PEDEFs.

The DPLDUTL Utility

You use the DPLDUTL utility to compile PEDEF object files from PEDEF source files (supplied with your Documerge installation). You also use DPLDUTL to compile fonts, and to store the compiled PEDEFs and fonts in their libraries.



All creation and maintenance of the PEDEFs and the PELIB must be done through DPLDUTL.

The DFXPHBE Utility

DFXPHBE generates reports on the contents of PEDEF object files. It also reverse-compiles PEDEF object files into the original source format.

Publishing Environment Definitions (PEDEFs)

Many Oracle products use the PELIB. Therefore, PEDEF functions and features may differ between products. This section describes the PEDEFs that apply directly to Documerge:

- CODEDEF

  Defines character translation from EBCDIC to ASCII.

- ENVDEF

  Defines processing options, such as the font list, for the Documerge normalization programs.

- FGRPDEF

  Defines the font group used by DMGMETP normalization.

- MERGEDEF

  Defines processing options for the DMGMERGE program.

- PRINTDEF

  Defines printer characteristics such as type, software parameters, and print job specifications.

This section presents an overview of the commands that create the PEDEFs, and the parameters and sub-commands for customizing PEDEFs to your environment.

NOTE

In the lists of PEDEF parameters and subcommands in this section, each PEDEF's NAME= parameter is listed first, followed by the other parameters and subcommands in alphabetical order. You do not need to code these items in any special order when creating a PEDEF.

CODEDEF (Font Translation Table)

Each FONTDEF entry in the FGRPDEF calls a CODEDEF. A CODEDEF is a table that defines the EBCDIC-to-ASCII character translation. This table is required for Metacode applications. A complete CODEDEF is provided at installation.

The fonts of most non-IBM printers are in ASCII format, but DCF, for example, uses EBCDIC. A Font Translation Table relates printable ASCII values to the corresponding EBCDIC values in your DCF source. The CODEDEF command sets up and identifies a Font Translation Table.

DMGMERGE uses the CODEDEF to translate the EBCDIC values of Replacement Characters in a Boilerplate Index Packet to the ASCII values of the corresponding Replacement Characters in the form. DMGMERGE also uses the CODEDEF to translate variable data from EBCDIC to ASCII.

The CODEDEF of the font used for Replacement Characters is specified at normalization, with the DMGMETP parameter CODEDEF=. If the CODEDEF= parameter is not specified, the CODEDEF for the first font in the form is used. If the CODEDEF does not contain the same EBCDIC characters, the ASCII translation will be incorrect.

CODEDEF Parameters

Parameter	Value
NAME=	The name of this CODEDEF. The delivered value is **ASCII9**. When the PEDEF object creation utility, DPLDUTL, is executed, this name is prefixed with **DC**.
POINTDEF	The sub-command that assigns ▪ a character identifier (or code point) for each character ▪ the ASCII and EBCDIC value for each character. There are only 256 possible 1-byte values to relate to the many alphanumeric, special, and non-Roman alphabet characters. You should define all 256 possible POINTDEFs. The POINTDEF sub-command accepts the following parameters:
	CAUTION: There should be a one-to-one correspondence between the FONTPT and CODEPT parameters. Do not map more than one FONTPT to the same CODEPT.
	FONTPT= The ASCII hex value of the character in the host-font form. A valid FONTPT is one for which there is a valid character in the font.
	IDEN= The 8-byte IBM name for the character. This name is an ISO (International Standards Organization) naming standard. Refer to the *IBM 3800 Printing Subsystem Model 3 Font Catalog* (SH35-0053) and the *About Type Technical Reference for Digitized Type* (G544-3516).
	CODEPT= The EBCDIC value used to specify the character in your DCF source.

Sample CODEDEF

Following is a portion of a sample CODEDEF named UDKISO. This sample relates the host-font form of the blank and exclamation point to EBCDIC.

Sample USA UDKISO CODEDEF Source for Xerox Decentralized Printers

```
- The CODEPTs listed in this file were picked to match, where possible,
- the CODEPT for that character in IBM's T1GI0395 codepage.
- This is the USA EBCDIC Codedef for Xerox Decentralized Printer ISO fonts.
- To print characters with an accent character, use the CODEPTs associated
- with the font characters in the range x'C0' - x'CF'.  You must follow
- the accent immediately with a non-blank character for correct results.
- The printer will determine whether to use a low accent or high accent
- by looking at the character you print after the accent character.
-
CODEDEF NAME=UDKISO
POINTDEF FONTPT=X'00'               CODEPT=X'00'
POINTDEF FONTPT=X'01'               CODEPT=X'00'
  ...
POINTDEF FONTPT=X'30' IDEN=ND100000 CODEPT=0
POINTDEF FONTPT=X'31' IDEN=ND010000 CODEPT=1
POINTDEF FONTPT=X'32' IDEN=ND020000 CODEPT=2
POINTDEF FONTPT=X'33' IDEN=ND030000 CODEPT=3
POINTDEF FONTPT=X'34' IDEN=ND040000 CODEPT=4
POINTDEF FONTPT=X'35' IDEN=ND050000 CODEPT=5
POINTDEF FONTPT=X'36' IDEN=ND060000 CODEPT=6
POINTDEF FONTPT=X'37' IDEN=ND070000 CODEPT=7
POINTDEF FONTPT=X'38' IDEN=ND080000 CODEPT=8
POINTDEF FONTPT=X'39' IDEN=ND090000 CODEPT=9
POINTDEF FONTPT=X'3A' IDEN=SP130000 CODEPT=:
POINTDEF FONTPT=X'3B' IDEN=SP140000 CODEPT=;
POINTDEF FONTPT=X'3C' IDEN=SA030000 CODEPT=<
POINTDEF FONTPT=X'3D' IDEN=SA040000 CODEPT=X'7E'
POINTDEF FONTPT=X'3E' IDEN=SA050000 CODEPT=>
POINTDEF FONTPT=X'3F' IDEN=SP150000 CODEPT=?
POINTDEF FONTPT=X'40' IDEN=SM050000 CODEPT=X'7C'
  .... A COMPLETE CODEDEF HAS 256 ENTRIES
  ....
  ....
END
```

ENVDEF (Output Environment Definition)

The ENVDEF (Output Environment Definition) describes processing options, such as the font list used during Normalization, for the Documerge programs. For example, if the DMGAFPP or DMGMETP parameter is COMMONFONTS=YES, the font list contained in the ENVDEF is invoked.

An ENVDEF is provided at installation. You can modify it or create others to suit your needs.

The ENVDEF is used by:

- the Documerge Normalization programs DMGAFPP and DMGMETP
- the COMMONFONTS command in the DMGMERGE program

You can use the ENVDEF to specify a default PRINTDEF (Printer Definition) for the DMGMETP program and the COMMONFONTS command. In turn, you can use this default PRINTDEF to specify a default FGRPDEF for DMGMETP and COMMONFONTS.

NOTE

The DMGAFPP program does not use a PRINTDEF or a FGRPDEF.

Default PRINTDEF

The ENVDEF determines the default PRINTDEF for the DMGMETP program and the COMMONFONTS command.

To Specify a Default PRINTDEF for DMGMETP and COMMONFONTS

1. Create or modify the ENVDEF with the DPLDUTL utility.
2. Code the PRINTDEF= parameter in the ENVDEF.
3. Code the ENVDEF in the ENVDEF= parameter for DMGMETP and COMMONFONTS.

The default PRINTDEF is invoked automatically.

IMPORTANT!

If you do not code the PRINTDEF= parameter in the ENVDEF, you must code the PRINTDEF= parameter in DMGMETP.

If you do not code the PRINTDEF= parameter in the ENVDEF and you code the ENVDEF in the COMMONFONTS parameter ENVDEF=, you must code the COMMONFONTS parameter FGRPDEF= also.

Default FGRPDEF

The default PRINTDEF can contain a default FGRPDEF for the DMGMETP program and the COMMONFONTS command.

When you create or modify the ENVDEF with the DPLDUTL utility, you can specify the default FGRPDEF by coding the PRINTDEF= parameter with a PRINTDEF that contains the default FGRPDEF.

The default FGRPDEF is invoked automatically when you code the ENVDEF in the ENVDEF= parameter for DMGMETP and COMMONFONTS.

IMPORTANT!

If the ENVDEF does not contain a default PRINTDEF that has a default FGRPDEF, you must code the FGRPDEF= parameter in DMGMETP.

If the ENVDEF does not contain a default PRINTDEF that has a default FGRPDEF, and you code the ENVDEF in the COMMONFONTS parameter ENVDEF=, you must code the COMMONFONTS parameter FGRPDEF= also.

ENVDEF Parameters

Parameter	Value
NAME=	The name of this ENVDEF. This is the name that the programs recognize. Default values are: ▪ **AFP** (AFP printers) ▪ **97ENV** (Metacode printers) When the PEDEF object creation utility, DPLDUTL, is executed, the object name is prefixed with **DE**.
DFLTFONT=	The default font used when a font is requested but is not available. Generally, this font is used when a particular orientation of a font is requested but is not available. Valid input is any allowable host font name up to 20 characters. The default font should be a small font that is always available on the printer.
DUPLEX=	(For Metacode printers only). Indicates whether a form is printed on both sides of the sheet (duplex) or on one side (simplex). Valid values are: **NO** Simplex **NONE** The DUPLEX= parameter will not appear in the Metacode DJDE packet. **YES** Duplex This is the delivered value. If TUMBLE=YES, then this value must be YES.
ENVDEF=	The name of another ENVDEF on which this ENVDEF is modeled. This ENVDEF's parameters override those in the model. When this parameter is not specified, DPLDUTL uses the ENVDEF in the NAME= parameter.
EXTRAPAGES=	Indicates where blank pages are placed. If the number of pages is not a multiple of 4, blank pages will be generated. Valid values are: **BACK** Blank pages are placed at the back of the document. **BALANCE** Blank pages are balanced between the front and back of the document. **FRONT** Blank pages are placed at the front of the document.

Parameter	Value
FONTS=	(For Metacode printers only). Describes the font list in the initial DJDE packet. This parameter indicates whether to create a user-specified font list, a downloadable font list, or not to create a font DJDE. Valid values are: **AUTO** Create a downloadable FORMAT= or FONTS= DJDE. If more than 16 fonts are referenced in a document or provided in a FONTS= list, and Release 2 Xerox software is available, DPLMAIN will create a FONTS= DJDE record instead of a FORMAT= DJDE record. AUTO is the delivered value. **(font1 font2...)** The output must conform to this font list. The list is contained in parentheses, and the fonts are listed by the host font name (for example: PR111E, not X0PR111E). The number of listed fonts is determined by the hardware configuration of your Metacode printer. If you continue the list on a new record, you must end each line with the continuation character. **NONE** Do not create a FORMAT= or FONTS= DJDE.
IMPOSE=	Specifies imposition printing. Two or more logical pages of a document are printed on one side of a physical page. This results in a document which can be bound in the center and distributed as a book. This is sometimes called signature printing. Valid values are: **NO** No imposition printing is performed. NO is the default value. **YES** Imposition printing is performed. **NOTE:** If you code IMPOSE=YES, you must also code DUPLEX=YES. This causes pages to be printed on both sides of a sheet.
IMPOSEDELTA=	The amount by which the pages on a sheet are shifted in response to the number of pages printed. The larger a document gets, the more the left margin increases and the gutter between the pages decreases. Valid values include any measurement accepted by DCF.
IMPOSELEFT=	The offset of the logical page on the left-hand side of the sheet (on both the front and back). Valid values include any measurement accepted by DCF. This value is usually zero.
IMPOSERIGHT=	The offset of the logical page on the right-hand side of the sheet (on both the front and back). Valid values include any measurement accepted by DCF. This value is usually half the width of the sheet of paper.
INITDJDE=	(For Metacode printers only). Specifies the data format for, or suppresses creation of, the initial DJDE packet. Valid values are: **ASCII** The initial DJDE packet is ASCII. **EBCDIC** The initial DJDE packet is EBCDIC. This is the delivered value. Documerge users must use EBCDIC. **NONE** Suppresses the creation of the initial DJDE packet. Parameters that apply to it are ignored.

Parameter	Value
JDE=	(For Metacode printers only). The initial JDE to be placed in the initial DJDE packet. This file must be available to the printer. Valid input is up to 6 alphanumeric characters. The delivered value is **ONLINE**.
JDL=	(For Metacode printers only). The initial JDL to be placed in the initial DJDE packet. This file must be available to the printer. Valid input is up to 6 alphanumeric characters. The delivered value is **DPLJDL**.
NUFRONT=	(For Metacode printers only). Indicates how the SIDE=NUFRONT DJDE is used in the initial DJDE packet. Valid values are: **NO** The SIDE=NUFRONT DJDE is not used. **NONE** The NUFRONT= parameter does not appear in the DJDE packet. **YES** The SIDE=NUFRONT DJDE is used. YES is the delivered value.
PAGEROT=	Indicates whether the document can rotate from portrait to landscape automatically based on page width and height as defined in the DCF source. This rotation is cumulative with any rotation done as a result of rotated named areas, or any other special processing. Valid values are: **NO** The document does not automatically rotate from portrait to landscape. **YES** The document rotates from portrait to landscape if the page width is greater than the page length. YES is the delivered value.
PREFIX=	Whether the PREFIX file named in the PRINTDEF is attached to the beginning of the output. The contents of the PREFIX file are not processed by DPLMAIN. Valid values are: **NONE** The PREFIX file is not attached. NONE is the delivered value in Metacode environments. **YES** The PREFIX file is attached. The PREFIXDD must be named in the PRINTDEF.
PRINTDEF=	The name of a default PRINTDEF. **IMPORTANT:** If you do not code this parameter and DMGMETP calls this ENVDEF, you must code the PRINTDEF= parameter in DMGMETP. If you do not code this parameter and you code this ENVDEF in the COMMONFONTS parameter ENVDEF=, you must code the COMMONFONTS parameter FGRPDEF= also.
RSTACK=	(For Metacode printers only). Indicates whether the RSTACK record defined in the PRINTDEF is added to the end of the output. Valid values are: **NONE** No PRINTDEF RSTACK is added. **YES** The PRINTDEF RSTACK is added. YES is the default value.
SUFFIX=	Indicates whether the SUFFIX file named in the PRINTDEF is attached to the end of the output. The contents of this file are not processed by the programs. Valid values are: **NONE** The SUFFIX file is not attached. NONE is the delivered value in Metacode environments. **YES** The SUFFIX file is attached. The SUFFIXDD must be named in the PRINTDEF.

Parameter	Value
TRANSDEF=	The name of a TRANSDEF which translates the post-processed device-specific data. If TRANSDEF=NONE (the delivered Metacode value) is specified, no translation is performed.
TUMBLE=	Indicates whether tumble printing is performed. Valid values are: **NO** No tumble printing is performed. The document is printed head-to-head. NO is the delivered value. **YES** The document is tumble printed (head-to-toe). If you code TUMBLE=YES, you must also code DUPLEX=YES so that the page images will appear on one sheet. If you code DUPLEX=NO, each page image is produced in the correct orientation, but on separate sheets. Metacode printers automatically print landscape in a head-to-toe fashion. If you want head-to-head landscape output, you must specify TUMBLE=YES.

Sample ENVDEF

The following figure shows an ENVDEF that defines a Metacode environment where the page automatically rotates based on the page dimensions. All other parameters of the model ENVDEF named 97ENV are used.

Sample ENVDEF Source

```
ENVDEF -                    <---Starts the Environment Definition
   NAME=SAMPLE -            <---Names this Environment Definition
   ENVDEF=97ENV -           <---Model Environment Definition
   TRANSDEF=NONE -          <---TRANSDEF to be used
   PAGEROT=YES              <---Automatic page rotation is in effect
```

FGRPDEF (Font Group Definition)

The FGRPDEF (Font Group Definition) is used only in Metacode environments. A FGRPDEF names a collection of fonts. Use font groups to indicate which fonts are available for which printer types or to give identical fonts for different printers the same name.

The FGRPDEF contains Font Definitions (FONTDEFs) that define the characteristics for all members of a non-IBM font family. Each FONTDEF describes a single member of the font family with information such as size, orientation, character height and width, and rotation.

The FGRPDEF is required for the DMGMETP Normalization program. You can specify the FGRPDEF for DMGMETP in three ways:

- In the ENVDEF= parameter, code the ENVDEF that contains the PRINTDEF that contains the FGRPDEF.
- In the PRINTDEF= parameter, code the PRINTDEF that contains the FGRPDEF.
- Code the FGRPDEF= parameter.

IMPORTANT!

If the ENVDEF does not contain a PRINTDEF that has a FGRPDEF, you must code the FGRPDEF= parameter in DMGMETP.

The FGRPDEF is required for the COMMONFONTS command in the DMGMERGE program. You can specify the FGRPDEF for the COMMONFONTS command in two ways:

- In the ENVDEF= parameter, code the ENVDEF that contains the PRINTDEF that contains the FGRPDEF.
- Code the FGRPDEF= parameter.

IMPORTANT!

If the ENVDEF does not contain a default PRINTDEF that has a default FGRPDEF, and you code the ENVDEF in the COMMONFONTS parameter ENVDEF=, you must code the COMMONFONTS parameter FGRPDEF= also.

The FGRPDEF is required for the DMGMERGE program to determine font names for Tumble or Imposition printing. You can specify the FGRPDEF for DMGMERGE in two ways:

- When you create the MERGEDEF with the DPLDUTL utility, code the PRINTDEF= parameter with a PRINTDEF that contains the FGRPDEF.

  The FGRPDEF is invoked automatically with the PRINTDEF when you code the MERGEDEF= parameter in the MERGE or FILEDEF command.
- Code the name of the FGRPDEF in one of the following:
  - the MERGE command in the DMGMERGE program (the FGRPDEF= parameter)
  - the FILEDEF command in the DMGMERGE program (the FGRPDEF= parameter)
  - the DMG.FDEF.groupname Reserved Tag in the VDR

IMPORTANT!

If the MERGEDEF does not contain a PRINTDEF that has a FGRPDEF, you must code the FGRPDEF in one of the following:

- The MERGE command in the DMGMERGE program (FGRPDEF= parameter)
- The FILEDEF command in the DMGMERGE program (FGRPDEF= parameter)
- The DMG.FDEF.groupname Reserved Tag in the VDR.

You must specify the FGRPDEF where you specify the MERGEDEF. For example, if you code a FGRPDEF for the MERGE command, then you must also code the MERGEDEF in the MERGE command. Otherwise, DMGMERGE ignores the FGRPDEF coded in the MERGE command.

FGRPDEF Parameters

Parameter	Value
NAME=	The name of this FGRPDEF. The delivered value is: 9700 (Metacode printers)
FONTDEF	The FONTDEF sub-command establishes font groups to indicate which fonts are available for each printer type. Furthermore, you can give fonts for different printers the same name. This is convenient if the fonts are externally identical, such as when you print proof images on a printer that uses different technology than the production printer. Each FONTDEF calls a CODEDEF (Font Translation Table) which describes the relation between an ASCII character and the AFP EBCDIC value which references that character. On an IBM Advanced Function Presentation (AFP) printer, fonts are programmatically rotated. On DCF/PLUS devices, the printers call different physical fonts for each rotation. The CODEDEF associates the rotation of a font in the DCF output with the appropriate rotated machine font. This sub-command describes a single member of a font family, using the following FONTDEF parameters: **NAME=** The name (up to six characters) used to create the DCF FONTLIB font and coded font names. This is the name DCF recognizes. **ATTR=** The attribute of the font. Valid attributes are: **N** Normal. Use **N** for logos and signatures. **R** Reversed (white character/black background) **U** Underlined **O** Outlined Obtain this information from your font vendor or a font sample sheet. **CODEDEF=** The CODEDEF that translates this font from EBCDIC to ASCII (the device code). **CODEPAGE=** The codepage to be used with this font. If a new codepage name is entered here, you must use the font conversion program DFXFNCV to create it. This is identified in the FONTLIB by the prefix T1 and the entered name.

Parameter	Value	

	FAMILY=	The family to which the font belongs. This is the family name to be used when defining fonts with the .DF command in a SCRIPT file. For example:
		`family  weight  posture`
		`  |         |         |`
		`  V         V         V`
		`  TITAN   Medium   Italic`
		In this example, you would specify **TITAN** for the FAMILY= parameter value. The family name must be in upper case.
		Valid input is a string up to 32 alphanumeric characters, blanks, and dashes.
		When blanks and dashes are used in the name
		• the string must be preceded by **E**
		• the string must be enclosed in single quotes.
		For example:
		`E'PRESS ROMAN'`

	FONTDEF=	The name of an existing FONTDEF serving as a model for this one. • New FONTDEF parameters override the same ones in the model. • All parameters in the FONTDEF referred to are valid in the new command. • If one is repeated in the new FONTDEF, it overrides the model FONTDEF. Refer to "Sample FGRPDEF Source" on page 89.
	INVLAND=	The host font name of the inverse landscape version of this font. Valid input is a string up to 32 alphanumeric characters, blanks, and dashes. The font must be entered exactly as it appears in the sample. You can also omit this parameter entirely. When blanks and dashes are used in the name: • the string must be preceded by **E** • the string must be enclosed in single quotes For example: `E'Titan10iso-L'`
	INVPORT=	The host font name of the inverse portrait version of this font. Valid input is a string up to 32 alphanumeric characters, blanks, and dashes. The font must be entered exactly as it appears in the sample. You can also omit this parameter entirely. When blanks and dashes are used in the name: • the string must be preceded by **E** • the string must be enclosed in single quotes For example: `E'Titan10iso-L'`

Parameter	Value	
	LAND=	The host font name of the landscape version of this font. Valid input is a string up to 32 alphanumeric characters, blanks, and dashes. The font must be entered exactly as it appears in the sample. You can also omit this parameter entirely. When blanks and dashes are used in the name: • the string must be preceded by **E** • the string must be enclosed in single quotes For example: `E'Titan10iso-L'`
	POINT=	The point size of the font. You can set this value according to local printing standards. Or, you can use this formula to obtain information from the font's sample print sheet: `(NOMINAL CHARACTER HEIGHT / 300) * 72` The value is an integer up to 999. You can round up or down on the number resulting from this formula. It does not affect line spacing and is used by DCF to identify a particular form. Point size is used for identification only.
	NOTE: Logos and signatures can have the same character height and, therefore, the same point size. If you include two fonts of the same point size and weight in the same family, DCF cannot distinguish between them. In this case, assign different point sizes or place the fonts in different families.	
	PORT=	The host font name of the portrait version of this font. Valid input is a string up to 32 alphanumeric characters, blanks, and dashes. The font must be entered exactly as it appears in the sample. You can also omit this parameter entirely. When blanks and dashes are used in the name: • the string must be preceded by **E** • the string must be enclosed in single quotes For example: `E'Titan10iso-L'`
	STYLE=	Indicates whether the font is: **R** Roman or Numeral (use R for logos and signatures) **I** Italic **S** Stressed (slanted Roman)
	Obtain this information from your font vendor or a font sample sheet.	
	WEIGHT=	The weight of the characters in the font. Valid values are: **1** Ultra-light **2** Semi-light **3** Light **4** Demi-light **5** Medium (use 5 for logos and signatures) **6** Demi-bold **7** Bold **8** Semi-bold **9** Ultra-bold
	Obtain this information from your font vendor or a font sample sheet.	

Parameter	Value	
	WIDTH=	The width of the characters in the font. The value can be any integer from 1 to 9. Use this listing to judge the width. You can, of course, use 2, 4, 6, and 8 as well. **1** Ultra-condensed **3** Condensed **5** Normal (use 5 for logos and signatures) **7** Expanded **9** Ultra-expanded
	Obtain this information from your font vendor or a font sample sheet.	
	MONOPITCH=	The type of font, fixed-pitched or proportional. If you already have FONTDEFs for proportional fonts, it is not necessary to add this parameter. If the font is fixed-pitch the tab spacing width is corrected. Valid values are: **Y** Monopitched font **N** Proportional font N is the default. The default is also applicable if the MONOPITCH parameter is not present.

Optional FGRPDEF Parameters

Because the font conversion program, DFXFNCV, determines the data for the following parameters from the font itself, they can be considered optional.

Parameter	Value
BASELINE=	The distance in pels from the bottom of the character cell to its baseline. The BASELINE value can be 0 to 99999 pels. This value is included on the font sample sheet. BASELINE is equal to CELLHT minus TOPBASE.
CELLHT=	The distance in pels from the top of the character cell to its bottom. The CELLHT value can be 0 to 99999 pels. This value is included on the font sample sheet. CELLHT is equal to the sum of BASELINE and TOPBASE. This value includes any leading above and below the character for fonts which contain leading in the character.
TOPBASE=	The distance in pels from the top of the character cell to the baseline of the character. The TOPBASE value can be 0 to 99999 pels. The value is included in the font sample sheet. TOPBASE is equal to CELLHT minus BASELINE.

Sample FGRPDEF

The following sample FGRPDEF (with comments) defines a font group called SAMPLE for a Metacode environment:

Sample FGRPDEF Source

```
FGRPDEF NAME=SAMPLE
- Helvetica fonts
-   We will use HELV as the model for the Helvetica family.  The model
-   contains the information that will be the same for every font in
-   the Helvetica family:  the CODEDEF, CODEPAGE, FAMILY, WIDTH and ATTR.
-   No FONTDEF entry should override this information, to avoid confusion.
FONTDEF NAME=HELV   CODEDEF=ASCII9  CODEPAGE=HELVET  -
                    FAMILY=HELVETICA  WIDTH=5  ATTR=N
- Fontdef entry "H106JP" will contain information about the Helvetica
- 6-point medium normal fonts.
FONTDEF NAME=H106JP FONTDEF=HELV POINT=6 -
                    PORT=H106JP LAND=H106JL INVPORT=H106JI INVLAND=H106JJ -
                    STYLE=R WEIGHT=5
- The "H206JP" entry is just like "H106JP" except it is for bold fonts.
- It uses entry "H106JP" as a model, thereby getting the information from
- the model "HELV" and additional POINTsize and STYLE information from
- "H106JP".
FONTDEF NAME=H206JP FONTDEF=H106JP WEIGHT=7 -
                    PORT=H206JP LAND=H206JL INVPORT=H206JI INVLAND=H206JJ
- The "H306JP" entry is italic.
FONTDEF NAME=H306JP FONTDEF=H106JP STYLE=I -
                    PORT=H306JP LAND=H306JL INVPORT=H306JI INVLAND=H306JJ
- The "H406JP" entry is bold and italic.
FONTDEF NAME=H406JP FONTDEF=H106JP STYLE=I WEIGHT=7 -
                    PORT=H406JP LAND=H406JL INVPORT=H406JI INVLAND=H406JJ
- The "H107JP" entry describes the 7-point medium normal Helvetica Fonts.
FONTDEF NAME=H107JP FONTDEF=HELV POINT=7 -
                    PORT=H107JP LAND=H107JL INVPORT=H107JI INVLAND=H107JJ -
                    STYLE=R WEIGHT=5
- Definition for the 7-point bold fonts.
FONTDEF NAME=H207JP FONTDEF=H107JP WEIGHT=7 -
                    PORT=H207JP LAND=H207JL INVPORT=H207JI INVLAND=H207JJ
- Definition for the 7-point italic fonts.
FONTDEF NAME=H307JP FONTDEF=H107JP STYLE=I -
                    PORT=H307JP LAND=H307JL INVPORT=H307JI INVLAND=H307JJ
- Definition for the 7-point bold italic fonts.
FONTDEF NAME=H407JP FONTDEF=H107JP STYLE=I WEIGHT=7 -
                    PORT=H407JP LAND=H407JL INVPORT=H407JI INVLAND=H407JJ
END
```

MERGEDEF (Merge Definition)

The MERGEDEF contains processing information for the DMGMERGE program, including:

- The name of a default PRINTDEF
- MSGCTL lines (which allow Metacode users to specify several DJDE records)
- Default VLAM chain name
- Dash (bar) code information

MERGEDEF Parameters

Parameter	Value
NAME=	The name of this MERGEDEF. This is the name that the programs recognize. The delivered values are: ▪ **3820** (AFP printers) ▪ **9700** (Metacode printers)
DASHCODE-OFF=	A 1-character value that turns off the dashcode.
DASHCODE-ON=	A 1-character value that turns on the dashcode.
DEFAULT-CHAIN=	The 4-character default VLAM chain name.
LANDPGSZ=	Specifies in pels, the available page space that Documerge can use to concatenate forms in landscape orientation. For example, a value of 2100 would equal 7 inches at 300 dpi. There is no **LANDPGSZ=** for line printers.

MSGCTLx=	This parameter is primarily for Metacode printers. However, it can also be used with AFP and line printers, although the MSGCTL text will printed along with other output text.
	Specifies up to 10 lines of printer control commands or text to be placed at the top of any page generated by DMGMERGE, including the default banner page and error message pages.
	The **x** in the parameter name represents the sequence number of the Message Control record. The data is entered after the equal sign (=) and can be up to 100 characters long.
	If single quotes are included within this field, double the quotes. Quotes on the end of the strings indicate that blanks are included. For example:
	Single quotes -
	`' $$XEROX OTEXT=('PULL BLUE PAPER' WAIT),;'`
	Double quotes -
	`' $$XEROX OTEXT=(''PULL BLUE PAPER'' WAIT),;'`
	A possible use for this parameter is a DJDE packet.

MSGLPP=	(Required) the number of lines per page for error messages. The minimum value is 25.
OUTCR=	Indicates whether DMGMERGE places comments in the output. Valid values are: **YES** DMGMERGE places comments in the output. **NO** DMGMERGE does not place comments in the output. NO is the default.

Parameter	Value
SWAPJDE=	Indicates whether DMGMERGE uses different JDEs from each individual form. This parameter is valid only for Metacode. JDE= is a DJDE command used to help set up the Xerox printing environment. Multiple JDEs can be used. Valid values are: **YES** DMGMERGE swaps JDEs. **NO** DMGMERGE does not swap JDEs. NO is the default.
PORTPGSZ=	Specifies in pels, the available page space that Documerge can use to concatenate forms in portrait orientation. For example, a value of 3000 would equal 10 inches at 300 dpi. Or for line printers, a value of 48 would equal 8 inches at 6 lines per inch.
PRINTDEF=	The name of the default PRINTDEF that DMGMERGE uses if no PRINTDEF is specified elsewhere. **IMPORTANT:** If you do not code the PRINTDEF and DMGMERGE calls this MERGEDEF, you must code the PRINTDEF in one of the following: ■ the MERGE command in the DMGMERGE program (PRINTDEF= parameter) ■ the FILEDEF command in the DMGMERGE program (PRINTDEF= parameter) ■ the DMG.PDEF.groupname Reserved Tag in the VDR You must specify the PRINTDEF where you specify the MERGEDEF. For example, if you code a FGRPDEF for the MERGE command, then you must also code the MERGEDEF in the MERGE command. Otherwise, DMGMERGE ignores the FGRPDEF coded in the MERGE command.

MERGEDEF Usage Guidelines

Here are some guidelines for using the MERGEDEF:

- The MERGEDEF is required for DMGMERGE. You cannot specify a default MERGEDEF.
- The MERGEDEF, PRINTDEF, or FGRPDEF specified for a Group — or a single Merge Set within a Group — must contain specifications for the device that will actually print the associated output Document Package(s).
- When you code a MERGEDEF, you have the option of including a default PRINTDEF and a default FGRPDEF (via the FGRPDEF= parameter coded in the PRINTDEF), which will apply for all of the Document Packages processed for the specified Group.

  For the whole Group — or for a single Merge Set for the Group — you can also individually specify the MERGEDEF, PRINTDEF and FGRPDEF.

For more information about MERGEDEF coding options, see the following section.

MERGEDEF Coding Options

You can specify the FGRPDEF, PRINTDEF, and MERGEDEF to process a Group in *one* of the following places:

- The **MERGE** command — Specify the MERGEDEF in the MERGE command when the output for all of a Group's Merge Sets will be printed on the same type of printer (this requirement holds, even if you specify independent routing for the output Document Packages).

  For MERGE command MERGEDEF coding details, see "Specifying a MERGEDEF in the MERGE Command" on page 92.
- The **FILEDEF** command — Specify the MERGEDEF in the FILEDEF command when the output for some of a Group's Merge Sets will be printed on different printers.

For example, you can use FILEDEFs to specify that the Document Package for the second Merge Set in the Insured Group will be printed on an AFP printer; and the Document Package for the third Merge Set will be printed on a Metacode printer.

You can use the FILEDEF when the MERGEDEF is not already specified in the MERGE command. For FILEDEF command MERGEDEF coding details, see "Specifying a MERGEDEF in the FILEDEF Command" on page 93.

NOTE

When compared with the DMG.MDEF. *groupname* Reserved Tags, the FILEDEF command offers you the most straightforward way of varying the type of printer by specifying a different MERGEDEF, FGRPDEF, or PRINTDEF for a Merge Set.

- The **DMG.MDEF.***groupname* Reserved Tag — Specify the MERGEDEF in DMG.MDEF.*groupname* when you want to dynamically select the MERGEDEF, PRINTDEF, or FGRPDEF as a Merge Set is being created, so that its output can be printed on a different printer.

  You must use this option if the MERGEDEF is not specified in either the MERGE or FILEDEF command. Otherwise, Documerge issues the following error message:

```
DMGMRG351E Documerge reserved tag not found, DMG.MDEF.groupname
```

  For Reserved Tag MERGEDEF coding details, see "Specifying a MERGEDEF in the DMG.MDEF.groupname Reserved Tag" on page 93.

Specifying a MERGEDEF in the MERGE Command

If	The output for all of a Group's Merge Sets will be printed on the same printer		
Then	Code the MERGEDEF to use in the MERGE command for the Group		
Documerge	Ignores any individual MERGEDEF, FGRPDEF, or PRINTDEF coded in the FILEDEF command for the Group Ignores any DMG.FDEF .*groupname*, DMG.PDEF .*groupname*, or DMG.MDEF.*groupname* Reserved Tag in the VRF		
	Rules for Coding PEDEFs in the MERGE Command		
	If the	**In the MERGE command**	**Documerge uses the**
	MERGEDEF to use specifies the PRINTDEF to use which specifies the FGRPDEF to use	Code only the MERGEDEF to use	Default FGRPDEF specified in the default PRINTDEF which is specified in the MERGEDEF
	FGRPDEF to use is specified in the PRINTDEF to use, which isn't specified in the MERGEDEF to use	Code both the MERGEDEF to use, and the PRINTDEF to use	PRINTDEF specified in the MERGE command, and the default FGRPDEF specified in the PRINTDEF
	FGRPDEF to use and the PRINTDEF to use aren't specified in the MERGEDEF to use	Code the MERGEDEF to use in a MERGEDEF parameter, the PRINTDEF to use in a PRINTDEF parameter, and the FGRPDEF to use in a FGRPDEF parameter	PRINTDEF and FGRPDEF specified in the MERGE command

Specifying a MERGEDEF in the FILEDEF Command

If	The output for some of a Group's Merge Sets will be printed on different printers ▪ **-and-** the MERGEDEF is not specified in the MERGE command
Then	Code a FILEDEF command with a MERGEDEF parameter for all clean (non-error) output files used by the Group
Documerge	Ignores any individual FGRPDEF or PRINTDEF coded in the MERGE command for the Group Ignores any DMG.FDEF .*groupname*, DMG.PDEF .*groupname*, or DMG.MDEF.*groupname* Reserved Tag in the VRF

Rules for Coding PEDEFs in the FILEDEF Command

If the	In the FILEDEF command	Documerge uses the
MERGEDEF to use specifies the PRINTDEF to use which specifies the FGRPDEF to use	Code only the MERGEDEF to use	Default FGRPDEF specified in the default PRINTDEF which is specified in the MERGEDEF
FGRPDEF to use is specified in the PRINTDEF to use, which isn't specified in the MERGEDEF to use	Code both the MERGEDEF to use, and the PRINTDEF to use	PRINTDEF specified in the FILEDEF command, and the default FGRPDEF specified in the PRINTDEF
FGRPDEF to use and the PRINTDEF to use aren't specified in the MERGEDEF to use	Code the MERGEDEF to use, the PRINTDEF to use, and the FGRPDEF to use	PRINTDEF and FGRPDEF specified in the FILEDEF command

*Specifying a MERGEDEF in the DMG.MDEF.***groupname** *Reserved Tag*

IMPORTANT!

If you vary the printer type using the DMG.MDEF. *groupname* Reserved Tags, you must ensure that the definitions do not change in the middle of the file. When such a change occurs, Documerge issues the following error message:

```
DMGMR090W Using different MERGEDEF and/or PRINTDEF names in file
xxxxxxxx
```

If you want a specific MERGEDEF, FGRPDEF, and PRINTDEF to apply for a single Merge Set, then you must code all three DMG.MDEF. *groupname*, DMG.FDEF. *groupname*, and DMG.PDEF. *groupname* Reserved Tags for that same Merge Set.

If	You want to dynamically select the MERGEDEF, PRINTDEF, or FGRPDEF as a Merge Set is being created, so that its output can be printed on a different printer -and- the MERGEDEF is not specified in either the MERGE or FILEDEF command
Then	Code a DMG.MDEF.*groupname* Reserved Tag for the Group
Documerge	Ignores any FGRPDEF or PRINTDEF coded in the MERGE or FILEDEF command for the Group

Rules for Coding the DMG.MDEF, .FDEF, or .PDEF.*groupname* Reserved Tags for the Same Merge Set

If the	In the	Documerge uses the
MERGEDEF to use specifies the PRINTDEF to use which specifies the FGRPDEF to use	DMG.MDEF.*groupname* Reserved Tag, code only the MERGEDEF to use	Default FGRPDEF specified in the default PRINTDEF which is specified in the MERGEDEF
FGRPDEF to use is specified in the PRINTDEF to use, which isn't specified in the MERGEDEF to use	DMG.MDEF.*groupname* Reserved Tag, code the MERGEDEF to use DMG.PDEF.*groupname* Reserved Tag, code PRINTDEF to use	PRINTDEF specified in the DMG.PDEF.*groupname* Reserved Tag, and the default FGRPDEF specified in that PRINTDEF
FGRPDEF to use and the PRINTDEF to use aren't specified in the MERGEDEF to use	DMG.MDEF.*groupname* Reserved Tag, code the MERGEDEF to use DMG.PDEF.*groupname* Reserved Tag, code PRINTDEF to use DMG.FDEF.*groupname* Reserved Tag, code FGRPDEF to use	PRINTDEF specified in the DMG.PDEF.*groupname* Reserved Tag, and the FGRPDEF specified in the DMG.FDEF.*groupname* Reserved Tag

Default PRINTDEF

The MERGEDEF can determine the default PRINTDEF for the MERGE or FILEDEF command. When you create or modify the MERGEDEF with the DPLDUTL utility, you can code the PRINTDEF= parameter to set the default PRINTDEF. Then that PRINTDEF is invoked automatically when you specify the MERGEDEF= parameter in the MERGE or FILEDEF command.

For details about default PRINTDEF coding options and the situations that dictate the use of a particular option, see "MERGEDEF Coding Options" on page 91.

IMPORTANT!

If you do not code the PRINTDEF= parameter in the MERGEDEF, you must specify the PRINTDEF in one of the following:

- the MERGE command in the DMGMERGE program (PRINTDEF= parameter)
- the FILEDEF command in the DMGMERGE program (PRINTDEF= parameter)
- the DMG.PDEF.groupname Reserved Tag in the VDR

You must specify the PRINTDEF where you specify the MERGEDEF.

For example, if you code a PRINTDEF for the MERGE command, then you must also code the MERGEDEF in the MERGE command. Otherwise, DMGMERGE ignores the PRINTDEF coded in the MERGE command.

Default FGRPDEF

The default PRINTDEF can contain a default FGRPDEF for the MERGE and FILEDEF commands. When you create or modify the MERGEDEF with the DPLDUTL utility, you can specify the default FGRPDEF by coding the PRINTDEF= parameter with a PRINTDEF that contains the default FGRPDEF.

The FGRPDEF is invoked automatically with the PRINTDEF when you code the MERGEDEF= parameter in the MERGE or FILEDEF command.

For details about default FGRPDEF coding options and the situations that dictate the use of a particular option, see "MERGEDEF Coding Options" on page 91.

IMPORTANT!

If the PRINTDEF does not contain a default FGRPDEF, you must code the FGRPDEF in one of the following:

- the MERGE command in the DMGMERGE program (FGRPDEF= parameter)
- the FILEDEF command in the DMGMERGE program (FGRPDEF= parameter)
- the DMG.FDEF.groupname Reserved Tag in the VDR.

You must specify the FGRPDEF where you specify the MERGEDEF. For example, if you code a FGRPDEF for the MERGE command, then you must also code the MERGEDEF in the MERGE command. Otherwise, DMGMERGE ignores the FGRPDEF coded in the MERGE command.

Sample MERGEDEF

The following figure shows an example of a MERGEDEF:

Sample MERGEDEF Source

```
MERGEDEF NAME=GRTAM -
    OUTCR=YES -
    DASHCODE-ON=E'_' -
    DASHCODE-OFF=E' — -
-   MSGCTL1='$$XEROX JDL=DPLJDL,JDE=ONLINE,FORMAT=FMT9,;' -
    MSGCTL1='$$XEROX JDL=DPLJDL,JDE=ONLINE,FONTS=P07TCI;' -
-   MSGCTL1='$$XEROX JDL=QAJDL,JDE=ONLINE;' -
    MSGCTL2='$$XEROX DUPLEX=YES,END;' -
    MSGLPP=60 -
    SWAPJDE=YES -
    PRINTDEF=GRTAM
```

PRINTDEF (Printer Definition)

The PRINTDEF (Printer Definition) describes a printer by the hardware type, the software or microcode it uses, and other characteristics. It also provides specifications for the print jobs, such as the lowest addressable pel values, the size of the paper in use, and prefix and suffix file names.

The PRINTDEF is required for the following commands in the DMGMERGE program, regardless of printer type:

- MERGE
- FILEDEF

Also, the PRINTDEF can be used by

- the Documerge Normalization program DMGMETP
- the COMMONFONTS command in the DMGMERGE program

You can specify a default PRINTDEF for each of these commands and for DMGMETP. In turn, you can use this default PRINTDEF to specify a default FGRPDEF.

For details about default FGRPDEF coding options and the situations that dictate the use of a particular option, see "MERGEDEF Coding Options" on page 91.

A PRINTDEF is provided at installation. You can modify it or create others to suit your needs.

PRINTDEF in the MERGE or FILEDEF Command

You can specify the PRINTDEF for the MERGE or FILEDEF command in two ways:

- When you create the MERGEDEF with the DPLDUTL utility, code the name of the default PRINTDEF in the PRINTDEF= parameter. The default PRINTDEF is invoked automatically when you code the MERGEDEF= parameter in the MERGE or FILEDEF command.

  For details about default PRINTDEF coding options and the situations that dictate the use of a particular option, see "MERGEDEF Coding Options" on page 91.

- Code the name of the PRINTDEF in one of the following:
  - the MERGE command in the DMGMERGE program (the PRINTDEF= parameter)
  - the FILEDEF command in the DMGMERGE program (the PRINTDEF= parameter)
  - the DMG.PDEF.groupname Reserved Tag in the VDR.

IMPORTANT!

If the MERGEDEF does not contain a default PRINTDEF, you must code the PRINTDEF in one of the following:

- the MERGE command in the DMGMERGE program (PRINTDEF= parameter)
- the FILEDEF command in the DMGMERGE program (PRINTDEF= parameter)
- the DMG.PDEF.groupname Reserved Tag in the VDR

You must specify the PRINTDEF where you specify the MERGEDEF. For example, if you code a PRINTDEF for the MERGE command, then you must also code the MERGEDEF in the MERGE command. Otherwise, DMGMERGE ignores the PRINTDEF coded in the MERGE command.

Default PRINTDEF

The MERGEDEF determines the default PRINTDEF for the MERGE and FILEDEF commands. The PRINTDEF to use can contain a default FGRPDEF. When you create the MERGEDEF with the DPLDUTL utility, you can specify the default FGRPDEF by coding the PRINTDEF= parameter with a PRINTDEF that contains the default FGRPDEF.

The FGRPDEF is invoked automatically with the PRINTDEF when you code the MERGEDEF= parameter in the MERGE or FILEDEF command.

IMPORTANT!

If the MERGEDEF does not contain a default PRINTDEF that has a default FGRPDEF, you must code the FGRPDEF in one of the following:

- the MERGE command in the DMGMERGE program (FGRPDEF= parameter)
- the FILEDEF command in the DMGMERGE program (FGRPDEF= parameter)
- the DMG.FDEF.groupname Reserved Tag in the VDR

You must specify the FGRPDEF where you specify the MERGEDEF. For example, if you code a FGRPDEF for the MERGE command, then you must also code the MERGEDEF in the MERGE command. Otherwise, DMGMERGE ignores the FGRPDEF coded in the MERGE command.

PRINTDEF with the DMGMETP Program and theCOMMONFONTS Command

You can specify the PRINTDEF for the DMGMETP program in two ways:

- In the ENVDEF= parameter, specify an ENVDEF that contains a default PRINTDEF. (You code the default PRINTDEF when you create the ENVDEF with the DPLDUTL utility.)
- Code the PRINTDEF= parameter.

IMPORTANT!

If the ENVDEF does not contain a default PRINTDEF, you must code the PRINTDEF= parameter in DMGMETP.

The COMMONFONTS command uses the PRINTDEF indirectly, through its ENVDEF= parameter. When you create the ENVDEF with the DPLDUTL utility, code a default PRINTDEF in the PRINTDEF= parameter.

The ENVDEF determines the default PRINTDEF for the DMGMETP program and the COMMONFONTS command. This PRINTDEF can contain a default FGRPDEF.

When you create or modify the ENVDEF with the DPLDUTL utility, you can specify the default FGRPDEF by coding the PRINTDEF= parameter with a PRINTDEF that contains the default FGRPDEF.

The FGRPDEF is invoked automatically with the PRINTDEF when you code the ENVDEF= parameter in DMGMETP or COMMONFONTS.

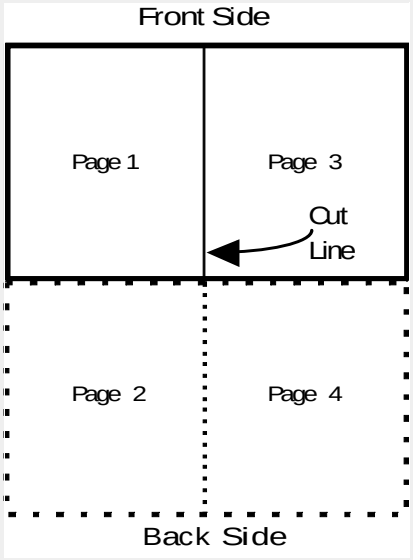
IMPORTANT!

If the ENVDEF does not contain a default PRINTDEF that has a default FGRPDEF, you must code the FGRPDEF= parameter in DMGMETP.

If the ENVDEF does not contain a default PRINTDEF that has a default FGRPDEF, and you code the COMMONFONTS parameter ENVDEF=, you must code the COMMONFONTS parameter FGRPDEF= also.

Using the PRINTDEF to Specify Sheet Feeds for AFP Wide Duplex Printers

Documerge 3.1 and later releases let you use the PRINTDEF to ensure correct sheet feeds for AFP Wide Duplex printers.



Incorrect sheet feeds resulting from too many EDTs (End Document Groups) and BDTs (Begin Document Groups) can cause unwanted blank pages after cutting the printed sheets into pages.

To avoid the occurrence of unwanted blank pages, DMGMERGE will generate IMMs for sheet feeds, and only one BDT and one EDT for each print dataset.

To Implement a PRINTDEF for an AFP Wide Duplex Printer

1. With the DPLDUTL program, define a new PRINTDEF (or change an existing PRINTDEF) specifying PDEV=AFPW for the new device type *AFPW* (AFP Wide).

   For details, see "PDEV=" on page 102.

2. Before processing with the DMGMERGE program, point to the PRINTDEF. You can do this in the following ways:

   - Define a new MERGEDEF (or change an existing MERGEDEF) to point to the new PRINTDEF (for details, see "Default PRINTDEF" on page 95).
   - Specify the PRINTDEF for DMGMERGE to use, in any of the following:
     - MERGE command (for details, see"PRINTDEF=" on page 438).
     - FILEDEF command (for details, see"PRINTDEF=" on page 414).
     - DMG.PDEF.*groupname* Reserved tag (for details, see"DMG.ERDD.*Groupname*" on page 319).

NOTE

With a device type of AFPW, Documerge forces DYNAMICCG=YES as the parameter for the GLOBAL command.

3. In the FORMDEF COPYGROUP for the application, specify that the printer select the next odd quadrant (instead of next sheet) when it detects an IMM structured field. (IBM refers to the page faces on a sheet as *quadrants*.)

   For details, see the *IBM InfoPrint Hi-Lite Color Reference*.

PRINTDEF Parameters

Parameter	Value
NAME=	The name of this PRINTDEF. This is the name that the ENVDEF references. The delivered values are: ▪ **3820** (AFP printers) ▪ **9700** (Metacode printers) When the PEDEF object-creation utility DPLDUTL, is executed, the object name is prefixed with **DP**.
AUXFEED=	(AFP printers only.) The COPYGROUP that should be used to feed paper to the printer from a secondary input tray. The name can be up to 8 characters long.
COLOR=	Indicates whether the output is printed in color. Valid values are: **NO** Suppresses color printing Code COLOR=NO to route colored SCRIPT documents to a non-color printer without changing the DCF source. COLOR=NO also suppresses the INKS= parameter for a colored IMG. **YES** Enables color printing
	Refer to *DCF/PLUS Support for Xerox Highlight Color Printers* for more information.
COPYMODx=	(AFP printers only.) The COPYGROUP name. For the lower-case **x**, substitute the number of the parameter. You can have up to 5 COPYMODx parameters in a PRINTDEF. The COPYGROUP name can be up to 8 characters long.
COPYMODx-FLAG=	This parameter identifies the string to be used to invoke a printer copy modification feature. This parameter can be up to 50 characters long, and applies only to Metacode printers. The **x** indicates the number of the flag. You can have up to 5 COPYMODx-FLAG parameters in a PRINTDEF.
COPYMODx-OFF=	(Metacode printers only.) The offset for the corresponding COPYMODx-FLAG.
CPI=	(Line printers only.) The number of characters per inch.
CPL=	(Line printers only.) The number of characters per line.
DEFAULTCG=	(AFP printers only.) The default COPYGROUP to be selected in the current form definition. This parameter can be up to 8 characters long. The default is the first COPYGROUP in a form definition.
DJDEOFF=	(Metacode printers only). The DJDE offset, which is the number of bytes in the record preceding the DJDE. This value must match the value in the JDL which is named in the ENVDEF. The delivered value is **0**.
DJDESKIP=	(Metacode printers only). The DJDE skip value, the number of bytes preceding the first command on a DJDE record. This value must match the value in the JDL that is named in the ENVDEF. The number of bytes can range from 0 to 255. The delivered value is **8**.

Parameter	Value
FGRPDEF=	(Metacode printers only.) The name of a default FGRPDEF. **IMPORTANT:** If you do not code this parameter and DMGMERGE calls this PRINTDEF directly or indirectly, you must code one of the following: ■ the MERGE command in DMGMERGE (FGRPDEF= parameter) ■ the FILEDEF command in DMGMERGE (FGRPDEF= parameter) ■ the DMG.FDEF.groupname Reserved Tag in the VDR If you do not code this parameter and DMGMETP calls this PRINTDEF directly or indirectly, you must code the FGRPDEF= parameter in DMGMETP. If you do not code this parameter and the ENVDEF in the COMMONFONTS parameter calls the PRINTDEF, you must code the COMMONFONTS FGRPDEF= parameter also.
GHO=	(Metacode printers only.) The Graphics Handling Option, which allows raster images to be downloaded to the printer by the GRAPHIC DJDE command. Valid values are: **NO** The printer does not have GHO. NO is the default value. **YES** The printer has GHO.
HPEL=	The horizontal pel (picture element) density of the printer. This parameter does not apply to line printers. The density can range from 1 to 3276 pels. The usual value of this parameter is 300 dots. The delivered value is **300**. The default value is **0**.
IDEN=	(Metacode printers only). The DJDE identifier. This value must match the print JDL, DPLJDL. The identifier is a string up to 255 characters. The delivered value is **$$XEROX**. There is no default value.
INVERTDJDE=	(Optional) generates an INVERT DJDE record that directs a Xerox 4635 printer to perform tumble printing. **CAUTION:** use this parameter only if the printer for your application is a Xerox 4635 or similar printer that performs DJDE-activated tumble printing. To implement this parameter, you must 1 Code a PRINTDEF with **INVERTDJDE=Y.** 2 Specify the tumble print option in the Rulebase Structure table for the Document Package. 3 Specify the PRINTDEF coded in step 1 in the associated MERGE or FILEDEF commands, or DMG.PDEF.Groupname Reserved tag. Valid values are: **N** or **NO** (Default) do not write an INVERT DJDE record. Documerge will perform any specified tumbling of the print stream data. **Y** or **YES** Write an INVERT DJDE record.
LOGICAL-EOF=	(AFP and line printers only.) The logical end-of-file defined on your printer. If a logical end-of-file is not required for your printer, you may choose to insert text that you would like to appear at the end of a file. This parameter can be up to 255 characters long.
LOWHPEL=	(Metacode printers only.) The lowest addressable pel, or page origin, as measured from the left edge of the sheet. Specify **25** dots for this parameter. This is the delivered default value.

Parameter	Value
LOWVPEL=	(Metacode printers only.) The lowest addressable pel, or page origin, as measured from the bottom of the sheet. The number specified should be in dots. This value is an integer ranging from 1 to 99999. The delivered value is **99** dots.
LPI=	(Line printers only.) The number of Lines Per Inch.
LPP=	(Line printers only.) The number of Lines Per Page.
MAXCOPY=	Sets the upper limit for the number of copies Documerge will produce. This is a global control that sets the maximum value allowed in the MERGE command COPIES= parameter or the **DMG.GCPY.groupname** Reserved Tag. This value must be an integer ranging from 1 to 99999. A zero value indicates no maximum.
MAXFNUM=	The maximum number of fonts per page allowed by your printer's software. This parameter applies only to Metacode printers. This value must be an integer ranging from 1 to 9999, and is restricted to the maximum font memory available on your printer and your printer's software. The delivered value is: **32** Metacode printers
PAPEROFFSET=	(AFP printers only.) The name of the COPYGROUP that invokes the paper offset in the output tray. This name can be up to 8 characters long.
PDEV=	The physical device associated with the logical device. This parameter indicates the type of data stream to be generated. Valid physical device types are: **9700** Xerox Centralized Printers, including the 8700/9700/8790/9790/4050 series. **AFP** IBM AFP printers or their emulators. **AFPW** IBM AFP Wide Duplex printers (such as the 3900) and their emulators.
PREFIXDD=	The DD name of the prefix file. This name must be entered in parentheses. A prefix file might be necessary when you are coding a command that must precede your entire output file. For example, you might want to code a command to send your file to an auxiliary printer tray. In such a case, you would want the command in a prefix file. The contents of the prefix file are not processed. If you want this file to be attached to the beginning of the output, you must indicate PREFIX=YES in the ENVDEF.
PRINTDEF=	The name of an existing PRINTDEF which is the model for this one. When a parameter is not provided, it defaults to the supplied PRINTDEF. New PRINTDEF parameters override the same ones in the model.
PRINTMODE=	Indicates whether your printer is in DUPLEX or SIMPLEX mode. This parameter is valid for Metacode, AFP, and line printers. The two valid values for this parameter are **DUPLEX** and **SIMPLEX**. The default value is **SIMPLEX**.

Parameter	Value
RBARFLAG=	(Metacode printers only) specifies the character string that controls the generation of the RBAR record that Xerox requires for BTEXT processing. DMGMERGE generates an RBAR record for any page that contains at least one DMG.BTEXT.SEQ Reserved tag. For an explanation of BTEXT, see "Guidelines for Specifying Output Files when Routing-by-Sheets and/or Segmenting Output" on page 418. The string can be any unique character string, but if you do not want the string to print, Oracle suggests that you use a hexadecimal string that starts with **01**, such as **X'01D9C2C1D9'**, which is he xi decimal 01 followed by the EBCDIC string *RBAR*. The value must match the value in the JDL/JDE that was started for the printer. If you code the RBARFLAG parameter, you must code the **RBAROFF** parameter also.
RBAROFF=	(Metacode printers only). Specifies the starting position (offset) of the RBARFLAG value for BTEXT RBAR record processing. For an explanation of BTEXT, see "Guidelines for Specifying Output Files when Routing-by-Sheets and/or Segmenting Output" on page 418. This parameter identifies the number of bytes (typically 0) in the record preceding the RBARFLAG parameter value. The value must match the value in the JDL/JDE that was started for the printer.
RELEASE=	The release of the printer's software or microcode (this does not apply to AFP or line printers).
ROFFFLAG=	(Metacode printers only.) Specifies the data string for the Metacode ROFF parameter. When a Xerox printer detects the string in input data for a page, that page and all the pages printed before the next ROFF string are stacked in the output bin by the amount specified by the ROFFOFF parameter (see the next parameter). For example: `ROFFFLAG=X'1212121212121212'-` specifies `1212121212121212` as the string that triggers offset of the stack of printed pages.
ROFFOFF=	(Metacode printers only.) Specifies the amount of output stack offset triggered by ROFF parameter. The delivered value is zero, but must match the value defined in the JDL for the printer. For example: `ROFFOFF=0-` specifies `0` as the value by which to off set the stack of pages.
RSTKFLAG=	(Metacode printers only). The RSTACK identifier. This parameter identifies the end of the report or print job to the printer and returns the printer to the default job. This value must match the value in the JDL invoked in ENVDEF. The delivered value is **REPORT END**. Use the ENVDEF RSTACK= parameter to include or exclude the RSTACK.
RSTKOFF=	(Metacode printers only). The RSTACK offset. This parameter identifies the number of bytes in the record preceding the RSTACK. Use the ENVDEF RSTACK= parameter to include or exclude the RSTACK. The value must match the value in the JDL created by the ENVDEF. It can range from 0 to 255 bytes. The delivered value is **0**.

Parameter	Value
SHEETLEN=	The length of the sheet in use on the printer, specified in dots. This parameter does not apply to line printers. The number is an integer ranging from 1 to 99999, and the delivered default is 3300.
SHEETWID=	The width of the sheet in use on the printer, specified in dots. This parameter does not apply to line printers. The number is an integer ranging from 1 to 99999, and the delivered default is 2550.
SUFFIXDD=	The DD name of the suffix file. This name must be entered in parentheses. You might want to create a suffix file to contain a command that must follow your entire output file. For example, you may want to code a command that will eject the last page of your output from the printer. In such a case, you would want the command in a suffix file. The contents of the suffix file are not processed. If you want this file to be attached to the end of the output, you must indicate SUFFIX=YES in the ENVDEF.
VERSION=	The version of the printer's software or microcode (this does not apply to AFP or line printers). If more than 16 fonts are used in a document, the VERSION must be two or higher. The delivered value is 2.
VPEL=	The vertical pel density of the printer. This parameter does not apply to line printers. The density can range from 1 to 3276 pels. The usual value for this parameter is 300 dots, and this is the delivered value.

Sample PRINTDEF

The following figure shows a PRINTDEF coding sample:

Sample PRINTDEF Source

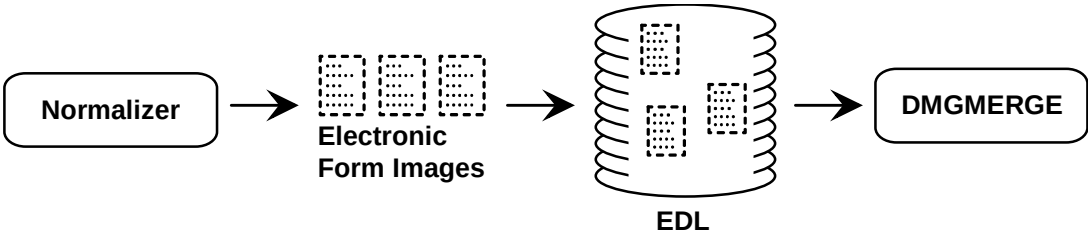
```
PRINTDEF NAME=SAMPLE -
         PDEV=9700 MAXFNUM=32 FGRPDEF=9700 -  <---Printer Description
         VERSION=2 -                          <---Available Software
         SHEETWID=2550 SHEETLEN=3300 -        <---Paper Description
         LOWHPEL=25 LOWVPEL=99 -              <---Hardware Limits
         IDEN=$$XEROX DJDEOFF=0 DJDESKIP=8 -  <---DJDE Information
         RSTKFLAG=E'REPORT END' RSTKOFF=0     <---RSTACK Information
```

The Electronic Document Library (EDL)

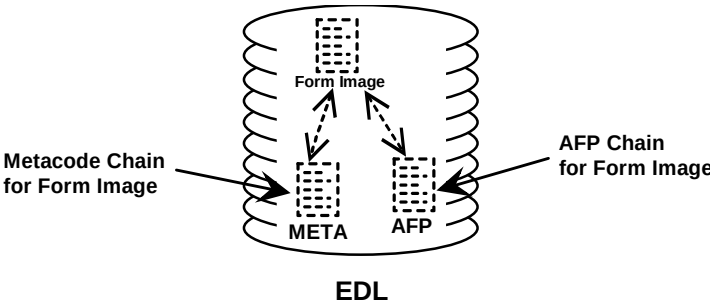
The Electronic Document Library (EDL) is a VLAM library. You use two utility programs for creation and maintenance of the EDL.

- IBM's VSAM utility (IDCAMS)
- Oracle VLAM Maintenance Utility (VLMMAINT)

You use the EDL to store normalized forms. You can also use it to store composition source files. Forms and composition source files are stored as members within the EDL.



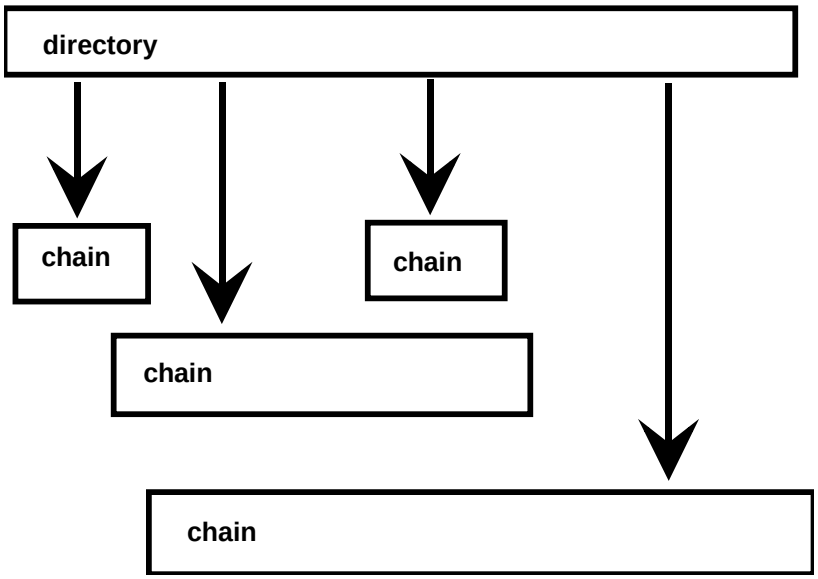
These EDL members are identified by a name, revision level and a chain. The data for each member is stored as a chain, with a maximum of 32 chains per member. One EDL member can have multiple chains. These chains can contain composition source; they can also contain forms for various printer data streams.



A member has a directory part and one or more data parts. The directory part is simply the member directory. The data parts are called chains. A member is made up of directory information and whatever data you choose.

Parts of a member: Every member has a directory. A member can have up to 32 chains.

A VLAM Member



The EDL is opened twice during a Documerge job:

- First by DMGRFMT, to verify that the requested EDL member exists and to retrieve data relevant to that member.
- Second by DMGMERGE, to retrieve the form represented by the member.

DMGMERGE selects the appropriate EDL member chain based upon the chain information specified in the MERGEDEF, the DMGMERGE control card CHAIN=, or the DMG.FLST.groupname Reserved Tag.

VLAM is used by several Oracle products and is documented in *Using VLAM*. This chapter highlights VLAM information that directly relates to Documerge applications.

There are only two VLMMAINT commands that are described fully in this chapter, LOAD and ALTER. These are the only two commands that have Documerge-specific parameters. The other VLMMAINT commands are described briefly.

The following VLMMAINT commands can be used with Documerge 3.0 and later releases. Refer to *Using VLAM* for more information.

Parameter	Value
FORMAT	Formats the VSAM RRDS as a VLAM Library. Used for: ■ Library maintenance, assigning and altering Library-level passwords ■ Increasing the number of Library blocks when a secondary space allocation is defined by IDCAMS.
GLOBAL	Defines parameter defaults. Can also be used with other VLMMAINT control card commands. Defaults can be reset by specifying a GLOBAL command with no parameters. GLOBAL command parameters not available to other VLMMAINT commands are ignored.

Parameter	Value
LOAD	Reads and records normalized forms or composition source from a file. It then writes these to a specified chain of a specified member. Can be used to: ▪ Create new members or chains ▪ Overwrite existing members or chains with the same name and revision level.
DIRECTORY	Reports information about one or more Library members. There are five different report formats, with one designed specifically for Documerge (TYPE=DOCUMERGE).
ALTER	Modifies EDL member information. Cannot be used to modify the member name or chain. RENAME is used for modification of this information.
BACKUP	Writes specified members or entire EDL libraries to a physical sequential file. Processes those members whose DISPOSITION matches the DISPOSITION setting for the BACKUP command.
COPY	Copies EDL members and/or specific chains and their directory information.
	IMPORTANT: If you want to transfer the DTN, Description, etc., of a member being copied to an existing member, you must code the **DIRECTORY=YES** parameter in the COPY command. You can copy members within a Library, or you can copy members from one Library to another.
DUMP	Copies a member chain from the EDL Library to a sequential file. A maximum of one chain can be dumped per command execution. DUMP can be used to copy the EDL chain to a sequential file for printing the chain.
RESTORE	Copies members from a sequential file that was created with the VLMMAINT BACKUP command to an EDL Library.
ERASE	Deletes members or chains of members from the EDL Library.
RENAME	Changes the name of EDL members and chains.

EDL Library Definition

One VLAM Library cannot contain members for both the Rulebase Tables and Electronic Data Library forms even though both libraries use the same VLAM access method. This is because Documerge uses two different paths to VLAM that cannot share the same physical file.

The number of EDLs your company creates and maintains depends on your company's Documerge implementation plans.

The Documerge 3.x Electronic Document Library (EDL) is maintained using Oracle Virtual Library Access Method (VLAM) Version 2.7 or 2.9. It provides a tool to store data using a VSAM RRDS (IBM's Virtual Storage Access Method Relative Record Data Set). An EDL is created as part of the Documerge product installation. Refer to *Installing Documerge* for more information and sample JCL of how this EDL was created and initially loaded with the demonstration forms.

The initial creation and definition of a Documerge 3.x EDL is a multi-step process. The first two steps only have to be done once for each EDL when it is first defined.

To Specify a New Library

1. Using IBM's IDCAMS utility, execute a DEFINE CLUSTER command. This allocates the file and defines the disk storage space to VSAM.
2. Using Oracle VLMMAINT utility, execute a FORMAT command. This establishes the file format and the initial VLAM directory entries within a newly defined VSAM RRDS.
3. Add new EDL members using Oracle VLMMAINT utility. This adds the new forms from your composition system to the newly created EDL.

NOTE

Do not use a mainframe file-management system to archive the EDL, especially if the EDL has been given secondary allocation space.

All creation and manipulation of the EDL must be accomplished with the VLMMAINT utility. IBM's *VSAM Catalog Administration: Access Method Services Reference* manual provides information on the use of the IDCAMS utility. You should refer to *Using VLAM* for more information about FORMAT and other VLMMAINT commands.

Contact your technical support group for any more information on procedures your company may have to allocate DASD space.

Contents of an EDL

An EDL Library is a VSAM RRDS that contains

- A VLAM Master Directory Block (library information and statistics)
- One or more VLAM-defined chains

The name given to the VSAM file when it is defined is also the EDL Library name.

There are some restrictions on the VSAM operands which can be used when defining a library:

- The record size of the VLAM Library must be exactly seven bytes less than its control interval size.
- The minimum control interval size for a VLAM Library is 1024 bytes with acceptable control interval sizes of integral multiples of 2048 bytes.
- SHAREOPTIONS(4) should be specified.

- REUSE should not be specified.

Sample JCL for Defining a VSAM RRDS

```
//(JOB card)
//* //IDCAMS     EXEC PGM=IDCAMS
//SYSPRINT   DD   SYSOUT=*
//SYSIN      DD   *
 DEFINE CLUSTER( NAME(VLAM2.EDL.LIBRARY) -
                 NUMBERED -
                 VOLUMES(VOLNO) -
                 RECORDSIZE(4089 4089) -
                 CONTROLINTERVALSIZE(4096) -
                 CYL(30 0) -
                 SHAREOPTIONS(4) – -
        DATA (NAME(VLAM2.EDL.LIBRARY.DATA))
/*
//
```

NOTE

VLAM libraries can have secondary allocation. For more information, refer to *Using VLAM*.

For more information on IDCAMS, refer to IBM's *VSAM Catalog Administration: Access Method Services Reference*.

Maintaining Your EDL

Maintain your Documerge 3.x Electronic Document Library (EDL) by using Virtual Library Access Method (VLAM), an Oracle product.

Use caution when upgrading libraries from earlier versions and levels of VLAM to a later version / level.

To Back Up an Old Library and Restrore It into a New Library

1. FIRST use the VLMMAINT BACKUP command from the version / level of VLAM *with which the library was created*.
2. THEN use the VLMMAINT RESTORE command from the VLAM version / level to which you are upgrading the library.

You should *always* apply this BACKUP / RESTORE sequence.

For further information, as well as for information about migrating from 1.x versions of VLAM, see *Migrating Documerge*.

Because VLAM is used across different Oracle products, there are features and command parameters that may not be directly applicable to Documerge. The following is a brief explanation of the VLMMAINT commands. These are followed by a brief description of the parameters in the ALTER and LOAD command that apply specifically to Documerge.

The VLMMAINT LOAD Command

The following is a brief explanation of the LOAD command and its parameters that are specific to Documerge. This does not describe every parameter available for this command.

For a more detailed explanation of this command and JCL examples, refer to *Using VLAM*.

The VLMMAINT Global command is available to LOAD.

VLMMAINT LOAD reads records from a sequential file. It then writes these to a specified chain of a specified member. LOAD can be used to:

- Create new members or chains.
- Overwrite existing members or chains with the same name and revision level.

VLMMAINT LOAD Command Control Cards

```
-
   LOAD Member='MAXIMUM 32 CHARACTERS'(REV#) -
        Chain=XXXX -
        Description='MAXIMUM 36 CHARACTERS' -
        DevType=XXXXXXXX -
        Disposition=XXXXXXX -
        DTN=##### -
        EFFDATE=YYYYMMMDD -
        EOFSTRING=XXXXXXXX -
        InputDD=INPUT1 -
        LibraryDD=VLAMLIB -
        NEWRWPASSWORD=XXXXXXXX -
        PASSWORD=XXXXXXXX -
```

VLMMAINT LOAD Command Control Cards

The following are VLMMAINT command control cards used with the LOAD command in "VLMMAINT LOAD Command Control Cards" on page 111.

Control Card	Value
MEMBER	The EDL member name assigned to the normalized form or composition source. This is a 1-32 character name and revision level. The name cannot contain characters of hexadecimal zero (X'00'), asterisk (*) or question mark (?). Member names containing blanks or single quotation marks must be enclosed in single quotation marks (the revision level remains outside the quotation marks). Revision level values are: **HIGH or 0** If the member does not exist at the time of LOAD, a revision level of 1 is assigned. **NEW** If the member does not exist at the time of LOAD, a new revision level of 1 is assigned. If the member already exists at the time of LOAD, a revision level one higher than that of the existing member is assigned. NEW is the default. **Numeric value** A specific revision level is assigned.

Control Card	Value
CHAIN	A 1-4 alpha or numeric character name. DMGMERGE retrieves matching EDL chain members. One EDL member can have up to 32 chains. Here are rules for Chain names processing: ■ If a chain name was not defined to DMGMERGE in the DMG.FLST.groupname reserved tag, DMGMERGE will retrieve EDL members with chains equal to the DEFAULT-CHAIN name assigned in the MERGEDEF. If a DEFAULT CHAIN name was not assigned in the MERGEDEF, then DMGMERGE retrieves EDL members with chain values that equal the PDEV value as contained in the PRINTDEF. ■ This is mandatory with no default.
Description	A 1-to-36 alphanumeric character description of the EDL member. If the description contains blanks, it must be enclosed in single quotes. This is optional with no default.
DEVTYPE	The device type applicable to the member. For valid values please refer to *Using VLAM*.
DISPosition	Determines BACKUP processing. If the member matches the MEMBER and DISPOSITION criteria of the BACKUP command, the backup operations are performed on the member. Valid values are: ■ **ALL** — write all selected members in the library to the backup DD, then delete any members with the ARCHIVE disposition from the library. ■ **ARCHIVE** — only write library members with the ARCHIVE disposition to the backup DD, then delete those members from the library. ■ **KEEP** — only write library members with the KEEP disposition to the backup DD. The members are not deleted from the library. The default is KEEP.
DTN	A number from 0 to 99999. The DTN is used by DMGRFMT to determine where the EDL form member is to be placed within a Merge Set. The default is 0.
EFFdate	Assigns a date to an EDL member. DMGRFMT uses this date to determine which EDL member to use. Valid coding rules are: Delimiters of slash (/), hyphen (-), comma (,) or blank —) can be used between the year, month and day. Example: `##/##/## or ##-##-## or ##,##,## or ## ## ##` **Year**, numeric value from 00 to 99. You can specify the value as the last two digits of a year or all four digits. If only one digit is specified and delimiters are used, the number will be padded with a leading zero. **TIP:** By default, VLAM stores the value you've entered as a 4-digit year according to the following rules: ■ If the digits (nn) are less than or equal to 50, VLAM stores them as **20nn**. ■ If the digits (nn) are greater than 50, VLAM stores them as **19nn**. You can change the cut off year from the default of 50. For details, see "YEAR2000=nn" on page 196. **Month**, numeric value from 1 to 12 or alpha value equal to the first three letters of the American spelling of the month. If delimiters are used, single digits are padded with a leading zero. **Day**, numeric value from 1 to 31. If you use delimiters, single digits are padded with a leading zero. There is no verification that a day is valid within a given year. The default value is 00/00/00.

Control Card	Value
EOFstring	A 1-8 byte character string used as a logical end of a file marker. Used when one INPUTDD is specified to LOAD which contains multiple data for multiple chains. This is optional.
INPUTDD	Designates the file name of the JCL control statement containing the data to be loaded. Multiple JCL file names follow the equal sign separated by one or more blanks, enclosed by a single set of parentheses. This is mandatory. For example: `INPUTDD=(INPUT1 INPUT2 INPUT3)`
LibraryDD	Designates the EDL's JCL file name. The default is VLAMLIB.
NEWRWpassword	A 1-8 alphanumeric value for read/write password. This is optional for the LOAD command with a default value of blanks.
	NOTE: VLAM permits the assignment of a read-only password, NEWROPASSWORD. VLAM Libraries used with Documerge applications cannot have a read-only password. Documerge needs read/write access to libraries and members.
PASSWORD	A 1-8 alphanumeric value indicating the previously assigned read/write password. Mandatory only when password protection is in effect for an existing member.

Additional information and JCL examples for the VLMMAINT LOAD command are available in *Using VLAM*.

The VLMMAINT ALTER Command

The following is a brief explanation of the ALTER command and its parameters that are specific to Documerge. This does not describe every parameter available for this command.

For a more detailed explanation of this command and JCL examples, please refer to *Using VLAM*.

The VLMMAINT ALTER command modifies EDL member information. ALTER cannot be used to modify the member name or chain. RENAME is used for modification of this information.

VLMMAINT ALTER Command Control Cards

```
-
  ALTER Member='XXXXX(XXXX)' -
        EXCLUDEMEMBER=*(HIGH) -
        SCAN=XXX -
        Description='MAXIMUM 36 CHARACTERS' -
        DevType=XXXXXXXX -
        Disposition=XXXXXXX -
        DTN=##### -
        EFFDATE=YYYYMMMDD -
        LibraryDD=VLAMLIB -
        NEWRWPASSWORD=XXXXXXXX -
        PASSWORD=XXXXXXXX
```

VLMMAINT ALTER Command Control Cards

The following are VLMMAINT DD command control cards used with the ALTER command in "VLMMAINT ALTER Command Control Cards" on page 114.

Control Card	Value
MEMBER	A 1-to-32 character name assigned to the EDL member to be altered. Member names containing blanks or single quotation marks must be enclosed in single quotation marks (the revision level remains outside the quotation marks). Multiple member names can be listed but must be separated by at least one blank and enclosed in a single set of parenthesis. Wildcard characters are permitted in any or all of the member names. Revision level is the 1 to 5 character value assigned to the member to be altered. Wildcard characters are permitted. MEMBER is mandatory, revision is optional with a default of HIGH.
EXCLudemember	The name of an EDL member or members to be excluded from ALTER. Wildcard characters are permitted. Member names containing blanks or single quotation marks must be enclosed in single quotation marks (the revision level remains outside the quotation marks). Multiple member names can be listed but must be separated by at least one blank and enclosed in a single set of parenthesis. Optional with HIGH as the revision level default and no default for member name. For example: `EXCLUDEMEMBER=(MEMBER1(*) MEMBER2(*) MEMBER3(*))`
SCAN	A value of YES indicates that ALTER is not to take place and that a report is generated indicating which EDL members and chains are effected by the proposed ALTER command. A value of NO indicates a SCAN report is not generated and the ALTER is to take place.

VLMMAINT ALTER Command Control Card Parameters

The following parameters are used by the ALTER command.

Parameter	Value
Description	A 1-to-36 alphanumeric character description to replace the current description. If the description contains blanks, it must be enclosed in single quotes. This is optional with no default.
DEVTYPE	The device type applicable to the member. For valid values please refer to *Using VLAM*.
Disposition	This determines BACKUP processing. Valid values are: ■ **ALL** — write all selected members in the library to the backup DD, then delete any members with the ARCHIVE disposition from the library. ■ **ARCHIVE** — write the member to the BACKUP file and delete it from the library. ■ **KEEP** — write the member to the BACKUP file but don't delete it from the library. KEEP is the default.
DTN	A number from 0 to 99999. The DTN is used by DMGRFMT and DMGMERGE to determine where the EDL form member is placed within a merge set. This is mandatory for the ALTER command with a default of 0.
EFFdate	This assigns a date to an EDL member. Documerge uses this date to determine which EDL member to use. Valid coding rules are: ■ Delimiters of slash (/), hyphen (-), comma (,) or blank —) can be used between the year, month and day. `Example: ##/##/## or ##-##-## or ##,##,##` ■ **Year** numeric value from 00 to 99. Value can be specified as last two digits of year or all four digits. If only one digit is specified and delimiters are used, the number is padded with a leading zero. **TIP:** By default, VLAM stores the value you've entered as a 4-digit year according to the following rules: ■ If the digits (nn) are less than or equal to 50, VLAM stores them as **20nn**. ■ If the digits (nn) are greater than 50, VLAM stores them as **19nn**. You can change the cut off year from the default of 50. For details, see "YEAR2000=nn" on page 196. ■ **Month** numeric value from 1 to 12 or alpha value equal to the first three letters of the American spelling of the month. If delimiters are used single digits are padded with a leading zero. ■ **Day** numeric value from 1 to 31. If you use delimiters, single digits are padded with a leading zero. There is no verification that a day is valid within a given year. ■ Default value of 00/00/00. Sometime you might encounter this situation: ■ A form is requested ■ It exists in an EDL ■ It has a date that is later than the effective date passed to the VDR In this case, Documerge generates a message saying that the date is invalid, and Documerge does not include the form in the Document Package.
LibraryDD	Designates the EDL Library name. This is mandatory.

Parameter	Value
NEWRWpassword	A 1-to-8 character alphanumeric value for read/write password. The default values are blanks.
	NOTE: VLAM permits the assignment of a read-only password, NEWROPASSWORD. VLAM Libraries used with Documerge applications cannot have a read-only password. Documerge needs read/write access to libraries.
PASSWORD	A 1-8 alphanumeric value indicating the previously assigned read/write password. Mandatory when password protection is in effect for existing member.

Generic VLMMAINT JCL

```
//VLMMAINT  ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* **             VLAM V. 3.2 MAINTENANCE UTILITY                    **
//* **                                                                **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//VLMMAINT EXEC PGM=VLMMAINT,REGION=2M
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FA,LRECL=133)
//*
//* Control Card Listing
//LISTING  DD SYSOUT=*,DCB=(RECFM=VBM,LRECL=133,BLKSIZE=137)
//*
//* VLMMAINT Message File
//MESSAGE  DD SYSOUT=*,DCB=(RECFM=VBM,LRECL=133,BLKSIZE=137)
//*
//* VLMMAINT Report  File
//REPORT   DD SYSOUT=*,DCB=(RECFM=VBM,LRECL=133,BLKSIZE=137)
//*
//* VLAM 2.9 Library
//VLAMLIB  DD DSN=YOUR.VLAM.V03R02.LIBRARY,DISP=SHR
//*
//* Work file
//ISIWORK  DD DSN=&&ISIWORK,           OVERFLOW WHEN STACKBUFF IS FULL
//            UNIT=sysda,
//            DISP=(NEW,DELETE),
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                 HALF TRACK
//*
//* LOAD uses the following file
//*INPUT    DD DSN=input.file,DISP=SHR
//*
//* BACKUP and RESTORE use the following file
//* The BACKUP file can be tape or disk
//*BACKUP   DD DSN=backup.file,
//*            DISP=(NEW,CATLG),DCB=BLKSIZE=23476,
//*            SPACE=(CYL,(100,50)),UNIT=sysda
//*
//*
//SYSIN   DD *
  vlmmaint control cards
/*
//
```

Additional information and JCL examples for the VLMMAINT ALTER command are available in *Using VLAM*.

Multiple EDLs

Documerge 3.x lets you to use multiple EDLs (also termed "concatenated," or "appended", EDLs) in any given execution of Documerge.

Access multiple EDLs by using the EDLNAMES= parameter. Both the VDR and DMGMERGE use this execution parameter to point to a list file of EDL names. See "EDLNAMES=" on page 375 or "EDLNAMES=" on page 194 for more information.

NOTE

To reduce storage requirements, specify the JCL parameter **STRNO=1** for each EDL in the DMGMERGE JCL DD.

Uses of Multiple EDLs

Use multiple EDLs with Oracle imaging product I.R.I.S., or in any other situation in which you might wish to delete one EDL, or specific members of an EDL, without deleting all EDLs. For example, you could test your Documerge run with a test EDL which you delete before beginning your production run with a production EDL. In this case, you would select forms from the test EDL using one of two methods:

- Concatenate the test EDL ahead of the production EDL in your EDLNAMES file.
- Ensure that each form name duplicated in the two EDLs had a *higher* revision level in the *test EDL* (see below).

Note the use of the slash ("/") with EDL names. The slash indicates to Documerge that the data sets in the specified EDL should not be placed in the Documerge Forms Buffer (see "EDLNAMES=" on page 375).

For example, you might use the slash when designating an EDL to contain only those forms that are unique for a single Merge Set (that is, forms not shared by several Merge Sets).

EDL Selection Sequence

Documerge selects the *highest* revision level of a form, regardless of the EDL. If the same form with the *same revision level* exits in more than one EDL, Documerge will select the form from the first EDL listed in the EDLNAMES file. In other words, the sequence of names in the EDLNAMES list file makes a difference *only* if there are identical form names with identical revision levels in more than one EDL.

CAUTION!

Due to space limitations, the SYSOUT listing for a Documerge run will typically only list the first three multiple (concatenated) EDLs.

Multiple DTNs

For flexibility in defining Document Packages, you can use the DTNS chain to specify multiple DTNs for one EDL member. This chain contains additional DTNs for the form, and it is optional. (You specify the primary DTN in the VLMMAINT SYSIN, with the DTN= parameter.)

Use the DTNS chain when one or more of the Groups that receive a particular form do not receive all the forms in the DTN. If you use the DTNS chain, you can assign different DTNs to a form without loading the same form under different names.

For example:

- DTN 10 consists of forms A, B, and C.
- Group 1 receives all three forms; however, Group 2 receives only form B.
- Using the DTNS chain, you assign DTN 15 to form B in addition to its primary DTN 10.
- In the Structure Rule for Group 1, you specify DTN 10; forms A, B, and C compose the Group 1 Document Package.
- In the Structure Rule for Group 2, you specify DTN 15; only form B composes the Group 2 Document Package.

You can print the DTNS chain for analysis by using the VLMMAINT command DUMP. See "DUMP" on page 107 for more information.

DTNS Chain JCL Example: Assigns DTN Values of 10, 20, 30, 40, and 50

```
//VLMLOAD  EXEC  VLMMAINT
//STEPLIB  DD  DSN=ISI.DOCUMERG.V03R02.LOADLIB,DISP=SHR
//VLAMLIB  DD DSN=your.edl,DISP=SHR
//LISTING  DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133,BLKSIZE=1330)
//MESSAGE  DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133,BLKSIZE=1330)
//REPORT   DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133,BLKSIZE=1330)
//METADATA DD DSN=your.Metacode.file.to.be.loaded,DISP=SHR
//AFPDATA  DD DSN=your.afp.file.to.be.loaded,DISP=SHR
//SYSIN    DD *
LOAD   MEMBER=LM.LIFE.OVERLAY(HIGH)      -
       DESC='INSTALLATION TEST OVERLAY'  -
       DTN=(10 20 30 40 50)              -  multiple DTNs
       CHAIN=(META AFP)             -
       INPUTDD=(METADATA AFPDATA)
/*
```

The Common Font Update (DMGCMFN) Utility

The **Common Font Update (DMGCMFN)** utility changes the font lists of Metacode forms in the EDL.

For each EDL member you select, DMGCMFN

(1) Generates a new font list matching a Common Font List that you specify

(2) Updates the chain that you specify for each member

The chains updated by DMGCMFN remain stored in the EDL. Therefore, you can change font lists without renormalizing forms. The updated chains are effective for multiple Documerge runs.

If you want to change the font lists of forms for a single Documerge run, use the DMGMERGE command COMMONFONTS. This command changes the font lists dynamically in memory. Refer to "The COMMONFONTS Command" on page 381 for more information.

IMPORTANT

Back up the EDL before you run DMGCMFN.

Use the VLMMAINT command **BACKUP** to write the EDL to a sequential file.

DMGCMFN can generate a new font list in one of two ways:

- from the font names that you specify in the FONTS= control card
- from an existing Output Environment Definition (ENVDEF), with an option to specify a Font Group Definition (FGRPDEF) other than the one from the ENVDEF

DMGCMFN issues an informational message if it can't update a chain in the EDL.

NOTE

Neither the DMGCMFN utility nor the COMMONFONTS command support Dynacomp forms. You must reprocess the Dynacomp source files with DCOPCOMP and specify the new Common Font List in the ENVDEF.

DMGCMFN Sample JCL

```
//DMGCMFN   ** put your job card here **
//*
//* ********************************************************************
//* ** DMGCMFN JCL executes DMGCMFN to update the font lists of EDL  **
//* ** members specified in the MEMLIST selection file to match a    **
//* ** common font list.  The common font list is either explicitly  **
//* ** defined by the FONTS= keyword or extracted from an environ-   **
//* ** ment definition named by the ENVDEF= keyword.                 **
//* **                                                               **
//* ** This sample JCL demonstrates the DMGCMFN control cards needed **
//* ** to use common font information from an environment definition **
//* ** and a font group definition.                                  **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//CMFN     EXEC PGM=DMGCMFN,REGION=4M,
//  PARM=' / WORKBUFF=500K'
//VLM2LIB  DD DSN=documerg.v03r02.edl,DISP=SHR
//PEDEF    DD DSN=your.documerg.v03r02.pelib,DISP=SHR
//SYSOUT   DD SYSOUT=*
//MESSAGE  DD SYSOUT=*,
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//STATLOG  DD SYSOUT=*,
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD SYSOUT=*
//WRKFIL   DD DSN=&&DSN,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476
//MEMLIST  DD DSN=your.vlam.member.select.list,DISP=SHR
/*
//SYSIN    DD *
  COMMONFONTS
  ENVDEF=xxxxxx
  FGRPDEF=yyyyyy
/*
//
```

DMGCMFN Input Files

SYSIN	The input to the DMGCMFN utility.
MEMLIST	The Member Selection List. Each record in this file specifies the form name, revision level, and chain name of an EDL member to update. These member selection records must be in a specific format. See "The Member Selection List (MEMLIST)" on page 121.
VLM2LIB	The Electronic Document Library (EDL) containing the chains to be updated.
PEDEF	The Output Environment Definition (ENVDEF) that contains the new Common Font List.

DMGCMFN Output Files

MESSAGE	DMGCMFN messages.
STATLOG	Lists each MEMLIST record that was processed. Indicates whether the corresponding EDL member was updated successfully. See "The DMGCMFN Status Log (STATLOG)" on page 123.
VLM2LIB	The Electronic Document Library (EDL) containing the updated chains.

The Member Selection List (MEMLIST)

The Member Selection List (MEMLIST) lists the forms whose font lists you want to change. You specify the MEMLIST file as input to DMGCMFN.

The following example shows the contents of a MEMLIST file:

Sample Contents of the DMGCMFN Member Selection List (MEMLIST)

```
MEM="LM.LIFE.BANNER"(00001)                    CHAIN=META
MEM="LM.LIFE.CONTRACT"(00001)                  CHAIN=META
MEM="LM.LIFE.CONTRACT.IMPOSE"(00001)           CHAIN=META
MEM="LM.LIFE.DEC"(00001)                       CHAIN=META
MEM="LM.LIFE.DEC.IMPOSE"(00001)                CHAIN=META
MEM="LM.LIFE.ENDORSE1"(00001)                  CHAIN=META
MEM="LM.LIFE.ENDORSE1.IMPOSE"(00001)           CHAIN=META
MEM="LM.LIFE.ENDORSE2"(00001)                  CHAIN=META
MEM="LM.LIFE.ENDORSE2.IMPOSE"(00001)           CHAIN=META
MEM="LM.LIFE.ENDORSE3"(00001)                  CHAIN=META
MEM="LM.LIFE.ENDORSE3.IMPOSE"(00001)           CHAIN=META
MEM="LM.LIFE.OVERLAY"(00001)                   CHAIN=META
MEM="LM.LIFE.TRAILER"(00001)                   CHAIN=META
14 members found for command DIRECTORY, statement 1
```

NOTE

The last record in a MEMLIST file is a summary record and is not valid input to DMGCMFN. For example:

```
14 members found for command DIRECTORY, statement 1
```

Delete this record, or place an asterisk (*) in the first column.

Creating the MEMLIST File

The VLMMAINT utility gives you a convenient way to create the MEMLIST file: Use the DIRECTORY command to create the MEMLIST as a special report and route it to a separate output file. You can modify this file before you run DMGCMFN.

NOTE

You can code the MEMLIST as a flat file, independent of VLMMAINT. If you choose this option, you must observe the rules listed in "Modifying the MEMLIST File" on page 123.

Code the DIRECTORY command with the following keyword parameters:

Parameter	Value
MEM=	The names of the forms. You can code one form name or multiple form names. Also, you can use wild card characters to specify all of the forms in the EDL. Refer to "VLMMAINT Control Statement Syntax" in *Using VLAM* .
OUTPUTDD=	The DD name of the dataset to which VLMMAINT writes the MEMLIST file. This parameter generates the MEMLIST separately from the rest of the VLMMAINT output.
TYPE=PARM	This parameter tells VLMMAINT to generate the MEMLIST file in the format that DMGCMFN expects.
CHAIN=	The name assigned to Xerox Metacode forms in the EDL.
LIBDD=	The DD name of the EDL that contains the forms.

Sample VLMMAINT JCL to Create the DMGCMFN Member Selection List (MEMLIST)

```
//* ** This s//MEMLIST  ** put your job card here **
//*
//* ****************************************************************
//* ** MEMLIST JCL executes VLMMAINT DIRECTORY command with TYPE=PARM**
//* ** to generate member selection file for input to DMGCMFN.       **
//* **                                                               **
//* ** This sample JCL demonstrates the VLMMAINT control cards needed**
//* ** to build a list of all members in VLM2LIB which have META     **
//* ** chains.                                                       **
//* ****************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//VLMDIR   EXEC PGM=VLMMAINT,REGION=2048K
//VLM2LIB  DD DSN=documerg.v03r02.edl,DISP=SHR
//MESSAGE  DD SYSOUT=*,
//            DCB=(RECFM=FB,LRECL=133,BLKSIZE=3990)
//LISTING  DD SYSOUT=*,
//            DCB=(RECFM=FB,LRECL=133,BLKSIZE=3990)
//REPORT   DD SYSOUT=*,
//            DCB=(RECFM=FB,LRECL=133,BLKSIZE=3990)
//MEMLIST  DD DSN=your.vlam.member.select.list,
//            DISP=(,CATLG,DELETE),
//            UNIT=sysda,SPACE=(TRK,1),
//            DCB=(RECFM=FB,LRECL=133,BLKSIZE=23408)
//SYSPRINT DD SYSOUT=*
//DD01     DD *
//SYSIN    DD *
 DIRECTORY MEM='*'(*)
           TYPE=PARM
           CHAIN=META
           OUTPUTDD=MEMLIST
           LIBDD=VLM2LIB
/*
//
```

NOTE

If a form requires a VLAM password, you must modify the form's MEMLIST record after you create the MEMLIST. See "Modifying the MEMLIST File" on page 123.

Modifying the MEMLIST File

You can modify the records in the Member Selection List before using it as input to DMGCMFN. However, you must observe the following rules:

- The entire form name must be enclosed with double quotation marks.
- The form's revision level is required. It must be enclosed with parentheses, and it must be numeric. No spaces are allowed between the double quotation mark following the form name and the left parenthesis preceding the revision level.
- If a form requires a password, you must code the **PASSWORD=** parameter with the VLAM password. For example:

```
MEM="LM.LIFE.BANNER"(00001)  CHAIN=META  PASSWORD=xxxxxxxx
```

- No spaces are allowed between parameters (MEM=, CHAIN=, and PASSWORD=) and their values.
- At least one space must separate each parameter and its value from the next.

The parameters and values can be coded in any order. For example, you could code:

```
MEM="LM.LIFE.BANNER"(00001)  PASSWORD=xxxxxxxx  CHAIN=META
```

NOTE

You can tell DMGCMFN to ignore a form by placing an asterisk (*****) in column one of the form's record. (The first column of each record in the MEMLIST is blank.)

The DMGCMFN Status Log (STATLOG)

This DMGCMFN output file lists the result of DMGCMFN processing for each form in the Member Selection List (MEMLIST).

For each form named in a STATLOG record, one of three results are possible:

Bypass	The form's font list already matches the Common Font List; therefore, the form was not changed. DMGCMFN does not generate a message.
Fail	The form was not changed because of an error. DMGCMFN generates a message that describes the error.
Update	The form's font list was changed successfully.

Sample DMGCMFN Status Log (STATLOG)

```
Record
Number  Result  Selection List Record
------  ------  ----+----1----+----2----+----3----+----4----+----5----+----6---
     1  Fail     MEM="DM202TST"(00001)                        CHAIN=META
     2  Update   MEM="LM.LIFE.BANNER"(00001)                  CHAIN=META
     3  Update   MEM="LM.LIFE.CONTRACT"(00001)                CHAIN=META
     4  Update   MEM="LM.LIFE.CONTRACT.IMPOSE"(00001)         CHAIN=META
     5  Update   MEM="LM.LIFE.DEC"(00001)                     CHAIN=META
     6  Update   MEM="LM.LIFE.DEC.IMPOSE"(00001)              CHAIN=META
     7  Bypass   MEM="LM.LIFE.ENDORSE1"(00001)                CHAIN=META
     8  Bypass   MEM="LM.LIFE.ENDORSE1.IMPOSE"(00001)         CHAIN=META
     9  Update   MEM="LM.LIFE.ENDORSE2"(00001)                CHAIN=META
    10  Update   MEM="LM.LIFE.ENDORSE2.IMPOSE"(00001)         CHAIN=META
    11  Update   MEM="LM.LIFE.ENDORSE3"(00001)                CHAIN=META
    12  Update   MEM="LM.LIFE.ENDORSE3.IMPOSE"(00001)         CHAIN=META
    13  Bypass   MEM="LM.LIFE.OVERLAY"(00001)                 CHAIN=META
    14  Update   MEM="LM.LIFE.TRAILER"(00001)                 CHAIN=META
```

DMGCMFN Commands

The COMMONFONTS Command

COMMONFONTS Control Cards

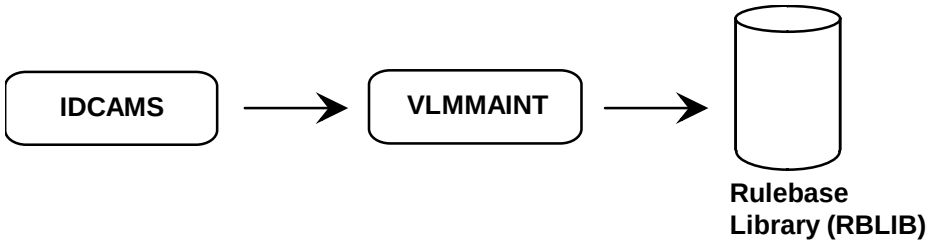
Control Card	Value
ENVDEF=	The name of the Output Environment Definition (ENVDEF) that contains the new Common Font List. You must code either ENVDEF= or FONTS=. You cannot code them both. Code ENVDEF= if you want DMGCMFN to use a Common Font List from an existing ENVDEF. Specify 1 to 6 characters, not including the DE prefix assigned by the PEDEF utility DPLDUTL. If you do not specify the FGRPDEF= control card, DMGCMFN uses the Font Group Definition (FGRPDEF) from the Printer Definition (PRINTDEF) named in this ENVDEF.
FGRPDEF=	The name of a Font Group Definition (FGRPDEF) other than the one from the Printer Definition (PRINTDEF) called by the Output Environment Definition (ENVDEF) named in the ENVDEF= control card. Optional. Valid only if you code the ENVDEF= control card. If you code the FONTS= control card, you cannot code FGRPDEF=. Specify 1 to 6 characters, not including the DE prefix assigned by the PEDEF utility DPLDUTL. The default is the Font Group Definition (FGRPDEF) from the Printer Definition (PRINTDEF) called by the Output Environment Definition (ENVDEF) named in the ENVDEF= control card.

FONTS=	The new Common Font List.
	You must code either FONTS= or ENVDEF=; you cannot code them both. If you code the FGRPDEF= control card, you cannot code FONTS=.
	Code FONTS= if you want to define the Common Font List explicitly, independent of an ENVDEF.
	Specify the names of the Metacode fonts you want to include in the Common Font List. The font names must be:
	▪ enclosed in parentheses
	▪ separated by blanks (do not separate font names with commas)
	▪ from 1 to 6 characters (only the first 6 characters are used)
	The FORMSX font name should be included, and can be coded anywhere within the parentheses.
	You can split the font names between lines by using the hyphen as a continuation character.
	For example:
	`COMMONFONTS FONTS=(PR111E PR211E -`
	`UN104E FORMSX -`
	`P07TDC)`
	Documerge does not limit the number of fonts you can code in FONTS=. However, the Xerox maximum is 127. And, depending on font size and available storage, printers often limit the number of fonts to less than 100.

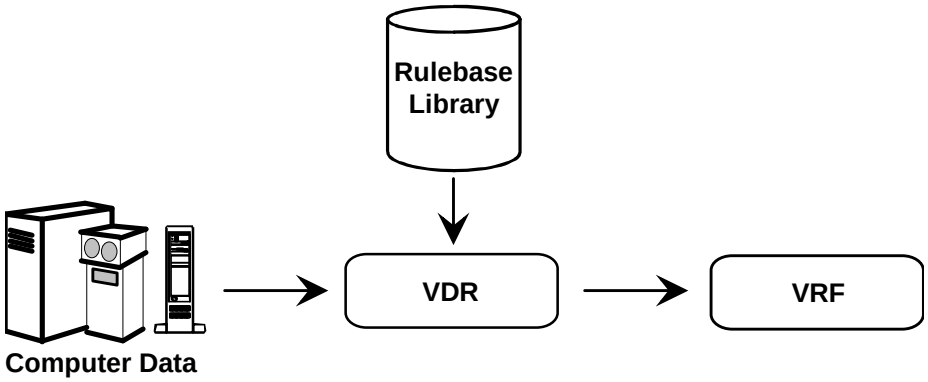
The Rulebase Library

The Rulebase Library (RBLIB) is a database that's defined as a VSAM Relative Record Data Set (RRDS) . You define this RRDS through IBM's IDCAMS utility. (Refer to *VSAM Catalog Administration: Access Method Services Reference* for more information.)

After defining the RRDS, you format it through Oracle VLMMAINT utility, using the FORMAT command. This is the same method that's used to create an EDL. You do all other maintenance through the Rulebase Maintenance Utility (DMGRBMUT).



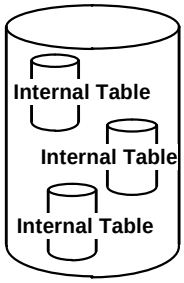
The Rulebase Library is a VLAM library that stores the rules under which a Document Package is constructed.



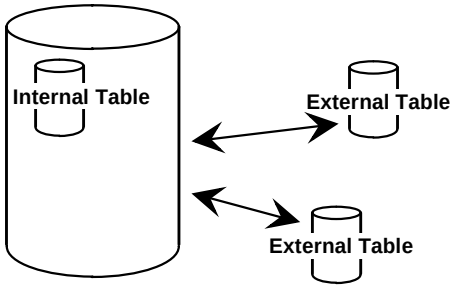
NOTE

Do not use a mainframe file-management system to archive the Rulebase Library, especially if the Rulebase Library has been given secondary allocation space.

The rules are stored as tables in the Rulebase Library. Rules can be created as internal tables or external tables. Internal tables can be created within another table. External tables can be appended within another table by using the INCLUDE command. (See "INCLUDE Command" on page 152 for more information). You can also create a combination of internal and external tables.



Rulebase Table with Internal Tables



Rulebase Table with Internal and External Tables

Only the following VLMMAINT utility program commands can be used for a Rulebase Library:

- FORMAT
- BACKUP
- RESTORE

The FORMAT command is used when the library is initially created. The BACKUP command is used to produce periodic backup files of the Rulebase Library. The RESTORE command is used in the event that the Rulebase Library has to be recreated from the backup file. There are other VLAM utility functions that may be needed for special circumstances. Refer to *Using VLAM* for more information.

Consult your technical support group for more information on how these procedures are handled in your company. All other maintenance to the Rulebase Libraries is done by the Rulebase maintenance program DMGRBMUT commands.

CAUTION

If you are using another Oracle product with Documerge, review that product's documentation for changes necessary to the Rulebase Library.

Rulebase Library Table Types

The Rulebase Library contains seven different types of tables:

(1) Rulebase Table
The primary table that is referenced by your application. When the VDR passes a Rulebase name to DMGRFMT, this table is accessed first. This is a mandatory table.

(2) Tag Table
Contains the BPSD tag names assigned to variable data, the BPSD tag length and the position of the variable data in the input file. This table is optional. However, it is strongly recommended that every Rulebase contain a Tag Table with the DMG.MERGESET.ID Reserved Tag. This Reserved Tag is used by Documerge to reference Document Packages in error.

IMPORTANT!

The prefix **DMG.** identifies Documerge Reserved Tags. Oracle reserves this convention for current and future Documerge Reserved Tags.

Code the prefix **DMG.** for a tag name only if you are coding one of the Documerge Reserved Tags documented in Chapter 8.

(3) Group Table
Contains the name(s) of recipients of Document Packages, an optional sort key, and specifies the Structure Rule. This is a mandatory table.

(4) Structure Rule
Specifies the order in which the forms are to be printed. The DTNs of the EDL members are coded with their Print Options in the order the form is to be printed within a group's output. This is a mandatory table.

(5) Forms Table
Specifies the EDL members that are mandatory (implicit) to a Document Package. This is an optional table.

(6) Imposition Definition
Contains the rules for Imposition (booklet) printing. This is an optional table.

(7) Logical Page Definition
Divides a Single-Sided Image (SSI) into smaller areas for creating special formats, such as columns.

The Rulebase Table (the primary table) can have any combination of the other tables.

Building Internal Tables

All Rulebase processing information can be contained within a single Rulebase member, the Rulebase Table.

Sample Control Cards for Building an Internal Table

```
ADD RULEBASE NAME=XXXXXXX(REV#)
  TAG NAME=XXXXXXX POS=### LENGTH=#####
  FORM NAME=XXXXXXX(REV3)
  GROUP NAME=XXXXXX -
    SORT=(XXXX)
    STRUCTURE RULE=(DTN  -
       (DTN IMPDEF LEFT='##' RIGHT='##' -
        DELTA='##' EXTRAPAGES=XXX) )
```

The above example creates a single Rulebase member, the Rulebase Table. The Rulebase Table contains all the following processing information:

- A tag name
- A form name
- A Group name that contains:
  - A sort field
  - A Structure Rule that contains an Imposition Definition

Table Identifiers

DMGRBMUT assigns a two-byte table identifier based on the specified member type at the time the table is added. The VLAM member name contains this two byte table identifier followed by the table name. Following are the two byte table indicators:

R	Rulebase Table
T	Tag Table
F	Form Table
G	Group Table
S	Structure Rule
I	Imposition Definition
P	Logical Page Definition

Revision Levels

Revision-level values can be assigned to Rulebase Library tables. Valid values are:

Numeric value	A user-specified revision number
0	If the table does not exist, a revision depends on whether you are referencing or adding a table.

Chains

Chain values are assigned by DMGRBMUT and cannot be user defined. DMGRBMUT assigned values are:

Chain Value	Description
TAG	Tag Table
FORM	Forms Table
GRP	Group Table
STR	Structure Rule Table
IMP	Imposition Definition Table
LPG	Logical Page Definition Table

The DMGRBMUT Program

The Rulebase Library (RBLIB) is maintained by the utility program DMGRBMUT. DMGRBMUT is used for initial creation, maintenance, and reporting of the Rulebase Tables.

Sample JCL for DMGRBMUT

```
//DMGRBMUT  ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* **      DOCUMERGE V. 3.2 RULEBASE MAINTENANCE UTILITY             **
//* **                                                                **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGRBMUT EXEC PGM=DMGRBMUT,REGION=3M,
//  PARM=' / WORKBUFF=500K'
//RBLIB    DD DSN=documerg.v03r02.rbl,DISP=SHR
//*
//* DMGRBMUT MESSAGE FILE
//MESSAGE  DD SYSOUT=*,DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//*
//* DMGRBMUT REPORT  FILE
//REPORT   DD SYSOUT=*,DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//*
//WRKFIL   DD DSN=&&WRKFIL,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                  HALF TRACK
//SYSPRINT DD SYSOUT=*
//SYSIN    DD  *
 dmgrbmut control cards
/*
//
```

DMGRBMUT EXEC Parameters

Parameter	Explanation
MAXTOKENS=	Optional. Specifies the maximum number of tokens in the ISITOKN utility's token table. Code a number from **1** to **32760**. (The default value is 8187.) At DMGMERGE run time, the ISITOKN utility divides the DMGRBMUT program commands into logical units. Then ISITOKN divides the units into tokens. ISITOKN stores these tokens in a table. You can use MAXTOKENS= to increase the size of the token table according to the complexity of your DMGRBMUT commands. This increase is limited only by the size of your available storage in the job step. Or, you can reduce your storage requirements by decreasing the MAXTOKENS= value. This decrease must be appropriate to the complexity of your DMGRBMUT commands. If DMGRBMUT commands result in more tokens than the token table can contain, processing stops and the following message is generated: `DMGRBU442F  Maximum number of source command tokens exceeded` If you receive this message, increase the MAXTOKENS= value and rerun DMGRBMUT from the point at which the processing stopped.

Parameter	Explanation

	Tip: There is a limit of 32760 tokens in the token table. A token is defined as any of the following:
	▪ Any word delimited by blanks
	▪ Any left or right parenthesis
	▪ Anything to the left or right of an equal sign (e.g., *keyword=value* is split into two tokens)
	This limit is per command—all SYSIN up to and including one that does NOT have a hyphen continuation. By design, either a new command follows, or the end of SYSIN follows, when there is no hyphen continuation on a SYSIN record.
	The following are the default settings and need not be coded in the structure: **SIM**, **SEP**, **POR**, **MAI**, **STA**, **ANY**, **(0)** (e.g., revision zero).
	For example, instead of coding:
	`( 7 DUP SEP POR FRO MAI STA -`
	`OVL=('TEST'(0) ALWAYS) -`
	`FVL=('BCINS1OV'(0) NONBLANK)) -`
	You could instead code:
	`( 7 DUP FRO -`
	`OVL=('TEST' ALWAYS) -`
	`FVL=('BCINS1OV' NONBLANK)) -`
	In the previous examples, the “old” code has 25 tokens, whereas the “new” code only contains 15. Note that by eliminating “(0)”, you eliminate three tokens.
	Even the overlay “when” option can be omitted—the NONBLANK and ALWAYS values. These instead can be specified with OVLDEFAULT= / FVLDEFAULT= / BVLDEFAULT= parameters in a DMGMERGE SYSIN GLOBAL command. This action would reduce the above to 13 tokens—almost half of the original 25.
	For example:
	`...DMGRBMUT SYSIN...`
	`( 7 DUP FRO -`
	`OVL=('TEST') -`
	`FVL=('BCINS1OV')) -`
	`...DMGMERGE SYSIN...`
	`GLOBAL OVLDEFAULT=ALWAYS FVLDEFAULT=NONBLANK`

NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program that manages storage for many Documerge programs. Code a number from **2** to **2048**. The memory allocated for LM/MM is this number multiplied by the block size of the WRKFIL. (See "WRKFIL" on page 378.) You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)
VLMACCESS=	Used to specify that DMGRBMUT is to open a Rulebase as *read only* so that REPORT commands can be used while the Rulebase is protected from unauthorized changes. **RO** *or* **READONLY** Read-only access. Access dates are NOT written to the VLAM Libraries. If a security system is installed, a user with READ only authority may submit this step for execution. **RW** *or* **READWRITE** (Default) read-write access. Access dates are written to the VLAM Library. If a security system is installed, a user must have UPDATE authority to submit this step for execution. READWRITE is the default and is more efficient than READONLY.

Parameter	Explanation
	NOTE: Unauthorized updating of the Rulebase can only be prevented by use of a file security system such as RAC/F. VLMACCESS does not verify authorization.
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. If the memory allocation is smaller than the amount of data, LM/MM uses the WRKFIL for secondary space. (See "WRKFIL" on page 378.) The WORKBUFF= value must be at least twice the value of the WRKFIL block size. The WORKBUFF= value must not exceed the WRKFIL block size multiplied by 2048. Valid values are: **nnnK** nnn units of 1024 bytes **nnnM** nnn units of 1,048,576 bytes (1024 — 1024) You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)

DMGRBMUT Files

The following are DMGRBMUT files in "Sample JCL for DMGRBMUT" on page 129.

- **DMGRBMUT EXEC**
  Execution of the Rulebase maintenance program DMGRBMUT. The region size varies depending on the combination of commands being processed. A region size of at least 3M is recommended for most functions.
- **RBLIB**
  Contains the Rulebase Library.
- **MESSAGE**
  Contains messages from DMGRBMUT.
- **REPORT**
  Contains the name of the file or SYSOUT class to which the REPORT command writes output. This report file is created using machine carriage control characters. Therefore, you must use the proper method of routing the file to your printer to produce the proper spacing.
- **WRKFIL**
  An internal work file.
- **SYSIN**
  Contains control cards for DMGRBMUT.

DMGRBMUT Syntax

Following are the rules for DMGRBMUT control card syntax:

- 72-byte card image.
- Command parameters can start in any column.
- Hyphen in column one with a trailing blank is treated as a comment.
- Hyphen after a command string is treated as a continuation character.
- Table names containing blanks or semicolons must be enclosed in single quotes (e.g., NAME='TABLE NAME').
- Table names containing single quotes must have the single quote coded twice (e.g., NAME='TABLE''S NAME').
- Table names are a maximum of 30 characters with a one (1) to five (5) character revision level.

Table Name Requirements

A Rulebase Table is identified by its table name and revision level in the NAME= parameter. The following rules apply to table names.

- Length is 1 - 30 characters
- Invalid table name characters are:
  - x'00' Hexadecimal zero
  - * Asterisk
  - ? Question mark
  - () Left and Right parenthesis

If a table name contains a blank or a single or double quotation mark, the name must be enclosed in single quotation marks. If the name contains a single quotation mark it must be doubled (keyed twice) for each quotation mark that will remain within the name.

The quotation marks for the table name *must not* enclose the revision level specification. For example:

```
NAME='Two ''Single'' Quotes'(2)
```

Wildcard Characters and Table Name References

Two wildcard characters may be used to specify a table name for NAME= parameter. The two characters are the question mark (?) and the asterisk (*). The question mark represents any single character. The asterisk represents any number of characters within the name.

Any combination of wildcard characters may be used so long as it does not require the table name to exceed 30 characters in length.

Wildcard characters are supported only by the following DMGRBMUT commands:

- DELETE
- REPORT

DMGRBMUT Major Commands

DMGRBMUT major commands indicate the function to be performed.

- ADD
- COPY
- DEFAULT
- DELETE
- END
- RENAME
- REPORT

ADD Command

The DMGRBMUT **ADD** command is used to write new tables to the Rulebase Library. The specific result depends on whether a table already exists with the same name as that supplied with this command, the revision level specified and the ACTION= parameter.

Sample Control Cards for DGMGRBMUT ADD Command Cards

```
ADD tabletype -
      NAME=XXXXXXX(REV#) -
      ACTION=XXXXX -
      DESCRIPTION=XXXXXXXXXXXXX
```

ADD Control Cards

The following are DMGRBMUT ADD control cards in "Sample Control Cards for DGMGRBMUT ADD Command Cards" on page 133.

tabletype

Indicates the type of external Rulebase Table to be added. With some restrictions some of the external tables can be included within another table. (See "INCLUDE Command" on page 152 for more information.) Valid values are:

- **RULEBASE**
  The primary Rulebase Table is created.
- **GROUPTABLE**
  A table containing group names is created.
- **TAGTABLE**
  A table containing tag names, lengths and positions is created.
- **FORMSTABLE**
  A table containing EDL form names that are mandatory (implicit) is created.
- **STRUCTURE**
  A rule containing the sequence of DTNs and their associated Print Options is created.
- **IMPDEF**
  An Imposition Definition specifying page layout is created.
- **LPGDEF**
  A Logical Page Definition specifying areas in an SSI is created.

NAME=tablename(revisionlevel)

- **tablename**

  The name of the Rulebase Table being added. A name is mandatory. If the ACTION=NEWREV parameter is specified, a new table is added at the first, the next highest or at the specified revision level. If the ACTION=REPLREV is coded, a table at the specified revision level is replaced.

  You can have a 1- to 30-character name for the table. ADD is a major command with minor commands that can also be coded after the NAME= parameter. See "DMGRBMUT Minor Commands" on page 149 for more information.

- **revisionlevel**

  The revision level of the table name. Revision level is optional. There can be a 1- to 5-character revision level supplied. The revision level that is created for a table depends on the existence of the table and whether the revision level is omitted or a specific value is coded.

  The valid values are:

0 *or* **omitted**	Add the table at the next highest level. If the table name does not exist, it will be created and assigned revision level one (1).
n	Add the table at a specific level. The value for the revision can range from 1 to 32,767.

ACTION=

What action to take if the table already exists with the same name and revision level. You cannot ADD duplicate tables with the same name and revision level. Valid values are:

- **NEWREV**

  If the table member name already exists write a new table with a revision level one higher than the existing table, otherwise the new table is added at revision level one (1). This is the default.

  If ACTION=NEWREV is coded and the named table and revision level does not exist, then a new table is created with the specified revision level.

  Another method for replacing tables is to first delete the table and revision level then *add* it with the ACTION=NEWREV parameter.

- **REPLREV**

  If no revision level is specified, the highest revision of the member is replaced. This is optional.

  If ACTION=REPLREV is coded, the named table and revision level is replaced in its entirety with the new information that is supplied in the ADD command.

DESCRIPTION=

A 1- to 36-character description of the table. This description is for reporting purposes only. Documerge does not depend on the description for processing. DESCRIPTION= is optional with no default value.

COPY Command

The DMGRBMUT COPY command is used to copy existing Rulebase Tables. Tables can be copied within the library.

Sample Control Cards for the DMGRBMUT COPY Command

```
COPY tabletype -
          FROM=XXXXXXX(REV#) -
          TO=XXXXXXX(REV#) -
          ACTION=XXXXX -
          DESCRIPTION=XXXXXXXX
```

COPY Control Cards

The following are the DMGRBMUT COPY control cards in "Sample Control Cards for the DMGRBMUT COPY Command" on page 135.

tabletype

Indicates the type of Rulebase Table copied. Valid values are:

- **RULEBASE**
  The parent table is copied.
- **GROUPTABLE**
  A table containing group names is copied.
- **TAGTABLE**
  A table containing tag names, lengths and positions is copied.
- **FORMSTABLE**
  A table is copied containing EDL form names that are implicit in this Merge Set.
- **STRUCTURE**
  A rule containing the sequence of DTNs and their associated Print Options is copied.
- **IMPDEF**
  An Imposition Definition specifying page layout is copied.
- **LPGDEF**
  A Logical Page Definition specifying areas in an SSI is copied.
- **ALL**
  Indicates that any table, regardless of type, with a name equal to the FROM name will be copied.

FROM=tablename(revisionlevel)

- **tablename**
  The name of the Rulebase Table being copied. A name is mandatory. You can have a 1- to 30-character name for the table.
- **revisionlevel**
  The revision level of the table name being copied. Revision level is optional. There can be a 1- to 5-character revision level supplied.

  The valid values are:

0 *or* **omitted**	Copy the highest revision level of the table.
n	Copy the specific revision level of the table. The value for the revision can range from 1 to 32,767.

TO=tablename(revisionlevel)

- **tablename**
  The name of the new Rulebase Table that will be created as a result of the COPY command. A name is mandatory. A new table is created at the first, the next highest, or the specified revision level, depending on the 'revisionlevel' coded. You can have a 1- to 30-character name for the table.
- **revisionlevel**
  The revision level of the table to be created. Revision level is optional. There can be a 1- to 5-character revision level supplied. The revision level that is created for a table depends on the existence of the table and whether the revision level is omitted or a specific value is coded.

  The valid values are:

0 *or* **omitted**	Create the table at the next highest level. If the table name does not exist, it will be created and assigned revision level 1.
n	Add the table at a specific level. The value for — may range from 1 to 32,767.

ACTION=
What action to take if the table already exists with the same name. Valid values are:

- **NEWREV**
  If the table member name already exists write a new table with a revision level one higher than the existing table, otherwise the new table is added at revision level one (1).
- **REPLREV**
  If no revision level is specified, the highest revision of the member is replaced.

This is optional with a default of NEWREV.

DEFAULT Command

The **DEFAULT** command sets the default value for the TAG command's Write If Blank (WIB=) parameter. You can use the DEFAULT command to avoid repetitious coding of the WIB= parameter. See "WIB=" on page 187 for more information about the WIB= parameter.

The DEFAULT command is optional. You can code it in any control card position that precedes your TAG commands. The tags that follow a DEFAULT command use that WIB= value.

Documerge allows multiple DEFAULT commands. Therefore, you can vary the WIB= value among tags. Each subsequent DEFAULT command changes the WIB= value for all TAG commands that follow.

The values for the Write If Blank (WIB=) parameter are:

Y	Write the tag to the VRF if the tag's value is blank. Y is the default value.
N	Do not write the tag to the VRF if the tag's value is blank.

The following example shows the DEFAULT command syntax:

```
DEFAULT WIB=N
```

For existing Rulebase Libraries, Documerge assumes a WIB= value of Y.

Using the DEFAULT command eliminates unnecessary coding when you set up an existing Rulebase Library to use the WIB= parameter.

DELETE Command

The DMGRBMUT **DELETE** major command is used to delete existing Rulebase members.

Sample Control Cards for the DMGRBMUT DELETE Command

```
DELETE tabletype -
        NAME=XXXXXXX(REV#)
```

DELETE Control Cards

Use care when deleting tables that are included by, or referenced by, other tables. No cross check is made. Therefore, you could get an error condition during the VDR or Documerge processing.

The following are the DMGRBMUT DELETE control cards in "Sample Control Cards for the DMGRBMUT DELETE Command" on page 137.

tabletype

Indicates the type of Rulebase Table deleted. Valid values are:

- **RULEBASE**
  The parent table is deleted.
- **GROUPTABLE**
  A table containing group names is deleted.
- **TAGTABLE**
  A table containing tag names, lengths and positions is deleted.
- **FORMSTABLE**
  A table containing EDL names that is mandatory is deleted.
- **STRUCTURE**
  A rule containing the sequence of DTNs and their associated Print Options is deleted.
- **IMPDEF**
  An Imposition Definition specifying page layout is deleted.
- **LPGDEF**
  A Logical Page Definition specifying areas in an SSI is deleted.
- **ALL**
  Indicates that any table regardless of type with a name equal to the value found in NAME= is deleted.

NAME=tablename(revisionlevel)

- **tablename**
  The name of the Rulebase Table to be deleted. A name is mandatory. You can have a 1- to 30-character name for the table.
- **revisionlevel**
  The revision level of the table name to be deleted. Revision level is optional. There can be a 1- to 5-character revision level supplied. The revision level of the table that is deleted depends on whether the 'revisionlevel' operand is omitted or a specific value is coded. Valid values are:

0 *or* **omitted**	Delete the highest revision level of the table.
n	Delete the specific level of the table. The value for the revision can range from 1 to 32,767.

END Command

The DMGRBMUT **END** command terminates the current major command. END lets you see the beginning and ending of one major command.

The END command is optional and used for readability. (Each major command terminates automatically when another major command is encountered.)

Sample Control Cards for the DMGRBMUT END Command

```
ADD -
       NAME=XXXXXXX(REV#) -
       ACTION=XXXXX -
       DESCRIPTION=XXXXXXXXXXXXX
END
COPY -
            FROM=XXXXXXX(REV#) -
            TO=XXXXXXX(REV#) -
            ACTION=XXXXX -
            DESCRIPTION=XXXXXXXX
END
DELETE -
       NAME=XXXXXXX(REV#)
END
RENAME -
            NAME=XXXXXXX(REV#) -
            NEWNAME=XXXX(REV#)
END

REPORT -
             NAME=XXXXXXXXXXX -
             FORMAT=XXXXXXX
END
```

RENAME Command

The DMGRBMUT **RENAME** command is used to assign new names to existing Rulebase Tables. This command renames all revision levels of the specified table name. The revision levels do not change.

The table name specified in the NEWNAME= parameter cannot already exist at the revision levels of the table selected in the NAME= parameter.

Sample Control Cards for the DMGRBMUT RENAME Command

```
RENAME tabletype -
    NAME=XXXXXXX(REV#) -
    NEWNAME=XXXX(REV#)
```

RENAME Control Cards

The following are the DMGRBMUT RENAME control cards in "Sample Control Cards for the DMGRBMUT RENAME Command" on page 138.

tabletype

Indicates the type of Rulebase Library table renamed. Valid values are:

- **RULEBASE**
  The primary Rulebase Table is renamed.
- **GROUPTABLE**
  A table containing group names is renamed.
- **TAGTABLE**
  A table containing tag names, lengths and positions is renamed.
- **FORMSTABLE**
  A table containing EDL names that are mandatory is renamed.

- **STRUCTURE**
  A rule containing the sequence of DTNs and their associated Print Options is renamed.
- **IMPDEF**
  An Imposition Definition specifying page layout is renamed.
- **LPGDEF**
  A Logical Page Definition specifying areas in an SSI is renamed.
- **ALL**
  Indicates that any table regardless of type with a name equal to the value of NAME= is renamed.

NAME=tablename

The name of the Rulebase Library table being renamed. A name is mandatory. You can have a 1- to 30-character name for the table.

NEWNAME=tablename

The new name assigned to the Rulebase Library table that was selected by the NAME= parameter. A new name is mandatory. You can have a 1- to 30-character name for the table.

If the table name specified by the NEWNAME= parameter contains multiple revision levels, all the revisions are renamed with this new name. The revision level values do not change.

REPORT Command

The DMGRBMUT REPORT command can generate a report for each type of Rulebase Library table.

Sample Control Cards for the DMGRBMUT REPORT Command

```
REPORT tabletype -
            NAME=XXXXXXXXXXX -
            FORMAT=XXXXXXX -
```

REPORT Control Cards

The following are the REPORT control cards:

tabletype

Indicates the type of Rulebase Library table to be included in the report. Valid values are:

RULEBASE	The parent Rulebase Library table.
GROUPTABLE	The Groups in the Rulebase Library.
TAGTABLE	The names, lengths, and positions of the tags in the Rulebase Library.
FORMSTABLE	The names of the forms that are specified by the Rulebase Library. These forms are called Implicit Forms.
STRUCTURE	The collation sequence and Print Options of DTNs for a Group.
IMPDEF	The Imposition Definitions, which specify page layout for booklet printing, in the Rulebase Library.
LPGDEF	The Logical Page Definitions, which divide an SSI into smaller areas, in the Rulebase Library.
ALL	A specific table of any type in the Rulebase Library. You specify this table in the NAME= control card.

NAME=tablename(revisionlevel)

- **tablename**
  The name of the Rulebase Library table to be selected for this report. A name is mandatory. You can have a 1- to 30-character name for the table.

  A generic table name can be specified. A generic table name consists of the asterisk (*) and/or question mark (?) characters. See "Table Name Requirements" on page 132 for more information on coding generic table names.
- **revisionlevel**
  The revision level of the table name to be reported. Revision level is optional. There can be a 1- to 5-character revision level supplied. The revision level(s) of the table selected depends on whether the revision level is omitted or a specific value is coded. The valid values are:

0 *or* **omitted**	Select all revision levels of the table name.
n	Select the specific level of the table name. The value for the revision can range from 1 to 32,767.

FORMAT=

The type of report to be produced. This is optional. Valid values are:

- **COMPLETE**
  Provides a report format that includes information contained in the STORAGE and CONTENT reports.
- **CONTENT**
  Provides a report format detailing the code used to create a Rulebase Table. This is the default.
- **INDEX**
  Provides a report format which includes table names, creation date, last update date, and description.
- **STORAGE**
  Provides a report format detailing the chain names, block information, and names of tables included in the member.

Sample DMGRBMUT Reports

Complete Report (FORMAT=COMPLETE)

The Sample Complete Report lists:

- The name of a specified table
- The parameters contained within that table. If the specified table contains an internal table, the Complete report also lists the parameters of the internal table.

Sample Complete Report

```
Run Date: 04/12/98           R U L E   B A S E   M A I N T E N A N C E   R E P O R T      Page   1
Run Time: 15:42:45                      D O C U M E R G E - V03 R02 L00                  Complete
                              All items with names like "MT97", all revisions
Rule Base: MT97(1)                                    Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                                                   Created:  4/12/95 15.13.33
                                                                   Changed:  4/12/95 15.13.35
Tag Definitions: MT97(1)
   Tag Name                          Position    Length   WIB
   ------------------------------   ---------   ------   ---
   NAME                                     1       20    Y
   POLICY.NUMBER                           45       10    Y
   POLICY.DATE                             81       12    Y

Group Definition: MT97(1)
   Group Name              Sort Fields                      Occurrence
   ---------------------   ------------------------------   ------
   INSURED                 NAME                                 1
   FILE                    POLICY.NUMBER                        1

   Structure Rule: MTS1(1)
      Dtn's   Options                          Copy Group          Bin
      -----   --- --- --- --- --- --- --- --- --- --- --- --- --- --------
         10   DUP SEP STA ODD POR                                  MAIN
           Concatenation Set...
         20   DUP CON STA ANY POR OVL                              MAIN
              OVL=('LM.LIFE.OVERLAY(MERGE)' ONLYBLANK)
         30   DUP CON STA ODD POR                                  MAIN
           end of Concatenation Set...

Implicit Forms: MT97(1)
   Form Name (Revision)
   ------------------------------------
   LM.LIFE.DEC(MERGE)
   LM.LIFE.CONTRACT(MERGE)
   External Reference Summary              (Err = Error processing item, NF = Item not found)
      Item Type                  Name (Revision)
      -------------------------  ----------------------------------------
      Structure Rule             MTS1(0)

      Logical Page Definition: MTLPG1(1)
                 --- S h i f —   C — o — d i — a — e — ----
         Type    HFront      VFront      HBack       VBack
         ----    ------      ------      ------      ------
          S        0.50I       1.00I       0.00I       0.00I

                 -------- L — g i c a —   P a g e — -------
         Type    HStart      HSize       VStart      VSize
         ----    ------      ------      ------      ------
          L        0.00I       4.25I       0.00I       5.50I
          L        4.25I       4.25I       0.00I       5.50I
          L        0.00I       4.25I       5.50I       5.50I
          L        4.25I       4.25I       5.50I       5.50I

   Chain Information
      Name       Blocks
      ----   ----------
      TAG             1
      FORM            1
      STR             1
      GRP             1
      LPG             1
             ----------
      Total           5
```

Content Report (FORMAT=CONTENT)

The Sample Content Report lists:

- The name of a specified table
- The parameters contained within that table.

NOTE

This report lists the names but not the contents of any internal tables.

Sample Content Report

```
Run Date: 04/12/98              R U L E   B A S E   M A I N T E N A N C E   R E P O R T     Page  1
Run Time: 15:48:09                          D O C U M E R G E - V03 R02 L00                Content
                                      All items with names like "MT97", all revisions
Rule Base: MT97(1)                                        Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXX

Tag Definitions: MT97(1)
   Tag Name                          Position    Length   WIB
   ------------------------------    ---------   ------   ---
   NAME                                     1       20    Y
   POLICY.NUMBER                           45       10    Y
   POLICY.DATE                             81       12    Y

Group Table: MT97(1)                                      Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXX

Group Definition: MT97(1)
   Group Name              Sort Fields                       Occurrence
   --------------------    ------------------------------    ------
   INSURED                 NAME                                  1
   FILE                    POLICY.NUMBER                         1

   Structure Rule: MT97(1)
      Dtn's   Options                          Copy Group          Bin
      -----   --- --- --- --- --- --- --- --- --- --- --- --- --- --------

   external Structure Table: MTS1(0)

Forms Table: MT97(1)                                      Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXX

Implicit Forms: MT97(1)
   Form Name (Revision)
   ------------------------------------
   LM.LIFE.DEC(MERGE)
   LM.LIFE.CONTRACT(MERGE)

Structure Rule: MTS1(1)                                   Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXX

   Structure Rule: MTS1(1)
      Dtn's   Options                          Copy Group          Bin
      -----   --- --- --- --- --- --- --- --- --- --- --- --- --- --------
         10   DUP SEP STA ODD POR                                  MAIN
           Concatenation Set...
         20   DUP CON STA ANY POR OVL                              MAIN
              OVL=('LM.LIFE.OVERLAY(MERGE)' ONLYBLANK)
         30   DUP CON STA ODD POR                                  MAIN
           end of Concatenation Set...

Logical Page Definition: MSLPG2(1)                        Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXX

      Logical Page Definition: MSLPG2(1)
                --- S h i f —   C — o — d i — a — e — ----
         Type    HFront      VFront      HBack       VBack
         ----    ------      ------      ------      ------
          S        0.50I       1.00I       0.00I       0.00I

                -------- L — g i c a —   P a g e — -------
         Type    HStart      HSize       VStart      VSize
         ----    ------      ------      ------      ------
          L        0.00I       4.25I       0.00I       5.50I
          L        4.25I       4.25I       0.00I       5.50I
          L        0.00I       4.25I       5.50I       5.50I
          L        4.25I       4.25I       5.50I       5.50I
```

Index Report (FORMAT=INDEX)

The Sample Index Report lists:

- The names of the tables in the Rulebase Library
- The date and time each table was created
- The date and time each table was modified
- A user-assigned description of each table.

Sample Index Report

```
Run Date: 04/12/98            R U L E   B A S E   M A I N T E N A N C E   R E P O R T      Page  1
Run Time: 15:48:49                        D O C U M E R G E - V03 R02 L00                  Index
                                    All items with names like "*", all revisions

Rule Bases
Name (Revision)                 Created            Changed             Description
-----------------------------   ----------------   ----------------    -----------------------------
MPRB(1)                          4/12/95 15.13.28   4/12/95 15.13.30   XXXXXXXXXXXXXXXXXXXXXXXXXXX

Imposition Definitions
Name (Revision)                 Created            Changed             Description
-----------------------------   ----------------   ----------------    -----------------------------
IMP2(1)                          4/12/95 15.13.25   4/12/95 15.13.26   XXXXXXXXXXXXXXXXXXXXXXXXXXX

Structure Rules
Name (Revision)                 Created            Changed             Description
-----------------------------   ----------------   ----------------    -----------------------------
MTS1(1)                          4/12/95 15.13.30   4/12/95 15.13.31   XXXXXXXXXXXXXXXXXXXXXXXXXXX

Logical Page Definitions
Name (Revision)                 Created            Changed             Description
-----------------------------   ----------------   ----------------    -----------------------------
MSLPG1(1)                        4/12/95 15.37.15   4/12/95 15.37.16   XXXXXXXXXXXXXXXXXXXXXXXXXXX

Forms Tables
Name (Revision)                 Created            Changed             Description
-----------------------------   ----------------   ----------------    -----------------------------
FORM.TABLE.NAME(REV#)            4/12/95 15.39.40   4/12/95 15.39.41   XXXXXXXXXXXXXXXXXXXXXXXXXXX

Group Tables
Name (Revision)                 Created            Changed             Description
-----------------------------   ----------------   ----------------    -----------------------------
GROUP.TABLE.NAME(REV#)           4/12/95 15.39.40   4/12/95 15.39.41   XXXXXXXXXXXXXXXXXXXXXXXXXXX

Tag Tables
Name (Revision)                 Created            Changed             Description
-----------------------------   ----------------   ----------------    -----------------------------
TAG.TABLE.NAME(REV#)             4/12/95 15.39.40   4/12/95 15.39.41   XXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Storage Report (FORMAT=STORAGE)

The Sample Storage Report lists:

- The name of tables as chains
- The number of blocks per table.

Sample Storage Report

```
Run Date: 04/12/98             R U L E   B A S E   M A I N T E N A N C E   R E P O R T     Page  1
Run Time: 15:50:56                        D O C U M E R G E - V03 R02 L00                Storage
                                       All items with names like "*", all revisions
Rule Base: MPRB(1)                                        Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                                              Created:  4/12/95 15.13.28
                                                              Changed:  4/12/95 15.13.30
   External Reference Summary                 (Err = Error processing item, NF = Item not found)
      Item Type                    Name (Revision)
      -------------------------    ----------------------------------------
      Structure Rule               IMP1(0)

   Chain Information
      Name       Blocks
      ----   ----------
      TAG             1
      FORM            1
      STR             1
      GRP             1
             ----------
      Total           4

Imposition Definition: IMP2(1)                            Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                                              Created:  4/12/95 15.13.25
                                                              Changed:  4/12/95 15.13.26
   External Reference Summary                 (Err = Error processing item, NF = Item not found)
   ------------------------------
   *** No external references ***
   ------------------------------

   Chain Information
      Name       Blocks
      ----   ----------
      IMP             1
             ----------
      Total           1

Structure Rule: MTS1(1)                                   Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                                              Created:  4/12/95 15.13.26
                                                              Changed:  4/12/95 15.13.28
   External Reference Summary                 (Err = Error processing item, NF = Item not found)
   ------------------------------
   *** No external references ***
   ------------------------------

   Chain Information
      Name       Blocks
      ----   ----------
      STR             1
             ----------
      Total           1

Logical Page Definition: MSLPG1(1)                        Description: XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                                              Created:  4/12/95 15.37.15
                                                              Changed:  4/12/95 15.37.16
   External Reference Summary                 (Err = Error processing item, NF = Item not found)
   ------------------------------
   *** No external references ***
   ------------------------------

   Chain Information
      Name       Blocks
      ----   ----------
      LPG             1
             ----------
      Total           1
```

DMGRBMUT Report Field Descriptions

The following describes each of the fields reported by DMGRBMUT. This is a consolidated list containing the fields from all the reports produced by DMGRBMUT.

Bin

The paper tray from which Documerge will select the paper for this DTN.

Chain Information - Blocks

The number of VLAM chain blocks required to store this Rulebase Table.

Chain Information - Name

The name of the chain of the VLAM member in which this Rulebase Table is stored.

Chain Information - Total

The total number of VLAM chain blocks required to store all the chains associated with this Rulebase Table.

Changed

The date and time the table was last updated.

Copy Group

The COPYGROUPs associated with the DTN.

Concatenation Set...

Comments that are printed to identify the beginning of the DTNs that comprise a Concatenation Set.

Created

The date and time the table was created.

Description

The description of the table that was supplied when the table was first added.

D O C U M E R G E - V03 R02 L01

The version, release and modification level of the Documerge product from which DMGRBMUT was executed.

DTN

The list of Document Type Numbers specified for this structure in the Rulebase.

End of Concatenation Set...

Comments that are printed to identify the end of the DTNs that comprise a Concatenation Set.

End of Imposition Booklet...

Comments that are printed to identify the end of the DTNs that comprise an Imposition Booklet.

External Forms Table:

The name of an external Forms Table that was included in this reported form table or Rulebase.

External Group Table:

The name of an external Group Table that was included in this reported Group Table or Rulebase.

External IMPDEF Table:

The name of an external Imposition Definition table that was referenced from this reported structure table.

External Reference Summary

This section of the DMGRBMUT report summarizes all the external references associated with the Rulebase Table(s). It lists all the *includes* and *references* to external tables.

External Reference Summary - Item Type

The type of external table that was included or referenced.

External Reference Summary - Name(Revision)

The name of the external table that was included or referenced.

External Structure Table:

The name of an external structure table that was referenced from this reported Group Table.

External Tag Table:

The name of an external Tag Table that was included in this reported Tag Table or Rulebase.

Form Name (Revision)

The name of the Implicit Form member in the EDL to be selected by this Rulebase.

Group Name

The name given to identify the group of recipients for the Rulebase.

IMPDEF Extra Pages

When Imposition processing is specified, this identifies the location of any extra blank pages will be placed to print an even number of pages in an imposed document.

IMPDEF Offset Left

For Imposition processing, this specifies the beginning location for the image that is printed at the left side of the page. This is the amount in inches that defines the left margin space.

IMPDEF Offset Right

For Imposition processing, this specifies the beginning location for the image that is printed at the right side of the page. This is the amount in inches that defines the left margin space which is measured from the left edge of the physical paper sheet.

IMPDEF Offset Delta

For Imposition processing, this specifies the amount that both images are to be adjusted to maintain the appearance of even left margins throughout the printing of a booklet. Because the physical pages of a document are folded to create a booklet, each page must be adjusted in or out to maintain even margins throughout the booklet.

Imposition Booklet...

Comments that are printed to identify the beginning of the DTNs that comprise an Imposition Booklet.

Item not found for INCLUDE:

A comment that DMGRBMUT reports when it is unable to read and locate the table in the Rulebase Library. The table name coded in the INCLUDE command is also printed.

Logical Page Definition:

The division of a Single-Sided Image (SSI) into one or more smaller areas, called logical pages. Logical Page Definitions are specified with the LPGDEF command. LPGDEF also shifts front and back SSIs for binding or for 3-hole paper.

Logical Pages — Type

A value that indicates whether the logical page is defined for front SSIs (F), back SSIs (B), or both front and back SSIs (L).

Logical Pages — HStart

The horizontal position (X coordinate) that is the start of the upper left corner of the logical page for the SSI.

Logical Pages — HSize

The width of the logical page.

Logical Pages — VStart

The vertical position (Y coordinate) that is the start of the upper left corner of the logical page for the SSI.

Logical Pages — VSize

The length of the logical page.

Name(Revision)

The name and revision level of the table.

No entries in table

A comment that is printed by DMGRBMUT to report that the table reported did not contain any data or information. Whenever a Rulebase is created without all five Rulebase tables (i.e., Tag, Structure, Group, etc.), this comment will be printed for those tables that were omitted.

*** No external references ***

A comment that DMGRBMUT prints when there are no external references associated with the selected table being reported.

Occurrence

When a tagged field can occur more than once, this tells Documerge which occurrence of the field to use during the sort process.

Options

The user-selected Print Options associated with the DTN. This reports the defaults for any Print Options that were not coded when the Structure table was added.

Shift Coordinates — HBack

The value by which the logical pages on back SSIs are moved horizontally (to the right). If the Logical Page Definition contains no shift values, the heading **Shift Coordinates** and its related fields are not displayed.

Shift Coordinates — HFront

The value by which the logical pages on front SSIs are moved horizontally (to the right). If the Logical Page Definition contains no shift values, the heading **Shift Coordinates** and its related fields are not displayed.

Shift Coordinates — Type

Type S indicates that the Logical Page Definition contains one or more shift values. (The logical pages are moved horizontally or vertically on SSIs according to the shift values.) If the Logical Page Definition contains no shift values, the heading **Shift Coordinates** and its related fields are not displayed.

Shift Coordinates — VBack

The value by which the logical pages on back SSIs are moved vertically (down). If the Logical Page Definition contains no shift values, the heading **Shift Coordinates** and its related fields are not displayed.

Shift Coordinates — VFront

The value by which the logical pages on front SSIs are moved vertically (down). If the Logical Page Definition contains no shift values, the heading **Shift Coordinates** and its related fields are not displayed.

Sort Fields

The list of tagged fields specified to control the sort order for the group.

Tag Length

The tag's field length. This is the maximum length of the variable data in the input file.

Tag Name

The name of the field within the input file. This name matches the BPSD tag name in the composition source.

Tag Position

The tag's starting position of the variable data in the input file. This value is relative to 1 and the first position of the input file.

WIB

A parameter that indicates whether the tag is written to the VRF if the tag's value is blank. Values are Y (the default) and N.

DMGRBMUT Minor Commands

The DMGRBMUT minor commands define the tables in a Rulebase Library. While major commands act upon the tables of a Rulebase Library, minor commands act upon the contents of those tables.

Parameters of minor commands are based upon the related table type. Minor commands are subordinate to the major command ADD or to other minor commands.

The following are the DMGRBMUT minor command and the Rulebase Library tables they define:

- **FORM**
  Implicit Forms Table entries
- **GROUP**
  Group Table entries
- **IMPDEF**
  Imposition printing parameters
- **INCLUDE**
  At run time includes external tables
- **LPGDEF**
  Logical Page Definitions
- **STRUCTURE**
  Structure Rules
- **TAG**
  Tag Table entries

Minor commands can have subcommands, keywords, and parameters to further define the function. The commands, subcommands, keywords, and parameters and the syntax that is required for their use by DMGRBMUT is explained in the following pages.

FORM Command

The DMGRBMUT **FORM** command specifies the EDL member name(s) that are included in the forms table. EDL member name(s) selected by the forms table are produced for every Merge Set that uses this named table. This table is optional and can contain an unlimited number of form names.

Sample Control Cards for the DMGRBMUT FORM Command

```
ADD FORMSTABLE -
           NAME=XXXXXX(REV#)
           FORM NAME=XXXXXXX(REV#)
```

FORM Command Format

This command defines a single entry in an Implicit Forms Table.

The following are the DMGRBMUT FORM control cards in "Sample Control Cards for the DMGRBMUT FORM Command" on page 149.

FORM

- **NAME=**
  The 1- to 32-character EDL member name.
- **revisionlevel**
  The revision level of the EDL member name. Revision level is optional. There can be a 1- to 5-character revision level supplied. Valid values are:

0 *or* **omitted**	Selects the highest revision level of the EDL member at the time of DMGMERGE processing.
n	A numeric value that selects a specific revision level. The value for — can range from 1 to 32,767.
VDR	Selects the highest revision level at the time the VDR is executed.
MERGE	Selects the highest revision level at the time of DMGMERGE processing.

GROUP Command

The DMGRBMUT **GROUP** command indicates the names of Document Package recipients. An unlimited number of Group names can be defined.

Each Group must have a related Structure Rule for collation and printing of forms. Only one Structure Rule can be used per group.

Variable data in the VRF can be sorted for each Group based on sort keys defined in the Group Table.

There are two variations when coding the GROUP command:

- You can code the Structure Rule as an internal table to the Group

Coding a Group with an Internal Structure Rule

```
ADD GROUPTABLE NAME=tablename(revisionlevel)
         GROUP NAME=name    -
         SORT=(xxxxxxx xxxxxxxx xxxxxxxx)  -
         STRUCTURE RULE= ....see "STRUCTURE Command" on page 164 for syntax
```

- You can code an external Structure Rule that contains the definition of the Structure Rule for this Group.

Coding a Group with an External Structure Rule

```
ADD GROUPTABLE NAME=tablename(revisionlevel)
         GROUP NAME=name    -
         SORT=(xxxxxxx xxxxxxxx xxxxxxxx)  -
         STRUCTURE=tablename(revisionlevel)
```

TIP

When you're using the SORT= parameter, its order in relation to two other parameters is extremely important. You're **not** allowed to code it following the STRUCTURE RULE= parameter; however, you can code it after STRUCTURE=.

GROUP Command Format

This command defines three entries in a Group Table.

- **NAME=**
  The 1 to 21 character name used to define each group of recipients. This is mandatory.

- **SORT=**
  The BPSD tag names contained in the VRF that are used for sorting of the VRF records on a group basis. This is an optional parameter, any number of BPSD tag names can be listed but their total field lengths cannot exceed 250 bytes.

TIP

When you're using the SORT= parameter, its order in relation to two other parameters is extremely important. You're **not** allowed to code it following the STRUCTURE RULE= parameter; however, you can code it after STRUCTURE=.

- **STRUCTURE RULE=**
  Specifies the definition of an internal Structure Rule to this Group. See "STRUCTURE Command" on page 164 for the syntax and definition of the DTNs and their Print Options.
- **STRUCTURE=tablename(revisionlevel)**
  References an external Structure Table that specifies the definition of the Structure Rule for this Group. The referenced external Structure Table is maintained separately from this Group and it can be shared between several Groups.
- **revision level**
  The revision level of the Structure table. Revision level is optional. There can be a 1- to 5-character revision level supplied. Valid values are:

0 *or* **omitted**	Selects the highest revision level of the table at the time the VDR is processed. The table must currently exist in the Rulebase Library at the time DMGRBMUT processes this command.
n	A numeric value that selects a specific revision level. The value for — may range from 1 to 32,767. The table and this specific revision level must currently exist in the Rulebase Library at the time DMGRBMUT processes this command. This revision level is stored in the parent table.
RB	Selects the highest revision level at the time DMGRBMUT processes this command. The table must currently exist in the Rulebase Library at the time DMGRBMUT processes this command. The highest revision level is stored in the parent table.
VDR	Selects the highest revision level at the time the VDR is executed. The Rulebase Table does NOT have to exist in the Rulebase Table when DMGRBMUT processes this command. This allows you to include or reference a table that is created at a later date. The VDR generates an error message if the table does not exist at that time.

IMPDEF Command

The DMGRBMUT **IMPDEF** command specifies an Imposition printing definition. Imposition printing is the printing of more than one form on a page in booklet format.

Sample Control Cards for the DMGRBMUT IMPDEF Command

```
ADD IMPDEF -
 NAME=XXXXXXX(REV#)
 IMPDEF LEFT=### -
        RIGHT=### -
        DELTA=### -
        EXTRAPAGES=XXXXX
```

IMPDEF Command Format

This command defines the parameters to generate an imposed-page booklet. These parameters are used during the DMGMERGE step.

- IMPDEF
- The LEFT=, RIGHT=, and DELTA= values can be specified in inches or device units. The valid range for device units is zero (0) through 3300. No fractions are allowed. The valid range for inches is 0.00 through 11.00. The default device is units and the default value is 0.
  - **LEFT=**
    Indicates the offset value in inches or device units from the left edge of the sheet of paper, to where the left logical page should begin. The default value is 0.
  - **RIGHT=**
    Indicates offset value in inches or device units from the left edge of the sheet of paper to where the right logical page should begin. This is mandatory with no default.
  - DELTA=

    This specifies the amount that the left and right images on an imposed page are to be adjusted to maintain the appearance of even margins throughout the printing of a booklet. Because the physical pages of a document are folded to create a booklet, each page must be adjusted in or out to maintain even margins throughout the booklet.

    The left and right offsets specify the beginning position for the outside pages. For each successive inside page, the left offset value is INCREASED by this delta value, and the right offset value is DECREASED by the delta value. Therefore the "text on left" and "text on right" of the imposed page gets closer to each other. A zero delta value tells Documerge not to adjust the left and right page images while printing an imposed document.
  - **EXTRAPAGES=**
    Indicates where blank pages can be generated within the booklet to balance the document. Valid values are:
    - **END**
      Places blank pages as needed at the end of the booklet.
    - **FRONT**
      Places blank pages as needed at the beginning of the booklet.
    - **BALANCE**
      Places blank pages as needed at both the start and the end of the booklet. If needed, an extra blank page is placed at the end.

INCLUDE Command

The INCLUDE command is used to include names of other Rulebase Tables within Rulebase Tables. When using the DMGRBMUT major command ADD and its *tabletype* parameter (see page 133), DMGRBMUT builds the specified table as a unique Rulebase Library member. These tables are referred to as external tables.

A tabletype of RULEBASE must always be created. However, all other table information created with minor commands can follow the RULEBASE parameter. Table information included within other tables are referred to as internal tables.

External Tag Tables, Group Tables and Forms Tables can be called for by other tables with the INCLUDE command. Structure Rules, Imposition Definitions and Rulebase Tables cannot be called for by the INCLUDE command. Structure rules names are called for by the Group Table command, STRUCTURE=. Imposition definition tables are called for by the Print Option IMPDEF=. Rulebase Tables cannot be included within other Rulebase Tables.

Tag, Form, and Group Tables can only include their own type of Table. Tag Tables can only include Tag Tables; Forms Tables can only include Forms Tables; and Group Tables can include Group Tables. There is no limit to the number of INCLUDE statements that can be contained within a table.

In "Sample Control Cards for the DMGRBMUT INCLUDE Command" on page 153, a Rulebase Table is created by the RULEBASE tabletype parameter, which includes the processing rules contained in the Tag Table, Group Table and Forms Table.

Sample Control Cards for the DMGRBMUT INCLUDE Command

```
- EXAMPLE 1
- THE FOLLOWING CREATES A RULEBASE TABLE THAT INCLUDES
- A TAG TABLE, FORM TABLE AND GROUP TABLE
           ADD RULEBASE -
             NAME=XXXXXXX
             INCLUDE TAGTABLE=XXXXXXX(REV#)
             INCLUDE FORMSTABLE=XXXXXX(REV#)
             INCLUDE GROUPTABLE=XXXXX(REV#)
```

In "Sample Control Cards for the DMBRBMUT INCLUDE Command" on page 153, a Tag Table is created by the TAGTABLE parameter, which will contain one tag name and include the tag names contained in the included Tag Table.

Sample Control Cards for the DMBRBMUT INCLUDE Command

```
- EXAMPLE 2
- THE FOLLOWING CREATES AN EXTERNAL TAG TABLE WITH ONE TAG AND INCLUDES
- AND INCLUDES ANOTHER TAG TABLE
            ADD TAGTABLE -
               NAME=XXXXXXX(REV#)
               TAG NAME=XXXXXXX POS=### LENGTH=#####
               INCLUDE TAGTABLE=XXXXXXX(REV#)
```

In "Sample Control Cards for the DMGRBMUT INCLUDE Command" on page 153, a FORMSTABLE is created by the FORMSTABLE parameter, which contains one EDL form name and includes the form names contained in the included Forms Table.

Sample Control Cards for the DMGRBMUT INCLUDE Command

```
- EXAMPLE 3
- THE FOLLOWING CREATES AN EXTERNAL FORMS TABLE WITH ONE FORM AND
- INCLUDES ANOTHER FORM TABLE
          ADD FORMSTABLE -
               NAME=XXXXXXX(REV#)
               FORM NAME=XXXXXXX(REV3)
               INCLUDE FORMSTABLE=XXXXXXX(REV#)
```

In "Sample Control Cards for the DMGRBMUT INCLUDE Command" on page 153, a GROUPTABLE is created by the GROUPTABLE parameter, which will contain one group name that references a structure and includes the Group name(s) contained in the included Group Table.

Sample Control Cards for the DMGRBMUT INCLUDE Command

```
- EXAMPLE 4
- THE FOLLOWING CREATES AN EXTERNAL GROUP TABLE WITH ONE GROUP WHICH
- REFERENCES AN EXTERNAL STRUCTURE RULE AND
- INCLUDES ANOTHER EXTERNAL GROUP TABLE
            ADD GROUPTABLE -
               NAME=XXXXXXX(REV#)
               GROUP NAME=XXXXXXXX -
               STRUCTURE=XXXXXXXXX
               INCLUDE GROUPTABLE=XXXXXXX(REV#)
```

Revision levels can be specified for INCLUDEd tables. Valid values are:

0 *or* **omitted**	Selects the highest revision level of the table at the time the VDR runs. The table must currently exist in the Rulebase Library at the time DMGRBMUT processes this command.
n	A numeric value that selects a specific revision level. The value for the revision may range from 1 to 32,767. The table and this specific revision level must currently exist in the Rulebase Library at the time DMGRBMUT processes this command. This revision level is stored in the parent table.
RB	Selects the highest revision level at the time DMGRBMUT processes this command. The table must currently exist in the Rulebase Library at the time DMGRBMUT processes this command. The highest revision level is stored in the parent table.
VDR	Selects the highest revision level at the time the VDR is executed. The Rulebase Table does NOT have to exist in the Rulebase Table when DMGRBMUT processes this command. This allows you to include or reference a table that will be created at a later date. An error message will be generated by the VDR if the table does not exist at that time.

LPGDEF Command

The DMGRBMUT **LPGDEF** command divides a Single-Sided Image (SSI) into smaller areas, called **logical pages**. A SSI or physical page can contain one or several logical pages, and a single logical page can have the same dimensions as the physical page. Through various combinations of logical pages, you can create special formats, such as columns.

Each logical page in an LPGDEF command consists of four values, relative to the SSI:

(1) Horizontal starting position (X coordinate)

(2) Vertical starting position (Y coordinate)

(3) Horizontal size (width)

(4) Vertical size (length)

TIP

Although you usually define several logical pages inside of a physical page, you can have a physical page with just one logical page defined for it. You can specify zero for the logical-page start-page position, and a non-zero value to cause the page to print lower or to the right (or even higher and to the left if using negative values).

This is a good method for specifying various concatenation page sizes, because the vertical size (VSIZE) of the logical page specifies the concatenation page size, and the logical page vertical size — not the portrait page size — specifies the size limit for concatenation.

The DMGRFMT sub-program places the values of the LPGDEF command in the VRF, in the DMG.LPG.*groupname* Reserved Tag. (See "DMG.LPG.Groupname" on page 316 for more information.)

"Sample LPGDEF Command Control Cards" on page 155 shows four logical pages, defined by the LPAGE= control card. (See "LPAGE=" on page 161 for more information.)

Sample LPGDEF Command Control Cards

```
ADD LPGDEF NAME=LOGICALPAGE1
       LPGDEF -
       FSHIFT=(0.25I 0) -
       LPAGE=(0.00i 0.00i 4.25i 5.5i  -
              4.25i 0.00i 4.25i 5.5i  -
              0.00i 5.50i 4.25i 5.5i  -
              4.25i 5.50i 4.25i 5.5i)
```

Each LPGDEF applies to a DTN or to a group of DTNs. The forms in the DTN are placed into the logical pages you define with the LPGDEF command.

You use the Structure Rule to specify the DTN and Print Options that apply to each LPGDEF. You can specify an LPGDEF for one DTN or a group of DTNs.

For one DTN, you use the standard DTN syntax, shown in the following example:

Coding the LPGDEF Command in Standard DTN Syntax

```
ADD STRUCTURE NAME=MSLPG1
       STRUCTURE -
       RULE=((10 SEP POR ODD MAI STA LPGDEF=MSGLPG2) -
       (20 SEP POR ODD MAI STA LPGDEF=MSGLPG2))
```

To apply an LPGDEF to a group of DTNs, you must use the factored DTN syntax. (See "Factoring DTNs" on page 165 for more information.)

Coding the LPGDEF Command in Factored DTN Syntax

```
ADD STRUCTURE NAME=MSLPG1
       STRUCTURE -
       RULE=((LPGDEF=MSLPG2 -
       ((10 SEP POR ODD MAI STA)
       (20 SEP POR ODD MAI STA))))
```

The LPAGE= control card example in "Sample LPGDEF Command Control Cards" on page 155 defines four logical pages. For this example, assume that: 1) this LPGDEF command applies to a six-page form; 2) the DUPlex and ODD Print Options are specified for the form in the Structure Rule. Due to ODD, DMGMERGE begins processing this LPGDEF by going to a new sheet.

The LPAGE= values for the first logical page are:

```
0.00i 0.00i 4.25i 5.5i  -
```

(1) The horizontal starting position is 0.00 inches.

(2) The vertical starting position is 0.00 inches.

(3) The horizontal size is 4.25 inches.

(4) The vertical size is 5.5 inches.

NOTE

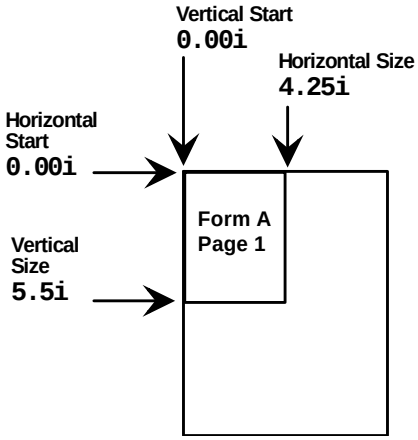
The following conditions are necessary for logical pages:

- The size of a form's current page cannot exceed the size of the current logical page. DMGMERGE does not automatically adjust forms to match logical page sizes.
- The sum of the starting horizontal position, the horizontal size, and the horizontal shift (optional) cannot exceed the total horizontal page size specified in the PRINTDEF.
- The sum of the starting vertical position, the vertical size, and the vertical shift (optional) cannot exceed the total vertical page size specified in the PRINTDEF.
- The horizontal and vertical shift values cannot begin before the first valid horizontal or vertical dot specified in the PRINTDEF.

If any of these conditions is not met, a DMGMERGE error message and printing errors can result. DMGRBMUT and DMGRFMT do not report these errors.

Oracle recommends that you test thoroughly before you use the LPGDEF command in production.

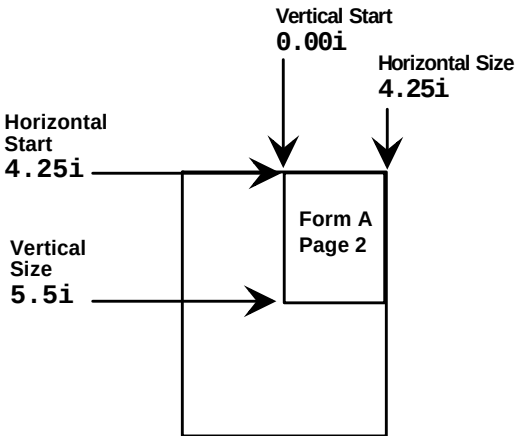
The first page of the form is placed into the SSI area defined by the first logical page.



The LPAGE= values for the second logical page are:

```
4.25i 0.00i 4.25i 5.5i  -
```

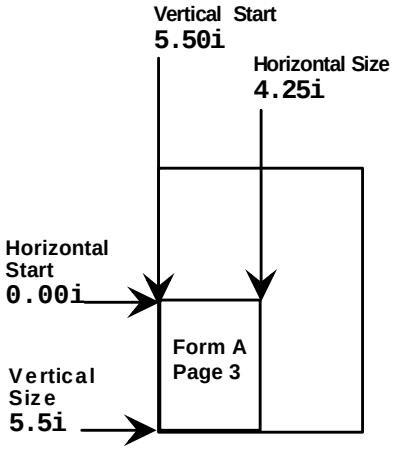
The form's second page is placed into the area defined by the second logical page.



The LPAGE= values for the third logical page are:

```
0.00i 5.50i 4.25i 5.5i  -
```

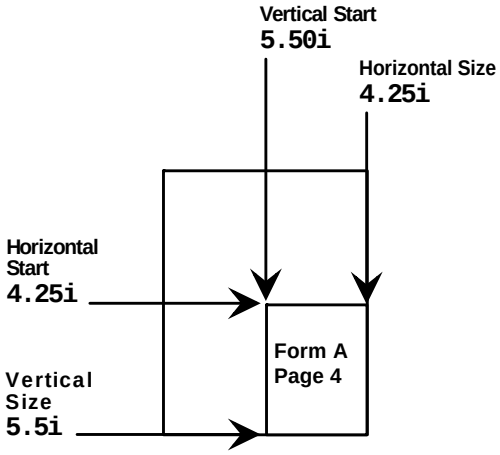
The form's third page is placed into the third logical page.



The LPAGE= values for the fourth logical page are:

```
4.25i 5.50i 4.25i 5.5i  -
```

The form's fourth page is placed into the fourth logical page.



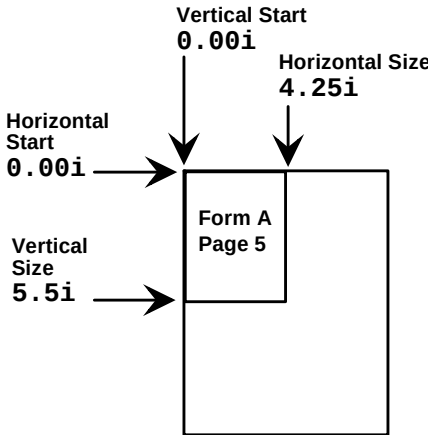
At this point, one page of the six-page form has been placed into each of the four logical pages defined for the SSI. Because two pages of the form remain, DMGMERGE goes to a new SSI. (The new SSI is the back of the current sheet, because DUPlex is specified.)

DMGMERGE applies the same LPAGE= values to the new SSI, beginning with the first logical page. (We use the LPAGE= control card in this example. The FLPAGE= and BLPAGE= control cards define different logical pages for the front and back sides of a sheet. We detail LPAGE=, FLPAGE= and BLPAGE= later.)

Because the fifth page of the form is placed in the first logical page for the new SSI, the first LPAGE= values apply:

```
0.00i 0.00i 4.25i 5.5i  -
```

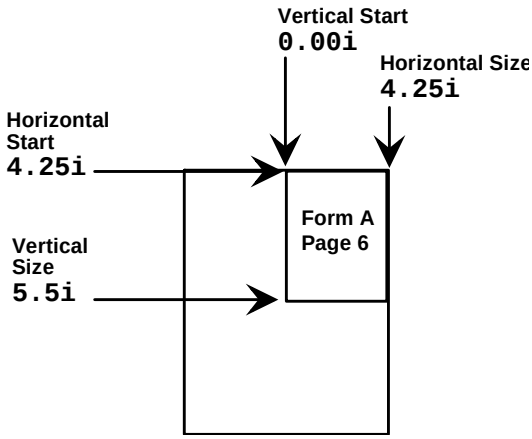
On the new SSI, the fifth page of the form is placed into the first logical page.



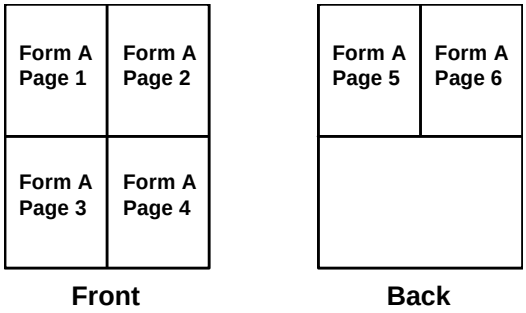
The LPAGE= values for the second logical page are:

```
4.25i 0.00i 4.25i 5.5i  -
```

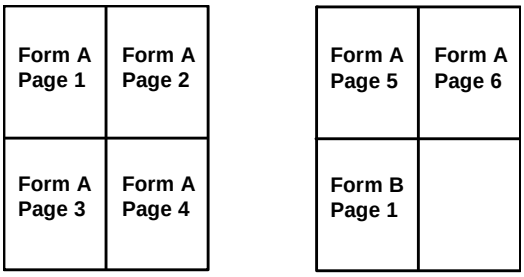
On the new SSI, the sixth page of the form is placed into the second logical page.



At this point, each of the form's six pages has been placed into a logical page. Although two logical pages are still available for the second SSI (the back of the sheet), processing for this form is complete.



If the LPGDEF applies to factored DTNs, the first page of the next form is placed into the next logical page. If this form has multiple pages, DMGMERGE uses the last logical page in the current SSI, and goes to a new SSI if necessary.



However, if the LPGDEF is coded in the standard syntax or if another LPGDEF applies, the remaining two logical pages are not used. DMGMERGE goes to a new SSI (in this case, the front of a new sheet). It places the first page of the next form into the first logical page, using the remaining logical pages and new SSIs if necessary.

When an LPGDEF applies, DMGMERGE goes to the next logical page instead of the next SSI in the following situations:

- The next page of a multiple-page form begins
- A new form that is specified in factored DTN syntax begins
- A new form with the CONcatenate Print Option begins, but the form does not fit in the current logical page

When an LPGDEF applies, DMGMERGE goes to a new SSI in the following situations:

- A new logical page begins but all logical pages for the current SSI are filled
- A new form that is specified in standard DTN syntax begins
- A form begins for which another LPGDEF applies
- No more logical pages are defined

NOTE

When an LPGDEF applies, DMGMERGE processes any page-parity options, such as ODD, EVeN, and FROnt, and Overlays after going to a new SSI. Only the Print Options for the form placed into the first logical page in an SSI are processed; Print Options for forms in the remaining logical pages in the SSI are ignored.

Overlays are printed only once per SSI, not once per logical page. The logical pages in an SSI are not shifted by the Overlay, unless the FSHIFT or BSHIFT control cards are specified. In this case, the entire SSI, including the Overlay, is shifted by the amount of the FSHIFT or BSHIFT value.

The DMGRFMT sub-program places the following values in the Option Fields of the DMG.OPT.*groupname* Reserved Tag in the VRF:

- **BLP** (Begin Logical Page)
- **CLP** (Continue Logical Page)
- **NLP** (No Logical Page — default for VRFs created before Documerge 3.0.)

The DMG.LPG.*groupname* Reserved Tag contains the values of one LPGDEF command for each BLP Option Field in the DMG.OPT.*groupname* Reserved Tag. When DMGMERGE finds a BLP Option Field, it goes to a new SSI. Then it gets the next LPGDEF values from the DMG.LPG.*groupname* tag.

DMGMERGE places the next form into the first logical page in the LPGDEF. When DMGMERGE finds a CLP Option Field, it continues to use the LPGDEF values invoked by the BLP Option Field. It places the next form into the next logical page.

NOTE

You can specify a new SSI while the CLP Option Field is in effect. In the Explicit Forms List (EXP2), the PLP option of the EOO-LOGICAL-PAGE field forces a new SSI but continues the current LPGDEF values. See "EXP2 (Explicit Forms List)" on page 228 and "EOO-LOGICAL-PAGE" on page 236.

You *cannot* specify a new SSI through the Rulebase Library.

LPGDEF Command Format

Use the following control cards to define an LPGDEF command.

NOTE

Metacode printers do not allow more than one FRM per SSI; nor do they allow shifting an FRM. Therefore, do not specify an LPGDEF command for Metacode forms that have a FORMS= parameter in the DJDE.

DMGMERGE ignores an LPGDEF command defined for Inline forms or line printers.

LPGDEF

- **NAME=**
  The 1- to 32-character, user-defined LPGDEF name.
- **FSHIFT=**
  Optional.
  A value by which the logical pages on all front (odd) SSIs are moved horizontally (to the right) and vertically (down).

  This value can be used for such purposes as creating extra space on the left of the SSI for binding or printing on 3-hole paper. FSHIFT= applies to the entire SSI, including Documerge Overlays. Valid values are:

[horizontal shift]	Specify a numeric value followed by **I** (inches).
	This value can have two decimal places. For example:
	`FSHIFT=(0.25I 0I) -`
	This value can be negative. For example:
	`FSHIFT=(-0.25I 0I) -`

[vertical shift]	Specify a numeric value followed by **I** (inches).
	This value can have two decimal places. For example:
	`FSHIFT=(0I 0.25I) -`
	This value can be negative. For example:
	`FSHIFT=(0I -0.25I) -`

	NOTE: A numeric value without **I** specifies the absolute number of dots-per-inch. Because this value varies according to printer type, this format is not recommended.

The default FSHIFT= value is (**0I 0I**). (The SSI is not shifted.)

- **BSHIFT=**
  Optional.
  A value by which the logical pages on all back (even) SSIs are moved horizontally (to the right) and vertically (down).

  This value can be used for such purposes as creating extra space on the left of the SSI for binding or printing on 3-hole paper. BSHIFT= applies to the entire SSI, including Documerge Overlays. Valid values are:

[horizontal shift]	Specify a numeric value followed by **I** (inches).
	This value can have two decimal places. For example:
	`BSHIFT=(0.25I 0I) -`
	This value can be negative. For example:
	`BSHIFT=(-0.25I 0I) -`

[vertical shift]	Specify a numeric value followed by **I** (inches).
	This value can have two decimal places. For example:
	`BSHIFT=(0I 0.25I) -`
	This value can be negative. For example:
	`BSHIFT=(0I -0.25I) -`

	NOTE: A numeric value without **I** specifies the absolute number of dots-per-inch. Because this value varies according to printer type, this format is not recommended.

  The default BSHIFT= value is (**0I 0I**). (The SSI is not shifted.)

- **LPAGE=**

  Optional. You cannot use LPAGE= with FLPAGE= or BLPAGE=.

  Defines all logical pages for both front (odd) and back (even) SSIs. Each logical page is a group of the following four values:

[horizontal start]	The horizontal position (X coordinate) where the upper left corner of the logical page starts. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example:
	`LPAGE=(`**`0.00I`**` 0.00I 4.25I 5.5I) -`
	This value can be negative. For example:
	`LPAGE=(`**`-0.25I`**` 0.00I 4.25I 5.5I) -`

[vertical start]	The vertical position (Y coordinate) where the upper left corner of the logical page starts. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example:
	`LPAGE=(0.00I `**`0.00I`**` 4.25I 5.5I) -`
	This value can be negative. For example:
	`LPAGE=(0.00I `**`-0.25I`**` 4.25I 5.5I) -`

[horizontal size]	The width of the logical page. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example: `LPAGE=(0.00I 0.00I `**`4.25I`**` 5.5I) -`

[vertical size]	The length of the logical page. (Page size for Concatenation.) Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example: `LPAGE=(0.00I 0.00I 4.25I` **`5.5I`**`) -`
	NOTE: A numeric value without **I** specifies the absolute number of dots-per-inch. Because this value varies according to printer type, this format is not recommended.

Default LPAGE= values are listed in the following table. Defaults for horizontal and vertical size depend on the page orientation specified in the Structure Rule.

[horizontal start]	0
[vertical start]	0
[horizontal size]	If PORtrait: **LANDPGSZ** from the MERGEDEF If LANdscape: **PORTPGSZ** from the MERGEDEF
[vertical size]	If PORtrait: **PORTPGSZ** from the MERGEDEF If LANdscape: **LANDPGSZ** from the MERGEDEF

- **FLPAGE=**

Optional. You cannot use FLPAGE= with LPAGE=.
Defines all logical pages for front (odd) SSIs. Each logical page is a group of the following four values:

[horizontal start]	The horizontal position (X coordinate) where the upper left corner of the logical page starts. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example:
	`FLPAGE=(`**`0.00I`**` 0.00I 4.25I 5.5I) -`
	This value can be negative. For example:
	`FLPAGE=(`**`-0.25I`**` 0.00I 4.25I 5.5I) -`

[vertical start]	The vertical position (Y coordinate) where the upper left corner of the logical page starts. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example:
	`FLPAGE=(0.00I` **`0.00I`**` 4.25I 5.5I) -`
	This value can be negative. For example:
	`FLPAGE=(0.00I` **`-0.25I`**` 4.25I 5.5I) -`

[horizontal size]	The width of the logical page. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example: `FLPAGE=(0.00I 0.00I` **`4.25I`**` 5.5I) -`
[vertical size]	The length of the logical page. (Page size for Concatenation.) Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example: `FLPAGE=(0.00I 0.00I 4.25I` **`5.5I`**`) -`
	NOTE: A numeric value without **I** specifies the absolute number of dots-per-inch. Because this value varies according to printer type, this format is not recommended.

Default FPAGE= values are listed in the following table. Defaults for horizontal and vertical size depend on the page orientation specified in the Structure Rule.

[horizontal start]	0
[vertical start]	0
[horizontal size]	If PORtrait: **LANDPGSZ** from the MERGEDEF If LANdscape: **PORTPGSZ** from the MERGEDEF
[vertical size]	If PORtrait: **PORTPGSZ** from the MERGEDEF If LANdscape: **LANDPGSZ** from the MERGEDEF

- **BLPAGE=**
  Optional. You cannot use BLPAGE= with LPAGE=.
  Defines all logical pages for back (even) SSIs. Each logical page is a group of the following four values:

[horizontal start]	The horizontal position (X coordinate) where the upper left corner of the logical page starts. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example:
	`BLPAGE=(`**`0.00I`** `0.00I 4.25I 5.5I) -`
	This value can be negative. For example:
	`BLPAGE=(`**`-0.25I`** `0.00I 4.25I 5.5I) -`

[vertical start]	The vertical position (Y coordinate) where the upper left corner of the logical page starts. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example:
	`BLPAGE=(0.00I` **`0.00I`** `4.25I 5.5I) -`
	This value can be negative. For example:
	`BLPAGE=(0.00I` **`-0.25I`** `4.25I 5.5I) -`

[horizontal size]	The width of the logical page. Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example: `BLPAGE=(0.00I 0.00I` **`4.25I`** `5.5I) -`
[vertical size]	The length of the logical page. (Page size for Concatenation.) Specify a numeric value followed by **I** (inches). This value can have two decimal places. For example: `BLPAGE=(0.00I 0.00I 4.25I` **`5.5I`**`) -`
	NOTE: A numeric value without **I** specifies the absolute number of dots-per-inch. Because this value varies according to printer type, this format is not recommended.

Default BLPAGE= values are listed in the following table. Defaults for horizontal and vertical size depend on the page orientation specified in the Structure Rule.

[horizontal start]	0
[vertical start]	0
[horizontal size]	If PORtrait: **LANDPGSZ** from the MERGEDEF If LANdscape: **PORTPGSZ** from the MERGEDEF
[vertical size]	If PORtrait: **PORTPGSZ** from the MERGEDEF If LANdscape: **LANDPGSZ** from the MERGEDEF

STRUCTURE Command

The DMGRBMUT **STRUCTURE** command specifies the sequence in which the forms in a Merge Set are printed. This sequence is specified by the *position* of the DTN of each form in the sequence of DTNs coded in the Structure Rule. The sequence of DTNs is independent of the numeric values of the DTNs. For example, the following sequence of the DTNs (the form with a DTN of 10 will print last) , and the repetition of the same DTN (the same form prints again) is valid.

```
STRUCTURE -
         RULE=((50 SIM SEP POR ODD MAI STA) -
               (30 SIM SEP POR ODD MAI STA ESS) -
               (20 SIM SEP POR ODD MAI STA) -
               (40 SIM SEP POR ODD MAI STA) -
               (20 SIM SEP POR ODD MAI STA) -
               (10 SIM SEP POR ODD MAI STA))
```

You can also code Print Options (SIM, SEP, etc.) for each DTN in the Structure Rule.

STRUCTURE Command Format

This command defines a Structure Rule.

```
STRUCTURE -
     RULE=
```

The RULE= specifies the sequence of DTNs to print and the associated Print Options to execute when the STRUCTURE RULE is called for by a GROUP.

NOTE

If Print Options are not specified in the Structure Rule, defaults are assigned.

Document Type Number (DTN)

A Document Type Number (DTN) is a number assigned to each form that is an EDL member. The DTN controls the printing and collation of output. DTNs are part of the information stored with each EDL member. The Print Options and sequence in which EDL members are printed is determined by the Print Option assigned to the DTN and the placement of the DTN within the Structure Rule.

DMGRFMT places the EDL member name and Print Options in the VRF, based upon the placement of the DTN within the Structure Rule.

The following rules apply to DTNs:

- DTNs can range from 0 through 99999.
- An unlimited number of DTNs can be included per Structure Rule.
- The default DTN is 0.

NOTE

No cross-checking is performed to assure the form exists in the EDL. If the DTN cannot be found at the time DMGRFMT and DMGMERGE are executed, error messages are generated.

EDL forms with the same DTN are collated as follows:

(1) EDL implicit forms as defined in the FORMS table

(2) VDR-requested EDL explicit forms, in the order requested.

(3) Inline forms.

Factoring DTNs

When coding a Structure Rule, you use *factoring* as a short hand way to specify concatenation of multiple forms on the same page, and also to manipulate the Print Options assigned to the forms. The following Print Options support DTN factoring:

- CONcatenate — specifies that forms can be combined on a page if there is room, and that when coded using factoring, only the Print Options for the first DTN apply to all of the factored DTNs.

  To read detailed coding instructions for CON, see "CONcatenate" on page 173.

- CON**NP** (**N**o **P**ropagation of Print Options) — specifies that forms can be combined on a page if there is room, and that when coded using factoring, the Print Options specified for each factored DTN apply.

  To read detailed coding instructions for CONNP, see "CONNP" on page 173.

DTN Factoring Syntax for CON and CONNP

The syntax for factored DTNs with CON or CONNP requires a different placement of parentheses than does standard DTN syntax.

Standard DTN syntax example:

```
STRUCTURE -
     RULE=((10 CON)(20 CON))
```

Each DTN is placed on a separate page (if there is more than one form with the same DTN, these forms are placed on the same page if they fit).

Factored DTN syntax example:

```
STRUCTURE -
     RULE=((CON((10)(20))))
```

If they fit, both forms (DTNs 10 and 20) are placed on the same page.

CON-Factored DTNs Syntax Example

```
ADD STRUCTURE NAME=TST31J2(1)
STRUCTURE -
     RULE=((CON((10DUPOVL=('DON1'LOGICAL)-
BVL=('JIM1'LOGICAL))-
(20)-
(30)-
(40))))
```

CON concatenates the factored DTNs 10, 20, 30 and 40 on the same page and applies the print options for the first DTN to all other DTNs in the statement. You can see the results in the output of the program that executes the CON command, the DMGRBMUT Rulebase program:

```
Structure Rule: TST31J2(1)
Concatenation Set
DTNOptions
10DUP CON STA ANY POR OVL
OVL='DON1(MERGE)'LOGICAL)
BVL='JIM1(MERGE)'LOGICAL)
20DUP CON STA ANY POR OVL
OVL='DON1(MERGE)'LOGICAL)
BVL='JIM1(MERGE)'LOGICAL)
30DUPCONSTAANYPOROVL
OVL='DON1(MERGE)'LOGICAL)
BVL='JIM1(MERGE)'LOGICAL)
40DUP CON STA ANY POR OVL
OVL='DON1(MERGE)'LOGICAL)
BVL='JIM1(MERGE)'LOGICAL)
end of Concatenation Set
```

When you use CON, the Print Options for the first DTN are applied to all the other factored DTNs.

CONNP-Factored DTNs Syntax Example

```
ADD STRUCTURE NAME=TST31J(1)
STRUCTURE -
     RULE=((CONNP((10DUPOVL=('DON1'LOGICAL)-
BVL=('JIM1'LOGICAL))-
(20)-
(30)-
(40))))
```

CONNP concatenates the factored DTNs 10, 20, 30, and 40 on the same page, and changes the input page parity and Overlays for DTN 20. You can see the results in the output of the program that executes the CONNP command, the DMGRBMUT Rulebase program:

```
Structure Rule: TST31J(1)
Concatenation Set
DTNOptions
10DUP CON STA ANY POR OVL
OVL='DON1(MERGE)'LOGICAL)
BVL='JIM1(MERGE)'LOGICAL)
20SIM CON STA ANY POR
30SIM CON STA ANY POR
40SIM CON STA ANY POR
end of Concatenation Set
```

When you use CONNP instead of CON, the Print Options can change from one DTN to the next.

NOTE

When using CONNP, if a Print Option is not given for a DTN, the default Print Option value for that DTN applies.

Coding a Structure Rule for an Imposition Definition

There are two variations when coding the STRUCTURE command for an Impostion Definition:

- You can code the Imposition Definition as an internal table to the Structure

Structure Rule with an Internal Imposition Definition Example

```
STRUCTURE RULE=( (DTN xxx xxx xxx) DTN (DTN xxx) -
                    (DTN xxx IMPDEF ...see IMPDEF command for syntax) -
                     DTN (CON((DTN xxx) DTN)) DTN )
```

- You can code an external IMPDEF Table that contains the definition of the IMPDEF for this Structure.

Structure Rule with an External Imposition Definition Example

```
STRUCTURE RULE=( (DTN xxx xxx xxx) DTN (DTN xxx) -
                    (DTN xxx IMPDEF=tablename(revisionlevel) – -
                     DTN (CON((DTN xxx) DTN)) DTN )
```

Coding a Structure Rule for Expanded Overlay Support

Documerge now offers expanded Overlay DTN-level options that let you

- Define an Overlay for the first page of a Document Package, and a different Overlay for all succeeding pages. For example, you could print CONTINUED FROM PREVIOUS PAGE on all but the first page of a Document Package.
- Define an Overlay for the last page of a Document Package, and a different Overlay for all preceding pages. For example, you could print CONTINUED ON THE NEXT PAGE on all but the last page of a Document Package.
- Place Overlays on logical pages as well as physical pages.
- Use bar codes on imposed forms.

Also with Documerge 3.1 and later versions, you can specify *General* or Document Package-level Overlays in the FILEDEF, GLOBAL, or MERGE commands.

For a more detailed explanation of Documerge Overlay options, see "Documerge Overlay Forms" on page 465.

Controlling Widows / Orphans with the KEEPLAST Option for Concatenated Forms

Documerge users can concatenate many one line forms to dynamically build a Document Package. When employing this technique, a user often wants to avoid printing the last form-line on a page by itself.

When an undesirable isolated line occurs, it is called an *orphan* or *widow*. However, controlling widows/orphans with program code that inserts blank forms to force related forms to stay together can result in complex, high-maintenance VDRs.

You can use the KEEPLAST parameter of the CONcatenate Print Option to specify the

- Printing on the same page of two or more forms with the same DTN or factored DTNs, without referring to the form names.

  For example, the following Structure rule specifies that the last 2 forms with the same DTN print together so that their related text is not placed on separate pages.

```
ADD STRUCTURE NAME=S141
STRUCTURE -
     RULE=((10 SIM SEP POR ODD MAI STA) -
           (20 SIM CON POR ODD MAI STA) -
           (30 SIM CON POR ODD MAI STA KEEPLAST=2))
```

- Printing of two or more forms with different DTNs on the same page.

For example, the following Structure rule uses a factored KEEPLAST option and factored concatenation to specify that the last two concatenated forms print together — whatever the last two forms are for DTNs 10, 20, and 30.

```
ADD STRUCTURE NAME=S141
STRUCTURE -
     RULE=( CON KEEPLAST=2
                      ((10 SIM POR ANY MAI STA) -
                       (20 SIM POR ANY MAI STA) -
                       (30 SIM POR ANY MAI STA)))
```

- Printing of all the forms with the same DTN on the same page.

  For example, the following Structure rule uses factored concatenation to specify that all the forms for DTN 30 be kept together:

```
ADD STRUCTURE NAME=S141
STRUCTURE -
            RULE=( CON     ((10 SIM POR ANY MAI STA — -
                              (20 SIM POR ANY MAI STA — -
                              (30 SIM POR ANY MAI STA KEEPLAST=ALL)))
```

For details about the KEEPLAST option, see "Print Option Parameters" on page 172.

Print Options

For a DTN, you can code the following types of Print Options.

- Orientation options (LANdscape, PORtrait) specify printing to occur parallel to the shorter (portrait) or longer (landscape) edge of the sheet.
- Print Layout options (DUPlex, ENDSUBset/ESS, IMPDEF=, LPGDEF=, SIMplex, TUMble) control how the DTN prints for each sheet face or how the DTN prints in relation to the top and bottom edges of the sheet or how the DTN prints in the sequence of sheets in a Document Package.
- Page Parity options (ANY, BACk, EVeN, FROnt, ODD) specify on which side of the sheet to start printing a form.
- Concatenation options (SEParate, CONcatenate) control whether or not more than one form can be printed on a page.
- Overlay options (BVL, FVL, OVL) control how one or more other forms can appear on the same page with a form represented by a DTN.
- Printer control options (PSF COPYGROUPs).
- Output Stacking options (OFFset, STAck) control how printed pages are placed in the printer's output bin.
- Paper Bin options (AUXiliary, CLUSTER=, FEED=, MAIn, PRV, SRC) specify the paper tray that will contain the required paper stock for the DTN.

Print Options by Print Data Stream Type

The following table lists the Print Options available for the AFP, Metacode, and line printer data streams supported by Documerge. The Print Options are listed in the order in which they appear in the DMG.OPT.*groupname* Reserved Tag in the VRF.

AFP	Metacode	Line Printer
SIM/DUP/TUM/IMP	SIM/DUP/TUM/IMP	SIM/DUP
SEP/CON	SEP/CON	SEP/CON
MAI/AUX	MAI/AUX/FEED/SRC/PRV	MAI
STA/OFF	STA/OFF	STA
ANY/ODD/EVN/FRO/BAC	ANY/ODD/EVN/FRO/BAC	ANY/ODD/EVN/FRO/BAC
POR/LAN	POR/LAN	POR
OVL/FVL/BVL	OVL/FVL/BVL	OVL
US1 (PSF COPYGROUP)		-----
US2 (PSF COPYGROUP)		-----
US3 (PSF COPYGROUP)		-----
US4 (PSF COPYGROUP)		-----
US5 (PSF COPYGROUP)		-----
ESS	ESS	ESS
LPG	LPG	-----

Print Option-Related Processing Functions by Print Data Stream Type

Use the following table to determine if you can apply a particular Documerge processing function to the AFP, Metacode, or line printer data stream.

Function Description	Print Data Stream Type		
	Metacode	AFP	Line Printer
Apply Imposition to EDL forms	Applicable	Applicable	NA
Apply Imposition to inline forms	NA	NA	NA
Apply Overlay to EDL form	Applicable	Applicable	Applicable in part — Overlays and other forms are always shifted downwards, and overprinting doesn't occur. All printing is simplex so no *back* Overlays or *when* options apply
Apply Overlay to inline form	Applicable, but an inline form cannot be shifted, whereas an EDL-based form is shifted down by the size of the largest Overlay	Not applicable (NA) because Overlay forms are in page mode, and AFP doesn't allow mixing of page mode and line mode (inline) forms on the same physical page	Applicable, an inline form is shifted down just like an EDL form because inline Documerge processes inline and EDL forms identically
Apply Sheet Overlays	Applicable	Applicable	NA
Apply Subsets (DMG.END.OF.SUBSET tag)	Applicable	Applicable	Applicable
Apply US1–US5 Print Options	Applicable, but typically not used	Applicable, but dynamic Copy Groups may be more useful	NA
Change Common Fonts dynamically	Applicable using COMMON FONTS command in DMGMERGE, or the DMGCMFN program	NA (Common Fonts list not required)	NA
Concatenate EDL forms	Applicable	Applicable	NA
Concatenate inline forms	NA	NA	Applicable, because inline Documerge processes inline and EDL forms identically
Create Logical Pages using EDL forms	Applicable	Applicable	NA
Create Logical Pages using inline forms	NA	NA	NA

Function Description	Print Data Stream Type		
	Metacode	AFP	Line Printer
Offset stack output	Applicable	Applicable	NA
Print duplex	Applicable	Applicable	NA
Process pre-tumbled forms	Applicable, but the GLOBAL FLIPINVF parameter must be set, each BPSD must use unique replacement characters, Overlays need to have BOTTOM=0 to avoid shifting; and Concatenation, Imposition, and Logical pages cannot be applied **TIP:** Oracle recommends not pre-tumbling forms, but letting Documerge do the tumbling using the INVERTDJDE parameter. For details, see "INVERTDJDE=" on page 101.	NA (AFP fonts tumble automatically)	NA
Select input tray	Applicable	Applicable	NA
Set Logical Page Bottom (only available for Normalized forms)	Applicable	Applicable	NA (inline forms are not Normalized and the line printer mode doesn't support vertical reversal of the printing position on a page)
Set Page Orientation (landscape)	Applicable	Applicable	NA
Set Page Parity	Applicable	Applicable	NA
Substitute fonts (DMG.FONT. SUBSTITUTION... tags)	Applicable	Applicable	NA
Tumble an inline form	Applicable only if INVERTDJDE=Y is set in the PRINTDEF	Applicable	NA
Tumble an EDL form	Applicable	Applicable	NA

Print Option Parameters

Following is a comprehensive explanation of the Print Options which can be used with Documerge. You typically specify one or more Print Options for each DTN in a Structure Rule.

Some options are shown in mixed case. You can abbreviate these options, using the leftmost uppercase characters as the minimum characters for the abbreviation. You can add any of an option's lowercase characters to the abbreviation, as long as their left-to-right sequence is not altered.

Valid abbreviations are shown in uppercase in the following table.

Parameter	Value
ANY	Indicates that the first form of the specified DTN starts on the next available side of a page (this could be the odd or even side). Available with all Documerge-supported printers that support duplex printing. ANY is the default.
AUXiliary	Indicates the paper tray defined as auxiliary to the printer to be used for the form. Available to all printers supported by Documerge with auxiliary tray processing.
	NOTE: AFP applications require that the COPYGROUP name for auxiliary processing be defined in the PRINTDEF with the parameter of AUXFEED= parameter.
BACk	Indicates that the first page of *each* form of the specified DTN starts on the even side of the paper.
BVL=( )	**Back Overlay**. The form coded in the parentheses is printed on the *even* (back) side of each form specified for the DTN. Available for Documerge-supported duplex printers. Not applicable for Inline (line-printer) forms. For detailed rules and coding examples, see "xVL=( )" on page 178. For an explanation of xVL options, see "Coding a Structure Rule for Expanded Overlay Support" on page 167. For further information about Overlays and Overlay coding options, see "Documerge Overlay Forms" on page 465.
CLUSTER=	Indicates the name or bin number assigned to a paper tray as defined to the printer. Available for Documerge-supported Metacode printers with more then two paper trays. This command is interchangeable with the FEED= command, both of which generate the SRC Option Field in the DMG.OPT.*groupname* Reserved Tag.
	NOTE: When using this command with Metacode printers, verify that cluster name is defined in the stock set in the JDE being used.

Parameter	Value

CONcatenate	Specifies that forms can be combined on a page if there is room.
	Standard syntax example:
	`RULE=((10 CON)(20 CON))`
	You can **factor** CONcatenate across several DTNs. Factoring allows forms with different DTNs to acquire the same Print Options and be placed on the same page. See "Factoring DTNs" on page 165 for more information.
	Factored syntax example:
	`RULE=((CON((10)(20))))`
	When concatenation is factored across DTNs using the Print Option CON (or CONCATENATE), it forces all DTNs to use the Print Options from the first DTN; while Print Options coded for DTNs other than the first are ignored.
	For example:
	`RULE=((CON ((10 FVL=FRONT.OVERLAY.FORM) (20) (30))))`
	DTNs 20 and 30 will also use FVL=FRONT.OVERLAY.FORM because all Print Options from the first DTN (10 in this example) are propegated to the other DTNs (20 and 30 in this example).
	For a detailed explanation of the output for CON, see "Factoring DTNs" on page 165.

	IMPORTANT: When CONcatenate is used with Metacode forms, these rules apply: ■ You must use a Common Font List. ■ If you use LANdscape also, you must set the DMGMETP control card BOTTOM= equal to the actual size of the form *including the last line*, or specify a larger LANdscape value. A BOTTOM= value that does not include the last line (the artificial bottom) can cause overprinting or an unprintable page. See "BOTTOM=" on page 58.
CONNP	Alternatively to CON, you can use CONNP (**CON**catenation with **N**o **P**ropagation) to request that Print Options **not be propegated** accross DTNs. For example: `RULE=((CONNP ((10 FVL=FRONT.OVERLAY.ONE) - (20 FVL=FRONT.OVERLAY.TWO) - (30))))` DTN 10 gets FRONT.OVERLAY.ONE, DTN 20 gets FRONT.OVERLAY.TWO, and DTN 30 gets no Overlay. **Note that if a DTN 20 form is concatenated to a DTN 10 form, then FRONT.OVERLAY.ONE is still in effect.** Overlays do not change until pages change. Therefore, FRONT.OVERLAY.TWO prints only when the first form on the front (odd) side is DTN 20. **Note that other Print Options also might not take effect until either a page or sheet changes (for example, input trays or offset stacking).**
	CAUTION: When using CONNP, if you don't specify Print Options for a DTN, then the default Print Options apply. Oracle recomends that you specify all Print Options for each DTN When using CONNP.
	For a detailed explanation of the output for CONNP, see "Factoring DTNs" on page 165.
DUPlex	Both sides of the physical page are available for printing. Available to all printers supported by Documerge where duplex printing is available.

Parameter	Value

ESS
or
ENDSUBset
	Defines a portion, or a subset, of a Document Package. This option is for all Documerge forms, but it is used most often for Overlays that are printed on all the forms in a range of DTNs.
	You use the DMG.END.OF.SUBSET Reserved Tag with this option. The DMG.END.OF.SUBSET tag indicates whether a dashcode is inserted in a form. Some finishing equipment, such as Bell & Howell AIM equipment, can read the dashcode to verify that all forms in the subset have been processed. (See "DMG.ERDD.*Groupname*" on page 306 for more information.)
	For example, suppose you create the following Structure Rule, specifying the ESS Print Option for DTN 30:
	`ADD STRUCTURE NAME=S142`
	`STRUCTURE -`
	`RULE=((10 SIM SEP POR ODD MAI STA) -`
	`(20 SIM SEP POR ODD MAI STA) -`
	`(30 SIM SEP POR ODD MAI STA ESS) -`
	`(40 SIM SEP POR ODD MAI STA) -`
	`(50 SIM SEP POR ODD MAI STA))`
	In this example, the first subset consists of the forms in DTNs 10, 20, and 30. DMGMERGE places a dashcode on the form that contains the last DMG.END.OF.SUBSET tag in this subset. (This tag receives the DASHCODE-ON value defined in the MERGEDEF. All previous DMG.END.OF.SUBSET tags receive the DASHCODE-OFF value.)
	The second subset consists of the forms in DTNs 40 and 50. Although you can specify the ESS Print Option for DTN 50, DMGMERGE automatically designates the last DTN of a Structure Rule as the end of the last subset.
	You can specify the ESS option for a DTN that does not contain forms. The maximum number of ESS options per Structure Rule is 100.

	TECHNICAL NOTE: Although you specify ESS in the Structure Rule, the DMGRFMT sub-program generates BSS (BeginSubSet) and CSS (ContinueSubSet) in the VRF. In the preceding example, DMGRFMT places BSS in the DMG.OPT.*groupname* Reserved Tag for the first form of DTN 10. Any other forms of DTN 10, as well as all forms of DTN 20 and DTN 30, receive CSS. The first form of DTN 40 receives BSS; all remaining forms in the example receive CSS. You use BSS and CSS to override an ESS Print Option in the VDR. See "EXP2 (Explicit Forms List)" on page 228 for more information.
EVeN	Indicates that only the *first page of the first form* in the specified DTN starts on the even side of the paper. Available with all Documerge-supported printers that support duplex printing.
FEED=	Indicates the name or bin number assigned to a paper tray as defned to the printer. Available for Documerge-supported Metacode printers with more than two paper trays. This command is intercahngeable with the CLUSTER= command, both of which generate the SRC Option Field in the DMG.OPT.*groupname* Reserved Tag.
	NOTE: When using this command with Metacode printers verify that cluster name is defined in the stock set in the JDE being used.
FROnt	Indicates that the first page of *each* form in the specified DTN starts on the odd side of the paper.

Parameter	Value
FVL=( )	**Front Overlay.** The form coded in the parentheses is printed on the *odd* (front) side of each form specified for the DTN. Available for all Documerge-supported printers. For detailed rules and coding examples, see "xVL=( )" on page 178. For an explanation of xVL options, see "Coding a Structure Rule for Expanded Overlay Support" on page 167. For further information about Overlays and Overlay coding options, see "Documerge Overlay Forms" on page 465.
IMPDEF	Specifies the definition of an internal Imposition Definition for a Structure. See "IMPDEF Command" on page 151 for the syntax and definition of its parameters.

IMPDEF= tablename (revisionlevel)	References an external IMPDEF Table that contains the specifications for the Imposition Definition for this Structure Rule. The referenced external IMPDEF Table is maintained separately from this STRUCTURE and it can be shared between several Structures. Indicates that the forms are to be printed according to the rules contained within the named Imposition Definition.
	Standard syntax example:
	`RULE=((10 IMPDEF=XXXXX)(20 IMPDEF=XXXXX))`
	IMPDEF= can be factored across DTNs. Factoring allows forms with different DTNs to be placed on the same page. See "Factoring DTNs" on page 165 for more information.
	Factored syntax example:
	`RULE=((IMPDEF=XXXXX((10)(20))))`

KEEPLAST=	(Optional) directs Documerge to try to keep the specified number of forms for concatenation together, or to print the specified number of forms together on a page (if the forms will fit).
	For further explanation, see "Controlling Widows / Orphans with the KEEPLAST Option for Concatenated Forms" on page 167.
	Formats: `KEEPLAST=nnnnn`
	`KEEPLAST=ALL`
	`nnnnn` specifies the number of forms in a the list of DTNs to keep on a page. `nnnnn` can be 1 to 5 digits with leading zeros (not required). The minimum value is 1 and the maximum value is 32767. `KEEPLAST=1` is the default, which means *No KEEPLAST*.
	`ALL` specifies that Documerge is to try to print on the page, every form specified for a DTN.
	For example, the following Structure rule uses a factored *KEEPLAST* option and factored concatenation to specify that the last two concatenated forms print together, whatever the last two forms are for DTNs 10, 20, and 30.
	`STRUCTURE -`
	`RULE=( CON KEEPLAST=2`
	`((10 SIM POR ANY MAI STA) -`
	`(20 SIM POR ANY MAI STA) -`
	`(30 SIM POR ANY MAI STA)))`

	TIP: You can use the GLOBAL parameter, CKPERROR to generate a W-level error message when a form specified for CKP (concatenate and keep with previous form if possible) does not have space to print on the page with the previous form. For details, see "CKPERROR=" on page 388.

Parameter	Value

	CAUTION: When coding the KEEPLAST parameter in a factored concatenated DTN set, the KEEPLAST must occur on the last DTN of the set. The following code is invalid.
	`ADD STRUCTURE NAME=S141`
	`STRUCTURE -`
	`RULE=( CON ((10 SIM POR ANY MAI STA KEEPLAST=ALL) -`
	`(20 SIM POR ANY MAI STA KEEPLAST=ALL) -`
	`(30 SIM POR ANY MAI STA KEEPLAST=ALL)))`
	The KEEPLAST cannot be coded at the factored level and the DTN level of the STRUCTURE RULE. The following code is invalid.
	`ADD STRUCTURE NAME=BADKEEP`
	`STRUCTURE-`
	`RULE=((CON KEEPLAST=3((50 DUP POR ODD STA MAI)-`
	`(45 DUP POR ODD STA MAI KEEPLAST=8)-`
	`(43 DUP POR ODD STA MAI))))`

LANdscape	Indicates the orientation of a page where the longer edge of the page is at the top and bottom. Typically defined in the PrintDef.

LPGDEF=	Identifies the name of the LPGDEF command that you want to assign to the forms in a DTN. See "LPGDEF Command" on page 154 for more information.
	Standard syntax example:
	`RULE=((10 LPGDEF=MSGLPG2)(20 LPGDEF=MSGLPG2))`
	LPGDEF= can be factored across DTNs. Factoring allows forms with different DTNs to be included on the same page. See "Factoring DTNs" on page 165 for more information.
	Factored syntax example:
	`RULE=((LPGDEF=MSLPG2((10)(20))))`

MAIn=	Indicates the paper tray defined as main to the printer is to be used for the form. Main is the default. Available to all printers supported by Documerge.
ODD	Indicates that only the first page of the first form in the specified DTN starts on the odd side of the paper. Available to all printers supported by Documerge.
OFFset	Indicates the printer output will be offset from the last stack. Available for all printers supported by Documerge which have offset capability.
	NOTE: Documerge requires that the stack offset for Metacode be defined using the PRINTDEF parameters ROFFFLAG and ROFFOFF. Documerge requires that the stack offset for AFP applications be defined using the printer offset Copy Group in the PRINTDEF's PAPEROFFSET parameter.
OVL=( )	**Overlay.** The form coded in the parentheses is printed on *both sides* of each form specified for the DTN. For detailed rules and coding examples, see "xVL=( )" on page 178. For an explanation of xVL options, see "Coding a Structure Rule for Expanded Overlay Support" on page 167. For further information about Overlays and Overlay coding options, see "Documerge Overlay Forms" on page 465.
PORtrait	Indicates the orientation on a page, where the shorter edge of the page is at the top and bottom. Typically defined in the PrintDef.
SEParate	Indicates that each form begins on a new page. SEParate is the default. CONcatenate is the override. Available to all printers supported by Documerge.
SIMplex	One side of the physical page is printed. If the printer is in duplex mode, a blank page is generated. Blank pages are counted and indicated in the Reserved Tags DMG.PAGE.NUMBER and DMG.TOTAL.PAGES, but are not counted in DMG.SSI.COUNT. Simplex is the default; Duplex, Tumble and Imposition are alternate values. Available to all printers supported by Documerge.

Parameter	Value
STAck	Indicates the printer output will be stacked. Stack is the default; Offset is the override. Available for all printers supported by Documerge.
TUMble	Both sides of the physical page are available for printing head to toe (used for Document Packages to be bound across the top of the page). Available to Metacode printers only. Available for AFP printers with the use of a FORMDEF COPYGROUP as specified in form or by the Print option tag USER#.
	CAUTION!: Forms containing graphics, .IMGs, page segments, or Overlays can not be tumbled by Print Options. When tumble is used for Metacode forms: ■ a Common Font List is required ■ the hex 08 Metacode used for drawing lines and shading can only be specified for a Documerge graphic font such as FORMSX, FORMS$, or ISISPX
USer1 to USer5	Indicates that one of 5 Print Options defined in the PRINTDEF as COPYMOD is to be used. See "Using COPYGROUP Print Options in an Older Documerge AFP Environment" on page 182 for more information.

Parameter	Value
xVL=( )	The Overlay coded in the parentheses is printed on *the front (FVL), or back (BVL), or front and back (OVL) of the* form specified for the DTN. For an explanation of xVL options, see "Coding a Structure Rule for Expanded Overlay Support" on page 167. For further information about Overlays and Overlay coding options, see "Documerge Overlay Forms" on page 465. Format: *x*VL=(*'member name'(revision) when appliesto where*) Replace the *x* placeholder with a value that indicates the type of Overlay you want: ▪ BVL — back side only ▪ FVL — front side only ▪ 0VL — both front and back side The outside parentheses are required; if omitting the revision, omit the inner parentheses. Valid values for the placeholders inside the parentheses are: **member name** (Required) the 1- to 32-character name of the form used as an Overlay. If there are blanks, parentheses, or commas in the member name, the delimiting single quotes are required, otherwise they are optional but recommended. **(revision)** (Optional) a 1- to 5-character revision number or identifier delimited with parentheses. Valid values are: **0** The highest revision level at the time DMGMERGE is executed. The default is **0**. **n** A specific revision level. The value for **n** can range from 1 to 32,767. **VDR** The highest revision level at the time the VDR is executed. **MERGE**The highest revision level at the time DMGMERGE is executed. **when** (Optional) specifies the printing an Overlay based on whether or not the page contains other forms. The **when** values are mutually exclusive; code at most one value per Overlay. **when** does not apply for logical page Overlays. For example: `0VL=('FIRSTSHEET' ONLYBLANK)` Specifies that an Overlay named `FIRSTSHEET` is to print *when* a page is a blank (`ONLYBLANK`) page. Valid values for **when** are: **DEFAULT** Use the default specified in the `GLOBAL FVLDEFAULT`, `BVLDEFAULT`, or `0VLDEFAULT` parameters. **ALWAYS**The Overlay prints on every page. `ALWAYS` is the default. **NONBLANK** The Overlay prints only if a non-Overlay form is printed on the page. **ONLYBLANK** The Overlay prints only if no other non-Overlay form is printed on the page.

Parameter	Value	
	appliesto	(Optional) specifies the type of page on which to print an Overlay. The **appliesto** values are mutually exclusive; code only one value per statement. For example: `OVL=('FIRSTSHEET' ONLYBLANK PHYSICAL)` Specifies that an Overlay named `FIRSTSHEET` is to print *when* there is a side of a blank (`ONLYBLANK`) page that *appliesto* the physical page.
	NOTE: For an explanation of **appliesto** and how it relates to other Documerge Overlay features, see "Specifying the Page Type for an Overlay — the APPLIESTO Option" on page 467.	
		Valid values for **appliesto** are: **LOGICAL**Prints the Overlay on each logical page (or physical page if no logical pages are defined).
	NOTE: The Overlay will print for all logical pages defined for the DTN because any front (FVL) or back (BVL) Overlays refer to the physical page.	
		PHYSICALPrints the Overlay on the physical page. `PHYSICAL` is the default currently used by Documerge. **SHEET** Prints the Overlay on one side of the sheet of paper.
	TIP: When using the Routing-By-Sheets (MAXSHEETS=) feature, and you want different Overlays for different files, use the SHEET option. Documerge applies SHEET overlays after it has counted the number of sheets and the final output data set has been determined.	
	where	(Optional) specifies the printing location of an Overlay in an Overlay set. The **where** values are mutually exclusive; code only one value per Overlay. **where** does not apply to inline forms or logical overlays. For example: `OVL=('FIRSTSHEET' ONLYBLANK FIRST)` Specifies that an Overlay named `FIRSTSHEET` is to print *when* there is a side of a blank (`ONLYBLANK`) page *where* it comes `FIRST` in a Document Package.
	NOTE: For an explanation of **where** and how it relates to other Documerge Overlay features, see "Specifying the Printing Location of an Overlay — the WHERE Option" on page 468.	
		Valid values for **where** are: **FIRST** Prints the Overlay only on the first page of the Overlay set. Complies with other options like **when** and page parity. **NOTFIRST** Prints the Overlay on all pages except the first, complying with other Overlay options such as **when** and page parity. **LAST** Prints the Overlay only on the last page of the Document Package. Complies with other options like **when** and page parity. **NOTLAST** Prints the Overlay on all pages except the last page of the Document Package, complying with other Overlay options such as **when** and page parity. **MIDDLE** Prints the Overlay on all pages except the first and last pages. Complies with other options like **when** and page parity.

Parameter	Value
	ALL (Default) prints the Overlay on all pages, except the banner and trailer pages, complying with other Overlay options such as **when** and page parity.
	TIP: The DMGMETP and DMGAFPP BOTTOM= parameter controls where the Overlay ends. (The BOTTOM= value is sometimes called the "artificial bottom.") If you do not specify a BOTTOM= value, DMGMERGE assumes that the Overlay is a running header; DMGMERGE prints the other Boilerplate after the last line of the Overlay. If you want the other Boilerplate printed actually on top of the Overlay, code BOTTOM=0 when you normalize the Overlay.

Using Dynamic COPYGROUPs in an AFP Environment

DMGMERGE can create COPYGROUP names dynamically, using them to perform many AFP printing functions automatically. (Dynamic COPYGROUP names are required for automatic invocation of functions such as Imposition and Tumble printing.)

You define these dynamic COPYGROUPs as you do any other COPYGROUP — in the AFP FORMDEF. Oracle recommends that you place dynamic COPYGROUPs in a separate FORMDEF that you use only for Documerge dynamic COPYGROUP names.

Oracle supplies PPFA source code on the Documerge 3.x installation tape. You can run this source code to create dynamic COPYGROUPs for AFP Imposition printing. Refer to *Installing Documerge* for more information.

DMGMERGE generates a 3-character name for the COPYGROUP according to the Print Options you assign in the Structure Rule. The DMGRFMT sub-program places these Print Options in the VRF, in the DMG.OPT.*groupname* tag.

The Print Options that relate to the dynamic COPYGROUP name are those available for Option Fields 1, 3, and 4 in the DMG.OPT.*groupname* Reserved Tag. (See "DMG.OPT.Groupname" on page 317 for more information.)

The following VRF example shows the Option Fields in the DMG.OPT.*groupname* Reserved Tag for one DTN in the Group named 9700.INSURED.

```
CHAR  DMG.OPT.9700.INSURED {DUP SEP MAI STA ODD POR OVL
ZONE 1CDC4DDE4FFFF4CDEEDCC0CCED4ECD4DCC4EEC4DCC4DDD4DED
NUMR 4447B673B9700B952495400447025704190231064407690653
    01...5...10...15...20...25...30...35...40...45...50
```

DMGMERGE copies the first character in Option Fields 1, 3, and 4 to the first, second, and third positions respectively in the dynamic COPYGROUP name. The remaining 3 positions are blank.

The GLOBAL parameter DYNAMICCG= indicates whether DMGMERGE uses the dynamic COPYGROUP and ignores:

- The default COPYGROUP
- A COPYGROUP specified by a USERx option
- A COPYGROUP that exists in a form.

See "DYNAMICCG=" on page 389.

The following table lists the DMG.OPT.*groupname* Option Field numbers, their related Print Options, and the characters for each position in the dynamic COPYGROUP name. Also listed for each Print Option are the dynamic COPYGROUP names that are possible when the Print Option is entered in the Structure Rule.

Option Field	Print Option	Position in Name			COPYGROUP Names			For More Information
		1	2	3				
1	**D**UP	D			DMS DMO	DAS DAO		"DUPlex" on page 173
1	**S**IM	S			SMS SMO	SAS SAO		"SIMplex" on page 176
1	**T**UM	T			TMS TMO	TAS TAO		"TUMble" on page 177
3	**M**AI		M		DMS DMO	SMS SMO	TMS TMO	"MAIn=" on page 176
3	**A**UX		A		DAS DAO	SAS SAO	TAS TAO	"AUXiliary" on page 172
4	**S**TA			S	DMS DAS	SMS SAS	TMS TAS	"STAck" on page 177
4	**O**FF			O	DMO DAO	SMO SAO	TMO TAO	"OFFset" on page 176

NOTE

Whether a COPYGROUP is generated dynamically or from a USERx option, DMGMERGE **cannot** verify that the COPYGROUP exists in the FORMDEF. Therefore, the COPYGROUP name in the FORMDEF must match the COPYGROUP name. You must ensure that there are no PSF errors.

Users of Documerge 3.1.2 and later versions can use dynamic COPYGROUPS to set up multiple paper trays, as described in the following procedure.

To Set Up COPYGROUPs for More than Two AFP Input Trays

1. Code the `DYNAMICCG=YES` parameter for the GLOBAL command. For details, see "DYNAMICCG=" on page 389.
2. In a Rulebase, code the `FEED=` parameter to specify the desired input tray names of one-to-six characters each. For details, see "FEED=" on page 174.

   In Documerge, tray names are also referred to as *cluster* or *feed* names. For AFP, the last non-blank character of the tray name must be unique because it is used to generate the dynamic COPYGROUP (IMM) name.

   For example, you might choose `TRAY1`, `TRAY2`, and `TRAY3` to represent trays 1, 2, and 3. Documerge only uses the last character — 1, 2, and 3 in this case.
3. Define all usable dynamic COPYGROUPs to AFP/PSF using the IBM PPFA AKQPPFA utility program.

With the GLOBAL parameter DYNAMICCG= set to YES, DMGMERGE will generate a three-character COPYGROUP (IMM) name, as follows:

- Character 1 — print mode — D (duplex), S (simplex) or T (tumble)
- Character 2 — Input tray selection — M (main tray), A (aux tray), x — Rulebase FEED=namex, where x is the placeholder for the last non-blank character.
- Character 3 — offset stack/jog option — S (no offset stack/jog), O (offset stack/jog — alpha O, not zero).

TIP

Using DYNAMICCG=YES, you can prepare the same FEED names for both Metacode and AFP printers, provided that the last non-blank character is unique (for AFP), and that the Metacode printer is properly set up for the desired feed/cluster names. This allows use of the same VRF to generate both AFP and Metacode data streams.

Using COPYGROUP Print Options in an Older Documerge AFP Environment

NOTE

The following two pages discuss a method of defining and selecting COPYGROUPs that is used for Documerge versions 1 and 2. These previous versions employ the USER1 through USER5 Print Options.

Documerge 3.x offers a way to create COPYGROUPs dynamically. We encourage AFP users to create dynamic COPYGROUPs rather than use the method in this discussion. See "Using Dynamic COPYGROUPs in an AFP Environment" on page 180.

IBM AFP environments rely on Print Services Facility (PSF) to control the print job. PSF associates one AFP FORMDEF with each print job to define certain characteristics, such as page margins and paper size, about the output pages.

Within the FORMDEF, you can define one or more COPYGROUPs. These COPYGROUPs supplement the FORMDEF with more specific information about the print job, such as input tray selection and Tumble or Offset printing.

IMPORTANT!

Oracle recommends that no COPYGROUP contain an Overlay option, and that instead you use the Insert Page Overlay (IPO) structure field command in your composition system.

This condition allows Documerge to perform most effectively the printing functions that require multiple forms to be placed on the same page, such as Concatenation, Imposition, Documerge Overlays, and Logical Page Definition.

When the AFP data stream references a COPYGROUP, PSF invokes the printer functions specified in that COPYGROUP. These printer functions remain in effect until the data stream references a different COPYGROUP.

Documerge enables you to reference COPYGROUPs to perform your processing requirements. The following steps must be performed to invoke this processing.

To Set Up COPYGROUPs in an Older Documerge Processing Environment

1. Define one COPYGROUP for each group of printer functions you perform.

   Since only one COPYGROUP can be active at a single time, you must create one COPYGROUP to invoke all of the processing functions you want to perform at that time.

   For example, if you want to perform both Tumble and auxiliary feed processing, you must create one COPYGROUP that invokes both printer functions.

   You use the COPYMODx parameter in the PRINTDEF to define COPYGROUPs. You can define up to five of these unique COPYGROUPs.

2. Define one default COPYGROUP, using the DEFAULTCG parameter in the PRINTDEF. This COPYGROUP should also be defined as the first COPYGROUP in the FORMDEF.

   You use this default COPYGROUP to discontinue an active COPYGROUP by invoking new processing options.

   For example, if you have invoked the COPYGROUP that produces Tumble processing for one section of a Document Package, you must invoke a different COPYGROUP to discontinue that function.

   A typical default COPYGROUP contains:

   - DUPlex printing option
   - MAIn input tray option
   - No OFFset option
   - No OVerLay option.

3. Invoke the COPYGROUPs by specifying them in the Structure Rule associated with the appropriate Group in the Rulebase Table.

   Each DTN in a Structure Rule may use up to five USERx options. These USERx options correlate with the COPYMODx parameters in the PRINTDEF.

   For example, if you want to Tumble DTN 20, code (20 DUP USER1) in the Structure Rule for DTN 20 and define COPYMOD1 to be the name of the COPYGROUP which invokes Tumble processing.

NOTE

Please test each COPYGROUP independently from Documerge to ensure its accuracy.

TAG Command

The DMGRBMUT TAG command indicates the names, lengths, and positions of the tags in a Tag Table. The Tag Table is optional; however, it is strongly recommended that a Tag Table be created with the DMG.MERGESET.ID Reserved Tag. DMGMERGE uses this tag to report which Document Packages are in error on the Documerge error sheet. An unlimited number of tags can be defined in a Tag Table.

IMPORTANT!

The prefix DMG. identifies Documerge Reserved Tags. Oracle reserves this convention for current and future Documerge Reserved Tags.

Code the prefix **DMG.** for a tag name only if you are coding one of the Documerge Reserved Tags documented in Chapter 8.

TAG Command Format

This command defines a single entry in a table. The TAG command associates a location in the user's input data record with a tag.

```
TAG -
          NAME=
```

The 1- to 30-character name assigned to the tag. This name must match exactly the name used as a BPSD tag name in the composition source.

Tag names cannot contain blanks or low values (x'00'). Use a period as a separator.

```
TAG NAME=INSURED.NAME
```

The use of special characters, such as apostrophes, depends on your composition system's ability to generate such characters. Therefore, Oracle does not recommend the use of special characters in tag names. However, if you use an apostrophe in a tag name, Documerge requires it to be specified as two successive apostrophes.

```
TAG NAME=INSURED''S.NAME
```

Wild-card characters are not permitted within tag names.

NOTE

A blank NAME= parameter tells DMGRBMUT to set the specified POS= and LEN= values without generating a tag in the Rulebase Library or the VRF. For example:

```
TAG NAME=                    POS=100    LEN=8
```

This type of entry adds flexibility to the POS= parameter.

- **POS=**

  The starting position of the variable data in the MSR1 (Merge Set Record) parameter passed by the VDR. This value is relative to 1.

NOTE

If the value of the RFCB-VERSION field in the DMGRFMT Control Block is 030000 or greater, the Merge Set Record passed by the VDR must begin with the 4-character string MSR1. The word MSR1 is not counted in the POS= value. POS=1 is the first character following the word MSR1 (that is, the fifth character in the Merge Set Record). See "DMGRFMT Parameter Syntax" on page 217 and "The Merge Set Record (MSR1)" on page 226 for more information.

If the RFCB-VERSION value is less than **030000**, the 4-character string **MSR1** does not precede the Merge Set Record. POS=1 is the first character in the Merge Set Record.

A single POS= value can be assigned to multiple tags, each pointing to the same data. Valid values are:

Value	Meaning
[a 1- to 9-digit numeric value]	Assigns the specific position.

*** [an asterisk]**	Assigns the next position after the *immediately preceding* tag in this Tag Table.
	In the following example, TAG2 starts in position 60, and TAG3 starts in position 65:
	`TAG NAME=TAG1 POS=50 LEN=10`
	`TAG NAME=TAG2 POS=*  LEN=5`
	`TAG NAME=TAG3 POS=*  LEN=3`
	If a tag with the POS=* value is the first tag in the Tag Table, DMGRBMUT sets the POS= value to 1.

@tagname	Assigns the same position, and therefore the same variable data, as that of a preceding tag in this Tag Table.
	Specify the name of a preceding tag whose position you want to assign also to this tag. For example:
	`TAG NAME=TAG1 POS=50 LEN=10`
	`TAG NAME=TAG2 POS=@TAG1 LEN=5`
	`TAG NAME=TAG3 POS=*  LEN=3`
	In this example, the tag name TAG2 is assigned to the variable data that starts in position 50, the same variable data to which TAG1 is assigned. Therefore, TAG3, which is assigned the next position after TAG2, starts in position 55.
	You can use this value in COBOL VDRs, for redefined entries and for the first entry under a Group.
	NOTE: The specified tag must precede this entry in this Tag Table. If not, DMGRBMUT generates an error message and ignores this entry. If the specified tag occurs more than once, the POS= value of the first occurrence is assigned. However, you can specify a specific occurrence with the @tagname(occurrence) syntax.

@tagname(occurrence)	Assigns the same position as that of a specific occurrence of the specified tag. Use this value if the specified tag occurs more than once and you do not want to assign the POS= value of the first occurrence.
	You can use this value to assign additional tag names to a single unit of variable data. For example:
	`TAG NAME=TAG1 POS=1 LEN=10`
	`TAG NAME=TAG1 POS=50 LEN=10`
	`TAG NAME=TAG2 POS=@TAG1(2) LEN=10`
	In this example, the tag name TAG2 is assigned to the variable data that starts in position 50, the same variable data to which the second occurrence of TAG1 is assigned.
	NOTE: The specified tag must precede this entry in this Tag Table and must occur the specified number of times. If not, DMGRBMUT generates an error message and ignores this entry.

Value	Meaning

END	Assigns the next position after the tag in this Tag Table that: 1) precedes this tag; 2) has the greatest sum of POS= and LEN= values.
	
	This position is assigned regardless of the position redefined by the *immediately preceding* tag.
	
	In the following example, the POS=END value is the sum of 50 and 10, the respective POS= and LEN= values for TAG1. The position redefined by TAG2 has no effect. Therefore, TAG3 starts in position 60.
	
	`TAG NAME=TAG1 POS=50 LEN=10`
	
	`TAG NAME=TAG2 POS=@TAG1 LEN=5`
	
	`TAG NAME=TAG3 POS=END LEN=3`
	
	Use the POS=END value to make sure that this tag does not overwrite data that follow the immediately preceding tag, such as a slash in a date.
	
	You can also use this value in COBOL VDRs, for redefined entries and for the first entry under a Group.
	
	Important: The effect of the POS=END value differs significantly from that of the POS=* (asterisk) value. POS=* assigns the next position after the *immediately preceding* tag, regardless of the greatest preceding sum of POS= and LEN=.
	
	In the following example, the POS=* value is the sum of 50 and 5, the respective POS= and LEN= values for TAG2. The sum of the same values for TAG1 has no effect. Therefore, TAG3 starts in position 55.
	
	`TAG NAME=TAG1 POS=50 LEN=10`
	
	`TAG NAME=TAG2 POS=@TAG1 LEN=5`
	
	`TAG NAME=TAG3 POS=* LEN=3`
	
	See "POS=" on page 184 for more information.

DMGRBMUT reports display the computed, numeric position of the tag, not the POS= values you enter (except the 1- to 9-digit numeric value).

- **LEN=**

  The maximum length of the variable data. This number must match the length specified in the VDR for the data.

  For best Documerge performance, this number should match the length of the BPSD tag in the composition source. A LEN= value less than the BPSD tag length is allowed, but this situation wastes Documerge resources because the data is padded with blanks. If the LEN= value is greater than the BPSD tag length, Documerge generates a warning that the data is truncated.

If the BPSD tag length is a block value (product of length and depth), specify the product of the BPSD tag length and depth.

NOTE

If you enter a TAG command with a blank NAME= parameter, you can specify a LEN= value of 0. This type of entry sets the POS= value without reserving additional positions.

If you specify LEN=0 for a non-blank NAME= parameter, DMGRBMUT issues error messages.

Two or more tags can have identical POS= values (pointing to the same data) but different LEN= values. One use could be to map an 8-character date field (mm/dd/yy), then remap the month, day, and year separately. For example:

```
TAG   NAME=DATE    POS=*       LEN=8
TAG   NAME=MONTH   POS=@DATE   LEN=2
TAG   NAME=        POS=*       LEN=1
TAG   NAME=DAY     POS=*       LEN=2
TAG   NAME=        POS=*       LEN=1
TAG   NAME=YEAR    POS=*       LEN=2
```

In this example, the NAME= value is blank for two tags so that no tag is created for those positions and lengths.

- **WIB=**

  The **WIB=** (Write If Blank) parameter indicates whether the VDR's DMGRFMT sub-program writes a tag to the VRF if the tag's value is blank.

  WIB= is optional. DMGRBMUT reports include the WIB= value in their Tag Table data. Valid values are:

  Y Write the tag to the VRF if the tag's value is blank.
  Y is the default value.

  N Do not write the tag to the VRF if the tag's value is blank.

  You can use WIB= with mandatory tags — tags whose names must be written to the VRF. (A mandatory tag is specified as TYPE=M in the BPSD.) When you code WIB=N for a blank mandatory tag, DMGRFMT does not write the tag to the VRF, and DMGMERGE generates message DMGMRG352W. (See *Documerge Error Messages* for more information). Therefore, WIB=N can tell you whether mandatory tags (and their BPSDs) contain unwanted blanks or actual data. However, with WIB=Y for a blank tag, DMGRFMT writes the tag to the VRF; no error message is generated.

  You can also use WIB= when you want any tag specified as DELETE=Y to appear in the BPSD even though it is blank. For example, you might have a form with three BPSDs, all with the same tag name and the DELETE=Y parameter. If the second tag were blank and you did not want the third tag's value in the second tag's BPSD, you would code WIB=Y for the second tag.

  Use the WIB=N value if a tag is not specified as TYPE=M or DELETE=Y. WIB=N saves VRF storage space and Documerge run time.

  To avoid coding WIB=N in every tag, you can use the DEFAULT command. See "DEFAULT Command" on page 136.

  For existing Rulebase Libraries, Documerge assumes a WIB= value of Y. Using the DEFAULT command eliminates unnecessary coding when you set up an existing Rulebase Library to use WIB=N.

TAG Command Control Card Sample

The following figure gives an example of the control cards for the TAG command.

Sample Control Cards for the DMGRBMUT TAG Command

```
ADD TAGTABLE
          NAME=XXXXXX(REV#) -
          TAG NAME=XXXX     POS=###     LENGTH=###     WIB=N
          TAG NAME=XXXX     POS=###     LENGTH=###     WIB=N
          TAG NAME=XXXX     POS=###     LENGTH=###     WIB=Y
          TAG NAME=XXXX     POS=###     LENGTH=###     WIB=Y
```

User Index Tags

Documerge User Index Tags are used mainly for ImageCreate, a Oracle product. However, User Index Tags can be used for any program that post-processes DMGMERGE output. User Index Tags give special information that's needed for DMGMERGE post-processing.

This post-processing depends on:

- The DMGMERGE GLOBAL command's USERTAG= control card
- The MERGEDEF PEDEF's OUTCR= control card.

See "USERTAG=" on page 400 for more information about the USERTAG= control card.

If the USERTAG= value is Y or YES and the OUTCR= value is YES, Documerge inserts User Index Tag values into each data stream that's designated for ImageCreate. In the data stream, User Index Tag values appear to the target printer as comments.

User Index Tags represent:

- Information about the documents' source files
- Information for storing image objects on DASD or optical disk
- Information about your site's printer-routing configuration.

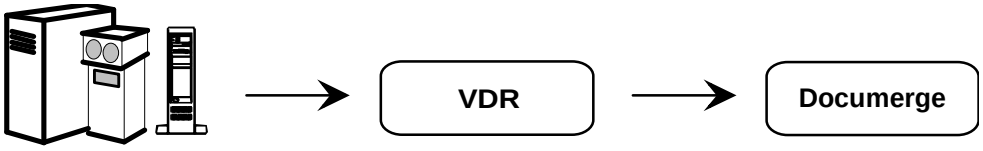
The names of all User Index Tags begin with the USER. prefix. User Index Tags for ImageCreate begin with USER.IMAGE. Before you use ImageCreate with Documerge you should add User Index Tags to the VDR or the Rulebase Library Tag Table.

The table "User Index Tags" on page 189 lists these tags. Refer to the *Imagecreate guide* for more information.

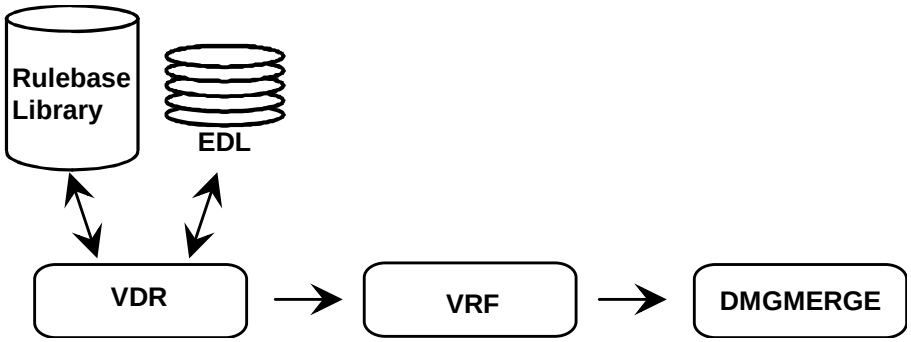
User Index Tags		
Tag Name	**Length**	**Description**
USER.IMAGE.FOLDER.ID	15	Required if you use IBM ImagePlus. The primary index key (for example, a policy number).
USER.IMAGE.FORM.NAME	10	The name of the form.
USER.IMAGE.RECEIVED.DATE	06	The date the document was received.
USER.IMAGE.FILED.DATE	06	The date the document was filed.
USER.IMAGE.ODM.SYSID	04	The system ID for the target ImagePlus ODM sub-system.
USER.IMAGE.FILE.TAB	03	The image object-filing tab category.
USER.IMAGE.LOCATION.CODE	03	The routing location.
USER.IMAGE.REGION.CODE	03	The routing region code.
USER.IMAGE.ROUTE.FLAG	01	Indicates whether routing is requested.
USER.IMAGE.PAPER.RET	01	Indicates whether paper is retained.
USER.IMAGE.DESCR.TEXT	60	A description of the input document.
USER.IMAGE.NOTES.TEXT	120	Auxiliary information.
USER.IMAGE.LINE.BUSINESS	05	The routing line-of-business.
USER.IMAGE.TRANS.TYPE	05	The routing transaction type.
USER.IMAGE.UNIT.CODE	04	The routing unit code.
USER.IMAGE.ROUTING.CODE	06	The routing code for the destination.
USER.IMAGE.PRIORITY	03	The routing priority.
USER.IMAGE.CONTROL	50	A special control for IMC data stream action verbs.
USER.IMAGE.WANG.INDEX	24	Required if you use Wang WIIS. A special Reserved Tag for Wang systems.
USER.IMAGE.FILENET.INDEX	640	Required if you use FileNet. A special Reserved Tag for FileNet systems.

The Variable Data Reformatter (VDR)

The Variable Data Reformatter (VDR) is a program that reads variable data files produced by the application system. The purpose of the VDR is to act as an interface between the application system and Documerge.



The VDR calls the Documerge Reformatter Program (DMGRFMT), a VDR subprogram. This subprogram accesses the Rulebase Library and the Electronic Document Library (EDL) that are specified in the VDR. Then, using the Documerge Variable Replacement File Writer program (DMGVRFWR), the VDR creates a Variable Replacement File (VRF). The VRF is used by the DMGMERGE program to create Document Packages.



The VDR can be written in any programming language that can call IBM assembler (ALC) programs using standard assembler calling conventions. COBOL, PL/I or ALC are examples of compatible languages. The examples contained in this manual are written in COBOL.

The Documerge distribution tape contains two sample VDRs, DMGUSER and DMGUSER2.

31-Bit Addressing

Documerge 3.x uses the extra storage available through the 31-bit address mode. With 31-bit addressing, Documerge runs in the storage above the line (above the 16 MB limit of the 24-bit address mode). Therefore, you can use more storage for buffers such as FORMSBUFF and WORKBUFF.

The REGION size determines the amount of storage below the line, but it does not determine storage above the line. Therefore, 31-bit addressing gives you more storage than the REGION allows.

If your operating system does not use 31-bit addressing, Documerge runs in the 24-bit address mode.

Existing VDRs

A Variable Data Reformatter (VDR) from previous Documerge versions can run in Documerge 3.x without changes. However, it runs in the 24-bit address mode; it does not run in the 31-bit address mode unless you recompile and relink it.

If running only the Documerge subprograms in the 31-bit address mode is enough for your requirements, changes to an existing VDR are not necessary.

Documerge Subprograms

The Documerge subprograms called by an existing VDR use 31-bit addressing automatically, with one exception — the ISICALL subprogram.

The ISICALL Subprogram

As distributed, ISICALL is `AMODE=31,RMODE=ANY`.

If your existing VDR runs with AMODE=24 and calls ISICALL dynamically, you must relink ISICALL as `AMODE=24,RMODE=24`.

CAUTION!

Do not relink the ISICALL1 subprogram.

Relink ISICALL only if your VDR calls ISICALL dynamically. If you statically link ISICALL, do not relink.

Do not relink any other programs in the Documerge load library without first consulting Oracle.

Typical VDR Flow

(1) Initialize (no calls to Documerge are required). Open any VDR input files.

(2) For each Merge Set:

- Build the RFCB, Explicit Forms List, Inline Forms List, and Merge Set record (RFCB command = "MERGESET").
- Optionally call Documerge program DMGVRFWR to write any unique or optional tags.
- Call Documerge program DMGRFMT to complete the Merge Set to the VRF.

(3) Terminate. Call DMGRFMT to close all files (RFCB command = "CLOSEALL").

VDR EXEC PARM Parameters

The following parameters offer options for VDR processing. You can code one or more of these parameters in the PARM parameter of the VDR EXEC statement. You can also code PARM parameters in a separate file. For details, see "Coding PARM Parameters in a PARMFILE" on page 374.

Parameter	Value
DMGRFPM=	Controls the creation of a DMGRFPM file for use in debugging at Oracle. This PARM captures the parameters passed to DMGRFMT, and writes them to a file called DMGRFPM. Because of file or other system inconsistencies, Oracle might not easily be able to rerun your VDR. Using the DMGRFPM file and a special VDR called DMGRFPMV that reads the DMGRFPM file, Oracle can easily recreate many problems, independently of the system on which the problems occurred. Generally, you should not enable the DMGRFPM PARM unless instructed to do so by Oracle. If so instructed, please send the DMGRFPM output file, along with your Rulebase and EDL libraries to the Oracle home office. The contents of the DMGRFPM file is internal to Oracle and is subject to change, and is not documented here. Valid values are: **N** or **NO** A DMGRFPM file is not created. **Y** or **YES** A DMGRFPM file is created. The default value is YES. If Oracle does request the DMGRFPM file, you need to implement DMGRFPM as follows: ▪ Code `DMGRFPM=Y` for the VDR step ▪ Add a DD for the DMGRFPM file with the following specifications: • LRECL=5000 • RECFM=VB • BLKSIZE=0 Most MVS systems will allocate an optimum block size when BLKSIZE=0; or you can specifically code a BLKSIZE value in your JCL for this file; minimum BLKSIZE is 5004.
DMGVRFA=	Indicates whether a DMGVRFA file is created. (Refer to "" on page 252 for more information.) When you code a sorted VRF as input to DMGMERGE, use the same DMGVRFA file for DMGMERGE that you used for DMGSORT. Valid values are: **N** or **NO** A DMGVRFA file is not created. **Y** or **YES** A DMGVRFA file is created. The default value is YES.
EDLNAMES=	Supplies the name of an appended list file that contains one to nine EDL DD names. Documerge treats the EDL names in the list as a "logical" EDL, or in other words as one big EDL. If a name in the list is preceded by a slash (/), DMGMERGE will not buffer any form from this EDL into the FORMSBUFF area. (The slash does not affect VDR processing.) The default is to assume one EDL name. See "EDLNAMES=" on page 375 and "DMGDELET" on page 458 for information about uses for multiple EDLs. Normally, this parameter is identical to the one in the corresponding DMGMERGE EXEC parameter EDLNAMES (see "EDLNAMES=" on page 375).

Parameter	Value
NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. The memory allocated for LM/MM is the number you specify multiplied by the block size of the WRKFIL. (see "WRKFIL" on page 197). Specify a value from **2** to **2048**. You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, the VDR uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, the VDR uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)

RBOPEN=	The format of the parameter is:
	RBOPEN=*value*
	Valid values are:
	OPENLIB This is the default.
	OPENLIBA If your VDR calls VLMSRVR with the OPENLIBA command, then specify RBLIB=OPENLIBA in your VDR EXEC PARM.
	Example:
	`//VDRSTEP EXEC PGM=MYVDR,PARM='RBOPEN=OPENLIBA'`
	You may specify this parameter with other VDR EXEC PARMS.
	Example:
	`//VDRSTEP EXEC PGM=MYVDR,PARM='VLMACCESS=RO,VLMCONTROL=KEEP,RBOPEN=OPENLIBA'`

VLMACCESS=	Specifies how VLAM accesses the VLAM libraries within this step. A value of 'RO' or 'READONLY' causes read-only access. Access dates are NOT written to the VLAM libraries. If a security system is installed, a user with READ only authority may submit this step for execution. A value of 'RW' or 'READWRITE' causes read-write access. Access dates are written to the VLAM Library. If a security system is installed, a user must have UPDATE authority to submit this step for execution. 'READWRITE' is the default and is more efficient than 'READONLY'. The VLMACCESS value applies to all EDLs listed in EDLNAMES.
VLMCONTROL=	Specifies how many times DMGRFMT accesses VLAM before VLAM releases exclusive control (lock) of the EDL allowing other users to access it. While maintaining exclusive control, VLAM holds the alphabetic index in memory, allowing for more efficient processing of members. VLMCONTROL applies to all EDLs listed in the EDLNAMES file. A value of 'KEEP' locks the library for the duration of DMGRFMT execution. A value of '200' allows that many accesses to the EDL before giving up control. When all requests from other users have been satisfied, VLAM will lock the EDL for another 200 accesses by DMGRFMT.
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. Specifies the amount of memory used to buffer Rulebase information. When this memory is full, areas the size of the WRKFIL blksize are paged out to WRKFIL. You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, the VDR uses WORKBUFF= and ignores NUMAREAS=.

Parameter	Value

YEAR2000=nn	(Used only in VDR processing) this parameter specifies a user-defined cut off year for Year 2000 processing. **nn** is a 2-digit number that the 2-digit year in the RFCB-EFFECTIVE-DATE field must be equal to or less than for Documerge to assign a century value of **20** to the year. (For details about the RFCB-EFFECTIVE-DATE field, see "RFCB-EFFECTIVE- DATE" on page 221.)
	For example, the MVS JCL
	`//VDRSTEP EXEC PGM=MYVDR,`
	
	`// PARM='VLMACCESS=RO WORKBUFF=500K YEAR2000=49'`
	specifies a cut off year of **49**, for which Documerge will process a value of **010149** as **01-01-2049,** and a value of **010150** as **01-01-1950**.
	If the YEAR2000 parameter does not have a value, Documerge assumes a cut off year of **50**.
	You can also code the YEAR2000 parameter in a separate file. For details, see "Coding PARM Parameters in a PARMFILE" on page 374.

VDR Files

The following are VDR files found in "VDR Sample JCL" on page 198.

VLM2LIB	Indicates the name of the EDL.
DMGVRF1	Indicates the file name that contains the VRF. The file's physical record format is undefined.
DMGVRFA	The name of the file that contains VRF space allocation information. Can be any RECFM, BLKSIZE, and LRECL (minimum 33 if variable — otherwise, minimum 29). Usually, this file is RECFM=F,BLKSIZE=80,LRECL=80.
RBLIB	Indicates the name of the Rulebase library. This is the file that contains the data from the application database.
WRKFIL	Indicates a work file used by the List Manager/Memory Manager (LM/MM) when data does not fit into memory allocated to LM/MM. DMGRFMT uses LM/MM to buffer Rulebase information. The previous name of this file was ISIWORK.
MESSAGE	Contains messages produced by DMGRFMT.
ISIWTL	Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTL is intended for programmer and application user error messages. Using Oracle standards, any program that has written to ISIWTL has encountered a fatal error.
ISIWTO	Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTO is intended for computer operator messages. Using Oracle standards, any program that has written to ISIWTO has encountered a fatal error.

VDR Sample JCL

```
// JOB CARD INFO
//*
//*   CLIENT VDR WITH DOCUMERGE V. 3.2 REFORMATTER (DMGRFMT)
//*
//JOBLIB   DD DSN=YOUR.VDR.LOADLIB,DISP=SHR
//         DD DSN=DOCUMERG.V03R02.LOADLIB,DISP=SHR
//*
//VDRSTEP  EXEC PGM=YOURVDR,REGION=2048K,
//            PARM='WORKBUFF=500K VLMCONTROL=KEEP VLMACCESS=RO'
//VLM2LIB  DD DSN=YOUR.VLAM.V02R07.EDL,
//            DISP=SHR
//DMGVRF1  DD DSN=YOUR.DOCUMERG.V03R02.VRF,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(TRK,(15,15)),
//            DCB=BLKSIZE=23476
//DMGVRFA  DD DSN=YOUR.DOCUMERG.V03R02.VRFA,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(TRK,(15,15)),
//            DCB=(RECFM=F,BLKSIZE=80,LRECL=80)
//RBLIB    DD DSN=YOUR.DOCUMERG.V03R02.RBLIB,
//            DISP=SHR
//WRKFIL   DD DSN=&&WRKFIL,
//            UNIT=SYSDA,
//            DISP=(NEW,DELETE,DELETE),
//            DCB=BLKSIZE=23476,
//            SPACE=(TRK,(15,15))
//MESSAGE  DD SYSOUT=*,      <= DMGRFMT MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//ISIWTL   DD SYSOUT=*
//ISIWTO   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//INPUT    DD *
 VDR INPUT DATA
/*
//*
//
```

Using BLKSIZE=0 with the VRF

Although the VRF is RECFM=U, you can specify BLKSIZE=0 if the VRF is a DASD or tape file. This value allows the MVS system to use the optimum block size for the output device (if your system supports BLKSIZE=0).

To Specify BLKSIZE=0 for a VRF

1. Run the DMGOPNCL program to allocate the VRF.
2. Run the VDR to create the VRF.

The following figure shows an example of the JCL for the procedure:

Sample JCL for Using BLKSIZE=0 with VRF

```
//*-------------------------------------------------------------------
//DMGOPNCL EXEC  PGM=DMGOPNCL
//DMGVRF1  DD DSN=Z513004.VRF1,
//            UNIT=SYSDA,
//            DISP=(NEW,CATLG),
//            SPACE=(TRK,(10,5)),
//            DCB=(RECFM=FB,LRECL=1,BLKSIZE=0)
//MESSAGE  DD SYSOUT=*
//SYSIN    DD *
DMGVRF1
/*
//*-------------------------------------------------------------------
//VDRSTEP  EXEC PGM=DMGUSER,REGION=4M,
//            PARM='WORKBUFF=200K VLMCONTROL=KEEP VLMACCESS=RO'
//SYSUDUMP DD SYSOUT=*
//*
//SYSOUT   DD SYSOUT=*
//MESSAGE  DD SYSOUT=*,                <= DMGRFMT MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//ISIWTL   DD SYSOUT=*
//ISIWTO   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//ISIWORK  DD DSN=&&ISIWORK,               VERSION 2
//            UNIT=SYSDA,
//            DISP=(NEW,PASS,DELETE),
//            SPACE=(TRK,(2,2)),
//            DCB=(BLKSIZE=23476)
//WRKFIL   DD DSN=&&WRKFIL,                VERSION 3
//            UNIT=SYSDA,
//            DISP=(NEW,PASS,DELETE),
//            SPACE=(TRK,(2,2)),
//            DCB=(BLKSIZE=23476)
//VLM2LIB  DD DSN=ISIV.Z513004.V03R02.EDL,
//            DISP=SHR
//DMGVRF1  DD DSN=Z513004.VRF1,
//            DISP=MOD,DCB=RECFM=U
```

Writing Your VDR

The following procedure assumes the use of COBOL. Sample DMGUSER and DMGUSER2 VDRs are on the Documerge installation tape.

NOTE

We recommend that you use the Rulebase Library and DMGRFMT. However, your VDR is not required to reference the Documerge Rulebase Library for the data dictionary used to complete Documerge processing. If you need to bypass Rulebase Library processing at your site, see "Bypassing the Rulebase Library" on page 244 for coding instructions.

To Create a Custom VDR Program

1. Code a site-specific program that accesses your application data.
2. Import the DMGRFMT control block from the TEXT dataset delivered with Documerge. The control block for COBOL is called DMGRFCBC. (Control blocks for Assembler and PL/I programs are called DMGRFCBA and DMGRFCBP, respectively.) An explanation of each of the fields in the control block may be found in the section entitled "Description of DMGRFMT Control Block Fields."
3. Create a logical record in memory which will be passed to DMGRFMT. This is known as the Merge Set Record (also called tagged data record). The fields in this record (which are to be output as tagged data in the VRF) must have already been defined in a Tag table in the Documerge Rulebase Library. DMGRFMT uses the position and length parameter values for each tag in the Tag table to map to the information in the Merge Set record. For example, the Merge Set record in Working Storage might look like this:

```
01  DMGRFMT-MERGE-DATA-RECORD.
    05  POLICY-NUMBER                          PIC X(10).
    05  NAME                                   PIC X(30).
    05  ADDRESS                          PIC X(30).
    05  CITY-STATE-ZIP                   PIC X(30).
```

The corresponding Rulebase Tag Table source would look like this:

```
ADD TAGTABLE NAME=TEST.TAG.TABLE
TAG NAME='DMG.MERGESET.ID'              LEN=00010  POS=1
TAG NAME='POLICY.NUMBER'                LEN=00010  POS=@POLICY.NUMBER
TAG NAME='INSURED.NAME'                 LEN=00030  POS=*
TAG NAME='INSURED.ADDRESS1'             LEN=00030  POS=*
TAG NAME='INSURED.ADDRESS2'             LEN=00030  POS=*
```

(Note that more than one tag may be mapped to the same field in the Merge Set record.)

4 The following is the basic format for the VDR logic:

a Program initialization (including opening input file or files).

b For each Merge Set, perform the following:

- Merge Set initialization.
- Move Explicit form names to the Explicit Forms Tables as needed.
- Move inline form names to the Inline Forms Tables as needed.
- Move data to the Merge Set Record as needed.
- Write any inline forms, or any tags required for this Merge Set which are not defined in the Rulebase Tag Table, to the VRF via individual calls to DMGVRFWR. If DMGVRFWR is called, all such calls must be performed before any calls to DMGRFMT for a specific Merge Set. See "The DMGVRFWR Subprogram" on page 240 for more information.
- Call DMGRFMT to create the Merge Set. The data in the control block, Explicit forms table, inline forms table, and merge data record will be passed. See the section entitled "Calling syntax for DMGRFMT" for more information.

c At program end, call DMGRFMT one last time, passing it the CLOSE or CLOSEALL command to close the VRF.

Compiling Your VDR

There are no special Documerge requirements for compiling your VDR. There are no special source libraries or copybooks.

You can compile for 24- or 31-bit addressing.

Linkediting Your VDR

When you linkedit your VDR, you must point to the Documerge LOADLIB in your SYSLIB, for program ISICALL.

You can linkedit your VDR with any valid system AMODE and RMODE.

The DMGUSER VDR

The DMGUSER VDR is designed to let the user define data streams in JCL. DMGUSER is also used to verify installation of Documerge.

Guidelines for Coding DMGUSER Input

Here are some guidelines for using or modifying DMGUSER:

- When preparing DMGUSER JCL, it is important to distinguish between the command processing and Merge Set data record (MSR1) processing, which maps data to positions in the specified Tag Table. The input file, DMGUSER SYSIN, has the following general structure:

  Command(s) for Merge Set 1
  Merge Set data record(s) for Merge Set 1
  Command(s) for Merge Set 2
  Merge Set data record(s) for Merge Set 2
  .
  .
  .
  Command(s) for Merge Set *n*
  Merge Set data record(s) for Merge Set *n*

  To signal the number of Merge Set data records to process, you need to use either

  - The *NUMREC* or *LENGTH* command coded with the number of Merge Set data records to process. (*NUMREC* has a default of 5 and specifies the actual number of records; you determine the value for *LENGTH* by multiplying the number of records by 80).

    -or-

  - The *END* command coded in a set of data records *before* a coded *NUMREC* or *LENGTH* command

- The *LENGTH* and *NUMREC* commands both specify the same thing. That is, the number of 80-byte SYSIN data records that comprise the Merge Set record (MSR1) data. If you code both (or multiples) of *LENGTH* and *NUMREC*, DMGUSER uses whichever is coded last.

  When specifying the length and number of input records, follow these rules:

  - As distributed, DMGUSER specifies the processing of up to a maximum of 1000 Merge Set data records. If you need to process more than 1000 records, change the OCCURS value for item *CARD-ENTRY* in the DMGUSER source code, and then recompile and relink.
  - **The SYSIN LRECL must be 80 with fixed length records**. You can vary the block size, **but it must be a multiple of 80**.

- *Single-SYSIN* commands require only one statement for execution. *Multi-SYSIN* commands require multiple SYSIN statements for execution. Multi-SYSIN commands have a "begin" command, then data, then an "end" command. The following table shows the single and multi-SYSIN commands:

Single-SYSIN Commands		Multi-SYSIN Commands	
CODE	*END*	*EXPBEG*	*EXPEND*
SWITCHES	*NUMREC*	*EXP2BEG*	*EXP2END*
FILE	*LENGTH*	*INLBEG*	*INLEND*
EFFDAT	*EFFDAT2*	*MSGBEG*	*MSGEND*
EFFMSG			

- **You cannot code a command as data in a Merge Set record.** DMGUSER checks to see if a given record is a command or a component of a multi-SYSIN command. If both of these checks are negative, then DMGUSER treats the record as a Merge Set data record.
- Although there is no limit (excluding the 23 table limit for the EXP, EXP2, and INL2 forms tables) on the number of commands you can code for a Merge Set, **there is no reason to code more than one of a given single-SYSIN command per Merge Set**. For example, you could code multiple *NUMREC* commands for a single Merge Set, but only the last one would be used to determine the number of data records.

Coding Multiple Forms Tables in DMGUSER

DMGUSER now supports multiple Explicit forms and Inline forms lists. Here are the rules for coding multiple forms lists:

- There can be up to 23 Explicit, Explicit-2, or Inline forms lists per Merge Set. (DMGRFMT can process as many as 25 parameters, including the required RFCB and the MSR1 Merge Set data record parameters.)
- You can now process an Inline form ahead of an Explicit form when both have the same DTN. To accomplish the intended forms sequencing, Documerge processes forms with the same DTN according to their sequence in the forms lists.

IMPORTANT!

DMGUSER now requires about 1 MB (1024K) more REGION, so you might have to increase the REGION size for your JCL for DMGUSER.

DMGUSER Sample JCL

```
//DMGUSER   ** put your job card here **
//*
//* *******************************************************************
//* **                                                               **
//* **    DMGUSER VDR WITH DOCUMERGE V. 3.2 REFORMATTER (DMGRFMT)     **
//* **                                                               **
//* *******************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//VDRSTEP  EXEC PGM=DMGUSER,REGION=4M,
//            PARM='WORKBUFF=500K VLMCONTROL=KEEP VLMACCESS=RO'
//VLM2LIB  DD DSN=your.documerg.v03r02.edl,
//            DISP=SHR
//DMGVRF1  DD DSN=your.documerg.v03r02.vrf,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(15,15),RLSE),
//            DCB=BLKSIZE=23476                  HALF TRACK
//DMGVRFA  DD DSN=your.documerg.v03r02.vrfa,
//            DISP=(,CATLG,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,1),
//            DCB=(RECFM=F,LRECL=29,BLKSIZE=29)
//RBLIB    DD DSN=your.documerg.v03r02.rbl,
//            DISP=SHR
//WRKFIL   DD DSN=&&WRKFIL,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                  HALF TRACK
//MESSAGE  DD SYSOUT=*,     <= DMGRFMT MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//ISIWTL   DD SYSOUT=*
//ISIWTO   DD SYSOUT=*
//* -------------------------------------------------------------------
//* EITHER CHANGE OR REPLACE THE FOLLOWING INPUT CONTROL CARDS
//* WITH YOUR DATA CONTROL CARDS.
//* -------------------------------------------------------------------
//INPUT    DD *
*CODE*   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX#####
*EFFDAT* YYMMDD
*LENGTH* #####
*NUMREC* #####
*EXPBEG*
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX#####
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX#####
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX#####
*EXPEND*
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
*END*
*EXPBEG*
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX#####
*EXPEND*
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
*END*
/*
//*
//
```

DMGUSER Files

The following are the DMGUSER datasets referenced in "Sample JCL for Using BLKSIZE=0 with VRF" on page 199.

Dataset	Contents
VLM2LIB	The EDL Library
DMGVRF1	The VRF
DMGVRFA	The DMGVRFA file
RBLIB	The Rulebase Library
WRKFIL	The List Manager/Memory Manager (LM/MM) overflow BSAM file
MESSAGE	The DMGRFMT message file
SYSPRINT	The System print file
SYSOUT	The COBOL message file
INPUT	The variable data input-stream. Code all parameter names in columns 1-9 and parameter values starting in column 11:

CODE (Optional) the name of the Rulebase table. Code in columns 11-40.

DMGUSER assumes a revision level of zero, which means that it uses the highest revision available at processing time.

EFFDAT (Optional — if *EFFDAT* is not coded, effective-date processing is not performed.) The date that invokes Effective Date Processing. Code in **yymmdd** format in columns 11-16.

Effective Date Processing overrides Revision Level Processing. Each EDL form specified for the Merge Set must have at least one revision with an effective date that is earlier than or the same as the date you code in *EFFDAT*. Otherwise the form is disqualified from the Merge Set.

If more than one revision of a form is effective, the revision with the highest level is used.

NOTE: If you do not code a revision level and *EFFDAT* parameter, the revision with the highest level at merge time is used. If you code *EFFDAT*, the highest-level revision with an effective date that is earlier than or the same as *EFFDAT* is used, regardless of any coded revision level.

EFFDAT2 (Optional) use to input a value for the FCB-ALT-EFFECTIVE-DATE field when running DMGUSER. Code in **yyyymmdd** format in columns 11-18.

For example:

```
*EFFDAT2*  20010101
```

inputs the date **01-01-2001** to DMGUSER.

Effective Date Processing overrides Revision Level Processing. Each EDL form specified for the Merge Set must have at least one revision with an effective date that is earlier than or the same as the date you code in *EFFDAT*. Otherwise the form is disqualified from the Merge Set.

If more than one revision of a form is effective, the revision with the highest level is used.

Dataset	Contents	
	EFFMSG	(Optional) the notification given if Effective Date Processing disqualifies a form from the Merge Set. This switch sets the RFCB-NO-EFF-DATE-MSG switch. Each EDL form for the Merge Set must have at least one revision with an effective date that is earlier than or the same as the date in *EFFDAT*. Otherwise the form is disqualified from the Merge Set. Code the value in column 11. Valid values are • **I**The disqualified form is ignored. No user notification is given. • **W**The disqualified form is ignored. A warning message is issued. • **E**(Default) the following notification is given: ▪ Messages DMGWRT435I and DMGWRT739I in the DMGMERGE message output file ▪ Name of disqualified form in the VRF's DMG.MISSING.FORMS Reserved Tag
	END	(Optional) signals the end of input variable-data records. You can use this command to terminate reading of Merge Set data records before the value specifed by *NUMREC* or *LENGTH* is reached. For example, if *NUMREC* has a value of 5 and you code an *END* because you are processing only 3 data records; records 4 and 5 are assumed to be blank (spaces or hex 40).
	EXPBEG	The beginning of the Explicit Forms List (optional). Code in columns 1-9; column 10 must be blank. Code the names of EDL forms on the lines following *EXPBEG*, one form name to a line. A maximum of 962 form names can be coded. You can code more than one *EXPBEG*/*EXPEND* table in the same Merge Set if you need more than 962 forms. Each form name can have a maximum of 32 characters, and can be followed by an optional five-character revision level starting in column 33. If you do not code the revision level and *EFFDAT* the revision with the highest level at DMGMERGE time is used. If you code *EFFDAT* the highest-level revision with an effective date that is earlier than or the same as *EFFDAT* is used, regardless of any revision level you code.
	EXPEND	The end of the Explicit Forms List. (Required if you code *EXPBEG*.)

Dataset	Contents
EXP2BEG	(Optional) the beginning of an EXP2 Explicit Forms table for user-selectable form print options. Follow the ***EXP2BEG*** with up to 272 two-statement entries (making up to 544 statements total). You can code more than one *EXP2BEG* /*EXP2END* table in the same Merge Set if you need more than 272 forms. Each form entry consists of two statements: 1 Code the name of an EDL form and other identifying details in a first statement that has the following format: • Columns 1 to 32, an alphanumeric form name • Columns 33 to 37, optional 5-character revision level 2 (Required, even if all blank) you can code overrides for the form's Rulebase specified DTN and/or print options. This second statement has the following format: • Columns 1 to 5, can be blank or a new DTN for the form • Columns 6 to 61, can contain up to 14 print option override values. These values are position sensitive: enter each value left-justified in its four-character field that matches the same position in the Explicit Forms List shown in "Format of the Explicit Forms List (EXP2)" on page 229 For definitions of print option override codes, see "EXPLICIT-OPTIONS-OVERRIDE Fields" on page 231. • Leave columns 62 through 80 blank.
EXP2END	The end of the EXP2 Explicit Forms table with user-selectable print options. (Required if you code *EXP2BEG*.)
FILE	(Optional) sets the VRF file name and optionally the VRFA (allocation) file name. For MVS, you need a DD for each file name. To set just the VRF file name, code the file name in columns 11-18. To set the VRF file name and associated VRFA allocation file name, code a slash (/) in column 11, code the VRF file name in columns 12-19, and code the allocation file name in columns 20-27.
INLBEG	(Optional) the beginning of an Inline Forms List (optional). Code in columns 1-9; column 10 must be blank. Code the names of the forms on the lines following *INLBEG*, one form name to a line. A maximum of 909 form names can be coded. You can code more than one *INLBEG*/*INLEND* table in the same Merge Set if you need more than 909 forms. Each form name can have a maximum of 30 characters, the name must be followed by a 5-character DTN and a 1-character line-end character that can be any character that doesn't appear in the print lines.
INLEND	The end of the Explicit Forms List. (Required if you code *INLBEG*.)
LENGTH	(Optional, not needed if you code ***NUMREC***.) The total number of characters (80000 maximum, default of 400) in the input variable data records to follow *EXPBEG*. Code 5 digits (leading zeros required) in columns 11-16. The *LENGTH* value is the product of the number of data records multiplied by 80. (Data records are 80 characters long.)

Dataset	Contents
MSGBEG	(Optional) provides the content of the DMG.VDR. ERRORS tag. The VDR uses this tag to generate messages when DMGMERGE routes all the Groups for the Merge Set to their respective error data sets. Follow *MSGBEG* by one to 10 data records; each record must contain one message. If you code more than one *MSGBEG* in a single Merge Set, DMGUSER uses only the last *MSGBEG* and its associated data.
MSGEND	(Required if you code **MSGBEG***) code after the last VDR error message following a *MSGBEG* command.
NUMREC	(Optional, not needed if you code ***LENGTH***.) The number of variable-data records (1000 maximum, default of 5) to follow *EXPBEG*. Code 5 digits (leading zeros required) in columns 11-15.
SWITCHES	(Optional) sets the RFCB SWITCHES in the RFCB (Reformatter) contol block. If you just code **SWITCHES*, you set all nine at the same time. Code a **N** or **Y** value for each switch in the column indicated: • **Column 11** — the RFCB-SUPPRESS-SYSPRINT switch • **Column 12** — the RFCB-WRITE-RFCB switch • **Column 13** — the RFCB-WRITE-EXPLICIT-FORMS switch • **Column 14** — the RFCB-SUPPRESS-TAG-TABLE switch. For DMGUSER, leave RFCB-SUPPRESS-TAG-TABLE blank or code **N**. • **Column 15** — the RFCB-ALTERNATE-PARM switch. For DMGUSER, always leave RFCB-ALTERNATE-PARM blank or code it as **N**. • **Column 16** — the RFCB-NO-EFF-DATE-MSG switch • **Column 17** — the RFCB-END-MERGESET7- switch • **Column 18** — the RFCB-ALLOW-MISSING-RULEBASE switch • **Column 19** — the RFCB-SUPPRESS-IMPLICIT-FORMS switch • **Column 20-72** — blank, reserved for future switches For details about these switches, see "Descriptions of DMGRFMT Control Block Fields" on page 219.
ISIWTL	Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTL is intended for programmer and application user error messages. Using Oracle standards, any program that has written to ISIWTL has encountered a fatal error.
ISIWTO	Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTO is intended for computer operator messages. Using Oracle standards, any program that has written to ISIWTO has encountered a fatal error.

DMGUSER2

DMGUSER2 permits the definition of variable-data input stream with the JCL as an input file. It provides additional flexibility and is provided as an all purpose VDR.

Sample DMGUSER2 JCL

```
//DMGUSER2  ** put your job card here **
//*
//* ****************************************************************
//* **                                                            **
//* **    DMGUSER2 VDR WITH DOCUMERGE V. 3.2 REFORMATTER (DMGRFMT)   **
//* **                                                            **
//* ****************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//VDRSTEP  EXEC PGM=DMGUSER2,REGION=4M,
//            PARM='WORKBUFF=500K VLMCONTROL=KEEP VLMACCESS=RO'
//VLM2LIB  DD DSN=your.documerg.v03r02.edl,
//            DISP=SHR
//DMGVRF1  DD DSN=your.documerg.v03r02.vrf,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(15,15),RLSE),
//            DCB=BLKSIZE=23476                  HALF TRACK
//DMGVRFA  DD DSN=your.documerg.v03r02.vrfa,
//            DISP=(,CATLG,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,1),
//            DCB=(RECFM=F,LRECL=29,BLKSIZE=29)
//RBLIB    DD DSN=your.documerg.v03r02.rbl,
//            DISP=SHR
//WRKFIL   DD DSN=&&WRKFIL,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                  HALF TRACK
//MESSAGE  DD SYSOUT=*,     <= DMGRFMT MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//VDRMSGS  DD SYSOUT=*      <=DMGUSER2 MESSAGES
//SYSUDUMP DD SYSOUT=*
//ISIWTL   DD SYSOUT=*
//ISIWTO   DD SYSOUT=*
//* ****************************************************************
//* SUPPLY YOUR INPUT CONTROL CARD DATA
//* ****************************************************************
//VDRINPT  DD *
CMT SUPPLY YOUR INPUT CONTROL CARDS HERE
/*
//
```

DMGUSER2 Files

The following are the DMGUSER2 datasets in "Sample DMGUSER2 JCL" on page 209.

Dataset	Contents
VLM2LIB	The EDL.
DMGVRF1	The VRF.
DMGVRFA	The VRF Allocation (DMGVRFA) file.
RBLIB	The Rulebase Library.
WRKFIL	The List Manager/Memory Manager (LM/MM) overflow BSAM file.
MESSAGE	The DMGRFMT message file.
ISIWTL	The message file for programmer and application user error messages. Using Oracle standards, any program that has written to ISIWTL has encountered a fatal error.
ISIWTO	The message file for computer operator messages. Using Oracle standards, any program that has written to ISIWTO has encountered a fatal error.
SYSPRINT	The System print file.
SYSOUT	The COBOL message file.
VDRINPT	The dataset used as input to DMGUSER2.
VDRMSGS	Contains DMGUSER2 messages.

DMGUSER2 Record Identifiers

This section describes a basic outline of what the input to DMGUSER2 is like and a description of each record identifier. In general, each record has a three-character record identifier as the first three bytes of the record. These identifiers indicate to the VDR what to do with that record. The recommended physical characteristics for VDRNPT are the following:

- Record format = FB
- Record length = 80

A sample input file is in "DMGUSER2 Input File" on page 246.

Begin Program Globals / End Program Globals (BPG/EPG)

Begin and end program globals indicates the beginning and end of program globals information. An eight-byte global name identifier can be specified. The RTD section is the only required information in this section which sets up the skeleton in working storage for the tag data passed to DMGRFMT. One program globals section is needed each time the Rulebase name is changed with a new outline of the Rulebase data being used. This goes before the Begin Merge Set. At least the first Merge Set needs a CBC section with a Rulebase name.

- **TBN** (Test Block Names)
  This consists of three codes. If the first one is 'Y', beginning and ending block names are matched up. If the second one is 'Y', the input records are displayed and if the third one is 'Y', READY-TRACE is set on.
- **RCV** (Override Return codes)
  Enter the return code values to be generated in the following order (each has four numeric positions): P-level, F-level, E-level, C-level, W-level and I-level.
- **DIA** (Diagnostic Formatting)
  For the diagnostic report, enter the number of lines per page (66 default), the diagnostic system (VDR default), and the diagnostic program (MSG default).

- **RTD** (Record Type Definition)
  Use this section to set up a working storage skeleton of the Rulebase. Enter a unique type to be used to associate the field with the VDR data entered, enter the length of that particular field, the number of times it should occur in the Rulebase and the sequence number it should have. The sequence is the order in the Rulebase in which it should appear. The program allocates storage in this order and when processing Merge Sets, only variables to be set up need to be referenced.

Begin Merge Set / End Merge Set BMS/EMS

Begin and End Merge sets indicates the beginning and end of Merge Set information. An eight byte Merge Set identifier can be specified. If the Test Block Names (TBN) check is 'Y', the beginning and ending identifiers are verified for accuracy. A Rulebase name is the only required Merge Set information.

Begin Control Block / End Control Block BCB/ECB

Indicates the beginning and end of DMGRFMT Control Block information that can be specified on input. Most of the DMGRFMT Control Block parameters can be specified. They are described below.

- **CBC** (Control Block Command)
  This is the eight byte DMGRFMT control block command. The valid values are MERGESET for creating Merge Sets and CLOSEDD for closing one VRF.
- **RBN** (Rulebase Name)
  This is the name of the Rulebase that is used for this Merge Set. Maximum of 30 characters.
- **RBL** (Rulebase Revision level)
  This is the five byte revision level of the Rulebase that is used for this Merge Set. The highest revision level is the default.
- **VRF** (VRF file name)
  This is the VRF file name of the JCL control statement.
- **EFD** (Effective Date)
  This is the effective date for specifying particular revision levels of forms in the EDL. Revision level processing overrides effective date.
- **SSP** (Suppress SYSPRINT Processing)
  This sets the suppress SYSPRINT value of the DMGRFMT control block.

Begin Explicit Forms / End Explicit Forms (BEF/EEF)

This is the Explicit Forms list. One form name and its revision level can be specified on each record. As many as 1999 Explicit Forms can be entered. An eight byte Explicit Forms list identifier can be specified. If the TBN names check is 'Y', the beginning and ending identifiers are verified for accuracy.

- **EFN** (Explicit Form Name / Revision Level)
  This is the form name as it is cataloged in the EDL and the revision level desired. The highest revision is the default.

Begin Inline Forms / End Inline Forms BIF/EIF

This is the Inline Forms list. One form name can be specified on one record. If the line end character is not on each record, the VDR places it in the byte just beyond the line length. An 8-byte Inline Forms list identifier can be specified. If the TBN names check is 'Y', the beginning and ending identifiers are verified for accuracy. As many as 1999 Inline Forms can be entered with BIF and EIF coded for each.

- **IFN** (Inline Form Name / DTN / Line End Char / Line Length / Line End Present)
  This is the form name and includes the DTN and line end character to be used if none is present on the inline form text records. Line length can be specified and defaults to 74. If the line end character is added to the end of each line, the line end present flag must be 'N'. This form name and its information should be placed before the text lines.
- **IFT** (Inline Form Text)
  This is the text for each record of the inline form. This should include the line end character, but doesn't have to if the line end present flag is coded as 'N'. The maximum amount of text that can be entered is 10,000 bytes.

Begin Variable Data / End Variable Data (BVD/EVD)

These are the variable data records. These records are sent to DMGRFMT to be written to the VRF. There is a maximum storage of 20,000 bytes. They are coded to match the RTD type from the globals section and only need to be entered when needed. The RTD section takes care of their placement to match the Rulebase Tag Table.

- **VDR** (Variable Data Record)
  The variable data for a Merge Set is on these records. Enter them in any order and code them to match the RTD type. The VDR knows from the RTD section where to place them to match the Rulebase Tag Table and what length to use.

Begin Tagged Data / End Tagged Data (BTD/ETD)

This is any tag information that is to be written directly to the VRF, via DMGVRFWR.

- **TDI** (Tagged Data Item)
  Each TDI record has the tag name, length of the data, and the variable data. There is no limit to the number of records that can be entered.

Error Processing

Error processing consists of

- Checking the presence of block sections (BMS and EMS, BEF and EEF, BIF and EIF, BTD and ETD, BVD and EVD, BCB and ECB, BPG and EPG) for matching beginning and ending identifier names.
- Checking for valid record identifiers.
- Checking for valid data in numeric fields that are passed to DMGRFMT.
- Checking for valid record type entries on Variable Data Records VDRs.
- Checking the globals section to ensure a Rulebase has been defined.
- Checking for too many Variable Data Record (VDR) occurrences for the working storage area to match with the previously coded Record Type Definitions (RTD).
- Checking for the maximum amount of Inline Forms text of 10,000 bytes.

- Checking for the end of the input file without the end of the Merge Set.

DMGUSER2 Record Identifiers

Required / Optional	Record Identifier	Description
required	**BPG**	Begin Program Globals
optional	**TBN**	Test Block Name, matches block names, display input recs, trace on
optional	**RCV**	Override Return Code Values
optional	**DIA**	Diagnostic formatting, lines per page, system file name and VRF
required	**RTD**	Matches outline of Rulebase for working storage allocation
required	**RTD**	Record Type Definition (matches VDR for variable), variable length, number of occurrences, sequence in Rulebase
required	**EPG**	End Program Globals
required	**BMS**	Begin Merge Set
required	**BCB**	Begin Control Block
optional	**CBC**	Control Block Command
required	**RBN**	Rulebase Name
optional	**RBL**	Rulebase Revision level
optional	**VRF**	VRF ddname
optional	**EFD**	Effective Date
optional	**SSP**	Suppress SYSPRINT
required	**ECB**	End Control Block
optional	**BEF**	Begin Explicit Forms
optional	**EFN**	Explicit Form Name, revision level, occurs one time
optional	**EEF**	End Explicit Forms
optional	**BIF**	Begin Inline Form
optional	**IFN**	Name, DTN Line end char, Line length, Line end present. Occurs multiple times
optional	**IFT**	Inline Form text and Line end character. Occurs multiple times
optional	**EIF**	End Inline Form
optional	**BVD**	Begin Variable Data
optional	**VDR**	Record Type Definition, Variable Data Record (up to 71 bytes)
optional	**EVD**	End Variable Data
optional	**BTD**	Begin Tagged Data (Tagged Data Item Tag Name, Length, Variable Data)
optional	**ETD**	End Tag Data
required	**EMS**	End Merge Set
optional	**CMT**	Comment (comment records can be inserted anywhere)

The VDR Subprograms

The VDR, whether supplied by Oracle or written at your site, depends heavily on the following subprograms to construct its output (the VRF) correctly:

- DMGRFMT
- DMGVRFWR
- ISIFLAST

These subprograms create a common structure for the data being passed to the next step in Documerge processing and are fully capable of handling undefined data structures. The functions they perform include formatting and writing the data, as well as performing all I/O against the Rulebase Library, EDL, and VRF.

The Documerge Reformatter (DMGRFMT)

DMGRFMT is the subprogram called by the VDR to produce the data for the VRF. The data it produces includes the tagged data elements, and the list of forms and sort keys for each Merge Set in the execution.

DMGRFMT calls the subprogram DMGVRFWR to write the tagged data to the output dataset specified in the DMGVRF1 DD statement. The number of Merge Sets and the number of tagged data elements included for each Merge Set are reported to the print file defined by the SYSPRINT DD statement.

These subprograms create a common structure for the data being passed to the next step in Documerge processing and are fully capable of handling undefined data structures. The functions they perform include formatting and writing the data, as well as performing all I/O against the Rulebase Library, EDL, and VRF.

Compatibility with Reformatters from Previous Versions

Documerge 3.x is compatible with:

- the DMGFORMT reformatter subprogram from version 1.7

  You can call DMGRFMT using the DMGFORMT input parameters, which are converted to the version 2 format.

- the version 2 format for:
  - DMGRFMT input parameters
  - the DMGRFMT Control Block (RFCB)

However, you cannot use the features of Documerge 3.x, such as DTN and Print Options override, with DMGFORMT or the version 2 DMGRFMT format.

Oracle recommends that you use DMGRFMT and the Documerge 3.x format when you write new VDRs.

DMGRFMT Processing

DMGRFMT follows these steps to perform its processing:

(1) Loads the Rulebase Library File specified in the RBLIB DD statement.

(2) References a 30-character Rulebase Library name with every call to DMGRFMT so that it can determine the correct rules for reformatting the input data records. This Rulebase Library also supplies information about the documents comprising the printed copies, and the rules followed for constructing the output, and the building of the sort keys used by DMGSORT.

(3) Loads each Rulebase Library into an internal memory area. The VDR EXEC parameter NUMAREAS= or WORKBUFF= defines the size of the internal memory area. If this area becomes full, data is written to WRKFIL by the List Manager/Memory Manager (LM/MM), an internal system of Oracle programs.

(4) Constructs tagged data by using the data dictionary contained in the Rulebase Library defined in the DMGTABL DD statement.

(5) Verifies the existence of all referenced document images and their Document Type Numbers in the Electronic Document Library.

(6) Builds a table of all the form names referenced by either the Rulebase Library or the input data, and a buffer in which to hold the data stream containing the forms. Using this buffer, library members that are repeated in subsequent Merge Sets do not have to be referenced on the disk more than once.

(7) Constructs DMG.FLST.*groupname* and DMG.OPT.*groupname* Reserved Tags for each Group defined in the Rulebase Library, which contains the form names specified in the Rulebase Library and the input data. DMGRFMT can construct other Reserved Tags, depending on processing requirements.

(8) Uses the tagged data elements to determine the documents to be printed and the order of collation for each Merge Set. The Rulebase Library also contains the Structure Rule which identifies the DTNs printed for each Merge Set. The output of the Merge Sets appears in the same order that the DTNs are specified. Boilerplates identified with the same DTN are printed in the following order:

- **a** Implicit Forms named in the Rulebase Library
- **b** Explicit Forms and Inline Forms in the sequence passed by the VDR

(9) Constructs sort key tags used by the DMGSORT program. These sort keys are identified by their associated Group Table in the Rulebase Library.

(10) Reports all errors and the number of Merge Sets present to the print file defined in the MESSAGE DD statement.

Refer to "Documerge Reserved Tags" on page 283 for more information on the Documerge Reserved Tags and their format in the VRF.

Calling DMGRFMT

The DMGRFMT subprogram is defined as:

```
01  DMGRFMT                              PIC X(8)  VALUE 'DMGRFMT'.
```

DMGRFMT must be called with the subprogram ISICALL. This Oracle program calls other subprograms dynamically. ISICALL is included in the Documerge LOADLIB.

The COBOL syntax for calling the DMGRFMT subprogram is displayed below. The syntax includes the DMGRFMT input parameters you use in the call. Each call to DMGRFMT completes one Merge Set.

Sample COBOL calling syntax for DMGRFMT

```
          CALL 'ISICALL'
              USING DMGRFMT
                    RFCB-CONTROL-BLOCK
                    MERGE-SET-EXPLICIT-LIST-1
                    MERGE-SET-EXPLICIT-LIST-2
                    MERGE-SET-INLINE-LIST
                    MERGE-SET-RECORD.
```

When you first execute this call, the value DMGRFMT is replaced with %%%% (four percent signs) followed by an internal ISICALL address.

Closing the Control Block

The final call to DMGRFMT must specify, as the only parameter, either:

- a five-byte string containing the word CLOSE.

-or-

- a Control Block with the **CLOSE-ALL** command.

This causes DMGRFMT to close the VRF by passing a parameter of CLOSE to DMGVRFWR.

CAUTION!

If the VDR erroneously supplies a second parameter, DMGRFMT searches the Rulebase Library File for a Rulebase Library named CLOS and attempts to create a Merge Set if a Rulebase Library by that name is found.

DMGRFMT Input Parameters

DMGRFMT receives input from the VDR. The following five types of parameters can be passed from the VDR to DMGRFMT the number of times indicated:

- **RFCB**
  DMGRFMT Control Block æ required (one only), and must be coded first
- **MSR1**
  Merge Set Record (variable data for tags) æ none or one, coded in any sequence
- **EXP1**
  Explicit Forms List (Standard) æ none to no limit of occurs, coded in any sequence
- **EXP2**
  Explicit Forms List (Override) æ none to no limit of occurs, coded in any sequence
- **INL1**
  Inline Forms List æ none to no limit of occurs, coded in any sequence

NOTE

The maximum number of parameters that can be passed to DMGRFMT (including the RFCB) is 25.

CAUTION!

Make sure that all RFCB fillers specify **VALUE SPACES**.

Input Parameter Format with Documerge Version 2

Documerge 3.x supports the DMGRFMT input parameter format from version 2. However, the flexibility and features of Documerge 3.x are not available with the version 2 format.

Oracle recommends that you use the Documerge 3.x format when you write new VDRs.

DMGRFMT Parameter Syntax

The syntax in which DMGRFMT input parameters are coded depends on the version of Documerge that you use. If the value of the RFCB-VERSION field in the DMGRFMT control block is less than **030000**, the syntax is the same as that of Version 2.

However, if RFCB-VERSION value is **030000** (indicating Documerge 3.0) or higher, you must use the DMGRFMT keyword format.

DMGRFMT Keyword Formats

Documerge 3.x VDR parameters must begin with one of the following keywords:

Keyword	Meaning
RFCB	**Required**. Must be the first parameter coded in a VDR. Identifies entries in the DMGRFMT Control Block. You can code only one RFCB parameter.
MSR1	Optional. Identifies the Merge Set Record, which contains variable data for tags. You can code only one MSR1 parameter in a VDR. As long as it follows the RFCB parameter, you can code the MSR1 parameter in any order.
EXP1	Optional. Identifies forms in an Explicit Forms List (EXP1). You can code more than one Explicit Forms List (EXP1) in a VDR, either with or without an Explicit Forms List (EXP2). As long as it follows the RFCB parameter, you can code EXP1 in any order. For forms with a common DTN, the order in the VDR and the sequence of entries within the list determine the printing order of the listed forms. For different DTNs, the Structure Rule determines the printing order.
EXP2	Optional. Identifies forms in an Explicit Forms List (EXP2). These entries can override DTNs in an EDL and Print Options in a Structure Rule. You can code more than one Explicit Forms List (EXP2) in a VDR, either with or without an Explicit Forms List (EXP1). As long as it follows the RFCB parameter, you can code EXP2 in any order. For forms with a common DTN, the order in the VDR and the sequence of entries within the list determine the printing order of the listed forms. For different DTNs, the Structure Rule determines the printing order.
INL1	Optional. Identifies forms in an Inline Forms List. You can code more than one Inline Forms List in a VDR. As long as it follows the RFCB parameter, you can code INL1 in any order. For forms with a common DTN, the order in the VDR and the sequence of entries within the list determine the printing order of the listed forms. For different DTNs, the Structure Rule determines the printing order.
NULL	Optional. Use to tell DMGRFMT to ignore a parameter. You can code NULL in place of any parameter but the RFCB, which is required. Makes it easier to pass a variable number of parameters to DMGFRMT

The DMGRFMT Control Block (RFCB)

```
01  RFCB-CONTROL-BLOCK.
*******************************************************
*                                                     *
*             Control Block Identification            *
*              for Documerge  3.2 DMGRFMT             *
*                                                     *
*******************************************************
     03  RFCB-ID                      PIC X(4)  VALUE 'RFCB'.
     03  FILLER                       PIC X(4).
     03  RFCB-VERSION                 PIC X(6)  VALUE '030100'.
     03  FILLER                       PIC X(2)  VALUE SPACES.
*
     03  RFCB-RETURN-CODE             PIC S9(9) COMP.
     03  RFCB-REASON-CODE             PIC S9(9) COMP.
*
*******************************************************
*                                                     *
*               Processing  Information               *
*                                                     *
*******************************************************
     03  RFCB-COMMAND                 PIC X(8).
     03  RFCB-RULEBASE-NAME           PIC X(30).
     03  RFCB-RULEBASE-RLVL           PIC S9(4) COMP.
     03  RFCB-DATA-BUFFER-ADDRESS     PIC S9(9) COMP.
     03  RFCB-VRF-DDNAME              PIC X(8)  VALUE 'DMGVRF1'.
     03  RFCB-EFFECTIVE-DATE          PIC X(6).
     03  RFCB-MAX-MESSAGE-LEVEL       PIC X         VALUE 'I'.
     03  RFCB-VRF-ALLOC-DDNAME        PIC X(8)  VALUE 'DMGVRFA'.
     03  RFCB-ALT-EFFECTIVE-DATE      PIC X(8)  VALUE —        '.
     03  FILLER                       PIC X(1).
*******************************************************
*                                                     *
*                 Processing  Flags                   *
*                                                     *
*******************************************************
     03  RFCB-SWITCHES.
         05  RFCB-SUPPRESS-SYSPRINT       PIC X     VALUE 'N'.
         05  RFCB-WRITE-RFCB              PIC X     VALUE 'N'.
         05  RFCB-WRITE-EXPLICIT-FORMS    PIC X     VALUE 'N'.
         05  RFCB-SUPPRESS-TAG-TABLE      PIC X     VALUE 'N'.
         05  RFCB-ALTERNATE-PARM-LIST     PIC X     VALUE 'N'.
         05  RFCB-NO-EFF-DATE-MSG         PIC X     VALUE 'E'.
         05  RFCB-END-MERGESET            PIC X     VALUE 'Y'.
         05  RFCB-ALLOW-MISSING-RULEBASE  PIC X     VALUE 'N'.
         05  RFCB-SUPPRESS-IMPLICIT-FORMS PIC X     VALUE 'N'.
         05  FUTURE-SWITCH-10             PIC X     VALUE — '.
     03  FILLER                       PIC X(46) VALUE SPACES.
*******************************************************
*                                                     *
*                Missing Forms Lists                  *
*                                                     *
*******************************************************
     03  RFCB-MISSING-IMPLICIT-LEN    PIC S9(9) COMP VALUE +10.
     03  RFCB-MISSING-EXPLICIT-LEN    PIC S9(9) COMP VALUE +10.
     03  RFCB-MISSING-OVERLAY-LEN     PIC S9(9) COMP VALUE +10.
     03  RFCB-REJECTED-INLINE-LEN     PIC S9(9) COMP VALUE +10.
     03  RFCB-MISSING-IMPLICIT-FORMS  OCCURS 10 TIMES.
         05 RFCB-MISSING-IMP-FORM     PIC X(32).
         05 RFCB-MISSING-IMP-RLVL     PIC S9(4) COMP.
     03  RFCB-MISSING-EXPLICIT-FORMS  OCCURS 10 TIMES.
         05 RFCB-MISSING-EXP-FORM     PIC X(32).
         05 RFCB-MISSING-EXP-RLVL     PIC S9(4) COMP.
     03  RFCB-MISSING-OVERLAY-FORMS   OCCURS 10 TIMES.
         05 RFCB-MISSING-OVL-FORM     PIC X(32).
         05 RFCB-MISSING-OVL-RLVL     PIC S9(4) COMP.
     03  RFCB-REJECTED-INLINE-FORMS   OCCURS 10 TIMES.
          05 RFCB-REJECTED-INL-FORM    PIC X(32).
```

Descriptions of DMGRFMT Control Block Fields

Listed below in the quick reference, and on the following pages are the DMGRFMT control block fields in "The DMGRFMT Control Block (RFCB)" on page 218.

DMGRFMT Control Block Fields Quick Reference

DMGRFMT Control Block Fields	Fields Set by VDR	Fields Returned by DMGRFMT
RFCB-ID	*	
RFCB-VERSION	*	
RFCB-RETURN-CODE		*
RFCB-REASON-CODE		*
RFCB-COMMAND	*	
RFCB-RULEBASE-NAME	*	
RFCB-RULEBASE-RLVL	*	
RFCB-DATA-BUFFER-ADDRESS	*	
RFCB-VRF-DDNAME	*	
RFCB-EFFECTIVE-DATE	*	
RFCB-MAX-MESSAGE-LEVEL	*	
RFCB-VRF-ALLOC-DDNAME	*	
RFCB-SUPPRESS-SYSPRINT	*	
RFCB-WRITE-RFCB	*	
RFCB-WRITE-EXPLICIT-FORMS	*	
RFCB-SUPPRESS-TAG-TABLE	*	
RFCB-MISSING-IMPLICIT-LEN	*	
RFCB-ALTERNATE-PARM-LIST	*	
RFCB-NO-EFF-DATE-MSG	*	
RFCB-MISSING-EXPLICIT-LEN	*	
RFCB-MISSING-OVERLAY-LEN	*	
RFCB-REJECTED-INLINE-LEN	*	
RFCB-MISSING-IMPLICIT-FORMS		*
RFCB-MISSING-EXPLICIT-FORMS		*
RFCB-MISSING-OVERLAY-FORMS		*
RFCB-REJECTED-INLINE-FORMS		*

DMGRFMT Control Block Fields

Field	Value
RFCB-ID	This field must contain the four byte control block identifier, RFCB, which DMGRFMT uses to determine if a valid control block parameter was passed. This information is sent to the VDR.
RFCB-VERSION	This field must contain the 6-byte number of the Documerge version you are using. This information is sent to the VDR. **Note:** When migrating to a newer release of Documerge, (i.e., release 3.2) you do not have to change the RFCB-VERSION to that of the new release; however, if you do not change the value of the RFCB-VERSION to the new value (i.e., 030200) you will not be able to use the RFCB-VERSION dependent new features offered for that release. If the value is less than **030000** any values in the following fields are ignored and the defaults are used: ▪ RFCB-MAX-MESSAGE-LEVEL ▪ RFCB-VRF-ALLOC-DDNAME ▪ RFCB-ALTERNATE-PARM-LIST If the value is less than **020100** any values in the following fields are ignored and the defaults are used: ▪ RFCB-WRITE-RFCB ▪ RFCB-WRITE-EXPLICIT-FORMS ▪ RFCB-SUPPRESS-TAG-TABLE ▪ RFCB-NO-EFF-DATE-MSG
RFCB-RETURN-CODE	This fullword field contains the return code from DMGRFMT. *Do not set this field.* This field is returned by DMGRFMT.
RFCB-REASON-CODE	This fullword field contains the reason code from DMGRFMT. *Do not set this field.* This field is returned by DMGRFMT.
RFCB-COMMAND	Contains the command which is to be executed. This field is set by the VDR. Valid commands are: ▪ **MERGESET** specifies that a Merge Set is created using the buffer area and the Rulebase specified in the control block. ▪ **CLOSEDD** specifies that a single file identified in the RFCB-VRF-DDNAME field is to be closed. ▪ **CLOSEALL** specifies that all open files opened by DMGRFMT are to be closed. The value in the RFCB-VRF-DDNAME field of the control block is ignored.
RFCB-RULEBASE-NAME	Contains the 30 character name of a Rulebase Table used to generate the VRF data for this Merge Set. This field is set by the VDR.
RFCB-RULEBASE-RLVL	Contains the VLAM revision level for the Rulebase member in the RFCB-RULEBASE-NAME field. This field is set by the VDR. A value of zero (0) automatically gets the highest revision of the Rulebase.
RFCB-DATA-BUFFER-ADDRESS	Contains the address of the buffer which contains the variable data. This buffer is mapped to the Tag table in the Rulebase. This field is set by the VDR. This field is ignored if you pass an MSR1 (Merge Set Record) parameter.
RFCB-VRF-DDNAME	Specifies the name of the JCL control statement corresponding to the output file (VRF) for this Merge Set. This field is set by the VDR. If this field is blank (all x'40'), the default value is used. The default value is the last valid RFCB-VRF-DDNAME. If no RFCB-VRF-DDNAME is specified previously, DMGVRF1 is used.

Field	Value
RFCB-ALT-EFFECTIVE-DATE	Supports Year 2000 processing of effective dates with a 4-digit year in yyyymmdd format (8 digits total). You set this field in the VDR, and Documerge uses the setting to select the revision levels for forms. **Important:** To use RFCB-ALT-EFFECTIVE-DATE, the RFCB-VERSION must be 030006 or higher and use the Documerge 3.x keyword format calling parameters for DMGRFMT. If both RFCB-ALT-EFFECTIVE-DATE and RFCB-EFFECTIVE-DATE have date values, Documerge uses RFCB-ALT-EFFECTIVE-DATE. If RFCB-ALT-EFFECTIVE-DATE is all blanks (x'40') or all low-values (x'00'), or if the RFCB-VERSION is less than "030006", then Documerge checks RFCB-EFFECTIVE-DATE for a valid 6-digit date. If neither RFCB-ALT-EFFECTIVE-DATE or RFCB-EFFECTIVE-DATE contains a valid date, Documerge uses the revision level requested in the Rulebase Forms Table. (An RFCB effective date overrides a revision level specification in the Implicit or Explicit Forms List.)
RFCB-EFFECTIVE-DATE	Contains a 6-digit effective date in YYMMDD format used to select the correct revision level of forms. The VDR sets this field. Processing logic for **RFCB-EFFECTIVE-DATE** is, as follows: 1 If the RFCB-ALT-EFFECTIVE-DATE field is all blanks (x'40') or all low-values (x'00'), or if the RFCB-VERSION is less than *030006*, then Documerge checks RFCB-EFFECTIVE-DATE for a 6-digit date. 2 If Documerge finds a valid 6-digit date, it checks the **YEAR2000=nn** EXEC parameter for a user-defined cut off year for Year 2000 processing. The specified cut off year is a 2-digit number that the RFCB-EFFECTIVE-DATE year must be equal to or less than for Documerge to assign a century value of **20** to it. For example, if you specify YEAR2000=**50**, then Documerge processes a value of **010120** as **01-01-2020**; and processes a value of **010180** as **01-01-1980**. For details about the YEAR2000=nn EXEC parameter, see "YEAR2000=nn" on page 196. 3 If the YEAR2000= parameter is not enabled, then • If the **RFCB-EFFECTIVE-DATE** is in the range of **51** – **99**, Documerge assumes the century is **19** and internally represents the two-digit century as **0**. For example, Documerge represents the year 97 as 097 (1997). • If the **RFCB-EFFECTIVE-DATE** is in the range of **00 – 50**, Documerge assumes the century is **20** and replaces the two-digit century with **1**. For example, Documerge internally represents the year 01 as 101 (2001).
RFCB-MAX-MESSAGE-LEVEL	Specifies the highest error-message Severity Code that can occur before the Merge Set is omitted from the VRF. This field is set by the VDR. The Severity Code is related to the Return Code. Refer to *Documerge Error Messages* for more information. Valid values are: **I** — Informational (RC=0). The Merge Set is not omitted from the VRF unless a message of level W (RC=4) or higher occurs. **W** — Warning (RC=4). The Merge Set is not omitted from the VRF unless a message of level C (RC=8) or higher occurs. **C** — Cautionary (RC=8). The Merge Set is not omitted from the VRF unless a message of level E (RC=12) or higher occurs.

Field	Value
RFCB-VRF-ALLOC-DDNAME	Name of the DMGVRFA file that corresponds to the VRF specified in the RFCB-VRF-DDNAME field. See "" on page 252 for more information. If this value is blank (x'40') or the value of the RFCB-VERSION field is less than 030000, this field is ignored and the default value is used. The default value is the last valid RFCB-VRF-ALLOC-DDNAME. If no previous RFCB-VRF-ALLOC-DDNAME is specified, DMGVRFA is used. This field is set by the VDR. **Important:** After the first call to DMGRFMT or DMGVRFWR, you cannot change the RFCB-VRF-ALLOC-DDNAME name for a given VRF during the VDR run. The first RFCB-VRF-ALLOC-DDNAME name for the VRF remains effective for the entire VDR run, regardless of different RFCB-VRF-ALLOC-DDNAME names in subsequent calls. If you want to use the same RFCB for more than one VRF and VRFA file, set the values for both RFCB-VRF-DDNAME and RFCB-VRF-ALLOC-DDNAME before you call DMGRFMT.
RFCB-SUPPRESS-SYSPRINT	Switch that indicates whether you want to see the error messages associated with return codes of less than **12**. This field is set by the VDR. A value of **Y** (x'E8') suppresses the messages.
RFCB-WRITE-RFCB	Switch that tells DMGRFMT whether to write the DMG.RFCB tag. This field is set by the VDR. A value of **Y** (x'E8') writes the tag; any other value will not write the tag. DMG.RFCB contains the reformatter control block passed by the VDR to DMGRFMT, ensuring that the Rulebase used at VDR execution time can be identified. If RFCB-VERSION is less than 020100, this field is ignored and assumed to be **N**. If you are using the generic VDR (DMGVDRG), then *both* RFCB-WRITE-EXPLICIT-FORMS and RFCB-WRITE-RFCB should be set to **Y**. (See "DMGVDRG (Generic VDR)" on page 459.) In addition, having this tag set to **Y** can be used as an aid to debugging the VDR.
RFCB-WRITE-EXPLICIT-FORMS	Switch that tells DMGRFMT to write the DMG.EXPLICIT.FORMS tag. This field is set by the VDR. If you are using the generic VDR (DMGVDRG), then BOTH RFCB-WRITE-EXPLICIT-FORMS and RFCB-WRITE-RFCB should be set to **Y**. (See "DMGVDRG (Generic VDR)" on page 459.) In addition, having this tag set to **Y** can be used as an aid to debugging the VDR. If RFCB-VERSION is less than 020100, this field is ignored and **N** is assumed.
RFCB-SUPPRESS-TAG-TABLE	Switch that tells DMGRFMT to ignore the Rulebase Library Tag Table. This field is set by the VDR and usually is **N**. A value of **Y** (x'E8') causes DMGRFMT to ignore the Rulebase Tag table. If RFCB-VERSION is less than 020100, this field is ignored and assumed to be **N**. This field is normally used only by the program DMGVDRG. **Caution:** If you suppress the Tag table, you must pass a Merge Set record in the MSR1 parameter to specify the Merge Set ID value. The MSR1 value must be at least 35 characters in length. For details about MSR1, see "The Merge Set Record (MSR1)" on page 226.
RFCB-ALTERNATE-PARM-LIST	Switch has a default value of **space** or **N**. Setting the value to **Y** enables processing of an alternative RFCB parameter list that has been dynamically built in the VDR. **Note:** Because Oracle primarily uses the RFCB-ALTERNATE-PARM-LIST switch for internal program testing, we do not recommend that you alter its value without first consulting with our Customer Support department (214-891-6696).

Field	Value
RFCB-NO-EFF-DATE-MSG	Switch determines whether you are notified if both of the following conditions exist: • The VDR is using effective date processing for the Merge Set. • A form is specified for the Merge Set but no revision of the EDL member is effective. This field is set by the VDR. At least one revision of the EDL member must have an effective date that is earlier than or the same as the date in RFCB-EFFECTIVE-DATE. Otherwise, the form is disqualified from the Merge Set. You can use this switch to choose whether you are notified of a form disqualified because of its effective date. Valid values are: **E** — (Default value) You receive the following notification: • The messages DMGWRT435I and DMGWRT739I, reporting that the EDL member was found but no revision was effective, written to the DMGRFMT message output file ("MESSAGE" on page 378). • The DMG.MISSING.FORMS Reserved Tag, including the name of the disqualified form, written to the VRF ("DMG.MISSING.FORMS" on page 316). **I** — The disqualified form is ignored. You do not receive any notification. **W** — A warning message (DMGWRT744W) is issued, and the disqualified form is ignored. You do not receive any notification.
RFCB-END-MERGESET	Normally, when the VDR calls DMGRFMT, it completes the processing of the current Merge Set. You can set this switch to **N** to tell DMGRFMT not to end the current Merge Set. Using the **N** setting for this switch, you can program the VDR to ▪ Use multiple Rulebases with different Tag tables to assemble the data for the Merge Set. (Rulebases can just contain Tag tables and/or implicit form names.) ▪ Call DMGVRFWR after (not before) calling DMGRFMT for a Merge Set. The default value of a **space** or **Y** specifies that DMGRFMT complete the Merge Set. A value of **N** tells DMGRFMT not to complete the Merge Set. Use of this switch to end each Merge Set requires a final call to DMGRFMT with **RFCB-END-MERGESET** set to **Y**. Because DMGRFMT processes the DMG.MERGESET.ID tag in the first call for the Merge Set, the Rulebase needs to have DMG.MERGESET.ID tag defined (if it is not defined, DMGRFMT uses a default value of *DMG.MERGESET.ID TAG NOT DEFINED*). If you use the **RFCB-END-MERGESET** flag to call DMGRFMT multiple times for a single Merge Set, the second and subsequent calls need to use a Rulebase that does *not* have the DMG.MERGESET.ID tag in it. For details about the DMG.MERGESET.ID tag, see "DMG.MERGESET.ID" on page 316. A side effect of calling DMGRFMT more than once for a given Merge Set is that it writes the implicit forms for the Merge Set to the VRF each time. To prevent this, the VDR can set the **RFCB-SUPPRESS-IMPLICIT-FORMS** switch as explained in the following topic.

Field	Value
RFCB-SUPPRESS-IMPLICIT-FORMS	The default value of a **space** or **N** for this switch tells DMGRFMT to process the implicit forms from the Rulebase normally. Setting the value to **Y** tells DMGRFMT to ignore all implicit forms coded in the active Rulebase (not write them to the DMG.FLIST tag in the VRF). Using this switch, you can program the VDR to complete a Merge Set left open by a previous DMGRFMT call. Do this by either ■ Setting **RFCB-SUPPRESS-IMPLICIT-FORMS** back to N, and then making a final call to DMGRFMT — passing the forms lists and data. DMGRFMT completes the Merge Set after the processing the parameters. **-or-** ■ Calling DMGRFMT with RFCB as the only parameter and with the Rulebase name (RFCB-RULEBASE-NAME) set to all blanks or binary zeroes (low values). DMGRFMT will write the end-of-Merge Set marker to the VRF and return to the VDR.
RFCB-ALLOW-MISSING-RULEBASE	The value of the RFCB-VERSION field in the DMGRFMT Control Block must be 030100 or higher for this switch to work.) If the DMGRFMT program attempts to access a non-existing (missing) Rulebase name, you can use the **RFCB-ALLOW-MISSING-RULEBASE** switch to cause one of the following to occur: ■ Code a value of **Y** to cause DMGRFMT to issue an C-level error message but continue processing. The C-level message generates a return code (RC) of **8**, and if the RFCB-MAX-MESSAGE-LEVEL flag is set to **C** (for *cautionary*) then processing of the Merge Set will continue despite the missing Rulebase. For details, see ""RFCB-MAX- MESSAGE-LEVEL" on page 221. ■ Code a value of **N** to cause DMGRFMT to issue an E-level error message and stop processing. The E-level message generates a return code (RC) of **12** or higher.
RFCB-MISSING-IMPLICIT-LEN	This fullword field identifies the number of entries in the missing Implicit Forms Table. This field is set by the VDR. This field requires only the *number of entries*, not the actual length of the Missing Implicit Forms Table (total number of bytes). The number of entries does not have to be the same as the number of forms in the table; it can be any number from zero up. If you set this field to zero, Documerge will not generate a list of missing forms. Because the last entry in the table is followed by an entry whose value is all high value to indicate the end of the missing forms list, you should set the length to at least two (2), to get valid results (since a length of one would include only the "high values" end of table delimiter).
RFCB-MISSING-EXPLICIT-LEN	This fullword field identifies the number of entries in the missing Explicit Forms Table. This field is set by the VDR. This field requires only the *number of entries*, not the actual length of the Missing Explicit Forms Table (total number of bytes). The number of entries does not have to be the same as the number of forms in the table; it can be any number from zero up. If you set this field to zero, Documerge will not generate a list of missing forms. Because the last entry in the table is followed by an entry whose value is all high value to indicate the end of the missing forms list, you should set the length to at least two (2), to get valid results (since a length of one would include only the "high values" end of table delimiter).

Field	Value
RFCB-MISSING-OVERLAY-LEN	This fullword field identifies the number of entries in the missing Overlay Forms Table. This field is set by the VDR. This field requires only the *number of entries*, not the actual length of the Missing Overlay Forms Table (total number of bytes). The number of entries does not have to be the same as the number of forms in the table; it can be any number from zero up. If you set this field to zero, Documerge will not generate a list of missing forms. Because the last entry in the table is followed by an entry whose value is all high value to indicate the end of the missing forms list, you should set the length to at least two (2), to get valid results (since a length of one would include only the "high values" end of table delimiter).
RFCB-REJECTED-INLINE-LEN	This fullword field identifies the number of entries in the rejected Inline Forms Table. This field is set by the VDR. This field requires simply the *number of entries*, not the actual length of the Rejected Inline Forms Table (total number of bytes). The number of entries does not have to be the same as the number of forms in the table; it can be any number from zero up. If you set this field to zero, Documerge will not generate a list of rejected forms. Because the last entry in the table is followed by an entry whose value is all high value to indicate the end of the rejected inline forms list, you should set the length to at least two (2), to get valid results (since a length of one would include only the "high values" end of table delimiter).
RFCB-MISSING-IMPLICIT-FORMS	DMGRFMT places the names of Implicit Forms which did not exist in the EDL into this table. This field is returned by DMGRFMT. The number of entries in this table must match the number specified in RFCB-MISSING-IMPLICIT-LEN. The end of the table is marked with an entry of all high value (all x'FF'). A first entry of high values indicates an empty table.
RFCB-MISSING-EXPLICIT-FORMS	DMGRFMT places the names of Explicit Forms which did not exist in the EDL into this table. This field is returned by DMGRFMT. The number of entries in this table must match the number specified in RFCB-MISSING-EXPLICIT-LEN. The end of the table is marked with an entry of all high value (all x'FF'). A first entry of high values indicates an empty table.
RFCB-MISSING-OVERLAY-FORMS	DMGRFMT places the names of Overlays which did not exist in the EDL into this table. This field is returned by DMGRFMT. The number of entries in this table must match the number specified in RFCB-MISSING-OVERLAY-LEN. The end of the table is marked with an entry of all high value (all x'FF'). A first entry of high values indicates an empty table.
RFCB-REJECTED-INLINE-FORMS	DMGRFMT places the names of Inline Forms which were rejected (had invalid print options for Inline Forms) in this table. This field is returned by DMGRFMT. The number of entries in this table must match the number specified in RFCB-REJECTED-INLINE-LEN. The end of the table is marked with an entry of all high value (all x'FF'). A first entry of high values indicates an empty table.

The Merge Set Record (MSR1)

The Merge Set Record (the MSR1 parameter) is an area containing the variable data values for the current Merge Set, such as policy number, name, address, amounts, etc. All values must be EBCDIC printable data. The Rulebase Tag Table position and length refer to this area; position 1 is the first data byte.

If the value of the RFCB-VERSION field in the DMGRFMT Control Block is **030000** or higher, a filler with a value of **MSR1** must precede each entry. This MSR1 filler is not counted in determining the Rulebase Library Tag Table POS= values. The actual Merge Set data begins with the fifth character.

If the RFCB-VERSION value is less than **030000**, the filler must not be coded (the parameter starts with the actual data), and the Merge Set Record must be the fourth (positional) parameter.

Only one Merge Set Record can be passed to DMGRFMT. See "DMGRFMT Parameter Syntax" on page 217 for more information.

CAUTION!

Typically, Documerge requires a Merge Set record. If you omit the MSR1 parameter, the RFCB-DATA-BUFFER-ADDRESS parameter must contain the address of the Merge Set record. Otherwise, the VDR step may terminate abnormally.

Following is an exceptional set of conditions for which a Merge Set record isn't required:

- The Rulebase has no Tag command
- The Rulebase has no SORT for any Group
- The RFCB-SUPPRESS-TAG-TABLE value is not YES

However, if there is no MSR1 record, then there is no way to pass a Merge Set ID tag value to DMGRFMT, which can make error detection more difficult.

The Explicit Forms List (EXP1 and EXP2)

The Explicit Forms List is a table containing one or more fixed-length entries. There are two formats for the Explicit Forms List:

- EXP1 (the standard format in previous Documerge versions)
- EXP2 (the Documerge 3.x override format)

To indicate the end of a table, include a high value (x'FF') in the first byte of the next available entry. For example, if you have 20 Explicit forms, set entry 21 to high value. There is no Documerge-imposed maximum number of entries in either format.

An Explicit Forms List is required if:

- The RFCB-COMMAND field specifies MERGESET
- An Inline Forms List or a Merge Set Record is present
- The value of the RFCB-VERSION field is less than 030000 (the DMGRFMT parameters of Documerge 2.x are positional).

Even if there are no Explicit Forms, an Explicit Forms List is required if these conditions exist. Code an empty Explicit Forms List by specifying a high value (x'FF') for the first byte of the first entry.

EXP1 (Explicit Forms List)

The Explicit Forms List (EXP1) is optional. Documerge places no limit on the number of times it can occur. It can coexist with the Explicit Forms List (EXP2), or it can exist by itself. The Explicit Forms List (EXP1) can occur in the VDR in any sequence, as long as it follows the DMGRFMT Control Block.

If the value of the RFCB-VERSION field in the DMGRFMT Control Block is **030000** or higher, a filler with a value of **EXP1** must precede each table.

If the RFCB-VERSION value is less than **030000**, a filler with a value of **EXP1** must not precede each table and must be the second (positional) parameter. See "DMGRFMT Parameter Syntax" on page 217 for more information.

Format of the Explicit Forms List (EXP1)

```
      ******************************************************
      *                                                    *
      * Explicit Forms (EXP1) parameter passed to DMGRFMT  *
      *                                                    *
      ******************************************************
       01  MERGE-SET-EXPLICIT-LIST.
           03  FILLER                      PIC X(4)  VALUE 'EXP1'.
           03  EXPLICIT-ENTRY
               OCCURS 10 TIMES.
               05  EXPLICIT-FORM-NAME      PIC X(32).
               05  EXPLICIT-FORM-REV       PIC S9(4)    COMP.
```

NOTE

The bolded "10" is for example only. In fact, the table can occur from one time to as many times as your compiler and storage can handle.

The following fields must be included in the EXP1 (Explicit Forms List):

- EXP1-EXPLICIT-FORM-NAME
  The name of the Explicit Form.
- EXP1-EXPLICIT-FORM-REV
  The revision level of the Explicit Form.

Revision Level	Description
0	Indicates the highest revision level at the time of VDR processing. This revision level determines DTN values.
1-32	Indicates a specific revision level.
(-1)	Indicates the highest revision level at DMGMERGE processing time.

EXP2 (Explicit Forms List)

The Explicit Forms List (EXP2 table) is optional. Documerge places no limit on the number of times it can occur. It can coexist with the Explicit Forms List (EXP1), or it can exist by itself. The Explicit Forms List (EXP2) can occur in the VDR in any sequence, as long as it follows the DMGRFMT Control Block.

To indicate the end of the EXP2 table, include a high value (x'FF') in the first byte of the next available entry. For example, if you have 20 Explicit forms, set entry 21 to high value. There is no Documerge-imposed maximum number of entries in the table. "Format of the Explicit Forms List (EXP2)" on page 229 outlines the format of each table entry.

With this format you can create an Explicit Forms List that specifies:

- A DTN that overrides the DTN in the EDL
- Print Options that override those in the Structure Rule.

NOTE

If you want to override the DTN and Print Options for an Implicit Form, you must delete the form from the Rulebase Library Forms Table and add it to an Explicit Form List (EXP2).

If the value of the RFCB-VERSION field in the DMGRFMT Control Block is **030000** or higher, a filler with a value of **EXP2** must precede each table. If the RFCB-VERSION value is less than **030000**, this parameter is not supported. See "DMGRFMT Parameter Syntax" on page 217 for more information.

Format of the Explicit Forms List (EXP2)

```
      ************************************************************
      *                                                          *
      * Explicit Forms (EXP2) parameter passed to DMGRFMT        *
      *                                                          *
      ************************************************************
       01  MERGE-SET-EXPLICIT-LIST.
           03  FILLER                          PIC X(4)  VALUE 'EXP2'.
           03  EXPLICIT-ENTRY
               OCCURS 10 TIMES.
               05  EXPLICIT-FORM-NAME          PIC X(32).
               05  EXPLICIT-FORM-REV           PIC S9(4)    COMP.
               05  FILLER                      PIC X  VALUE SPACES.
               05  EXPLICIT-DTN-OVERRIDE       PIC X(5).
               05  EXPLICIT-OPTIONS-OVERRIDE.
                   07  EOO-DUP-SIM-TUM         PIC X(4).
                   07  EOO-SEP-CON             PIC X(4).
                   07  EOO-MAI-AUX             PIC X(4).
                   07  EOO-STA-OFF             PIC X(4).
                   07  EOO-ODD-EVN-ANY         PIC X(4).
                   07  EOO-POR-LAN             PIC X(4).
                   07  EOO-ON-OFF-OVL          PIC X(4).
                   07  EOO-USER1               PIC X(4).
                   07  EOO-USER2               PIC X(4).
                   07  EOO-USER3               PIC X(4).
                   07  EOO-USER4               PIC X(4).
                   07  EOO-USER5               PIC X(4).
                   07  EOO-BSS-CSS             PIC X(4).
                   07  EOO-LOGICAL-PAGE        PIC X(4).
                   07  FILLER                  PIC X(24)  VALUE SPACES.
```

NOTE

The bolded "10" is for example only. In fact, the table can occur from one time to as many times as your compiler and storage can handle.

The following fields must be included in the EXP2 (Explicit Forms List):

- **EXPLICIT-FORM-NAME**
  The name of the Explicit Form.
- **EXPLICIT-FORM-REV**
  The revision level of the Explicit Form.

Revision Level	Description
0	Indicates highest revision level at the time of VDR processing.
1-32	Indicates a specific revision level.
(-1)	Indicates the highest revision level at DMGMERGE processing time.

The highest revision level at VDR processing time determines DTN values.

- **EXPLICIT-DTN-OVERRIDE**
  The DTN that overrides the Explicit Form's DTN in the EDL. This value must be 5 numeric characters, with leading zeroes if necessary. If you want to use the DTN from the EDL, code spaces in this field.
- **EXPLICIT-OPTIONS-OVERRIDE**

  The values that override the Print Options specified for the form in the Structure Rule. Each value relates to an Option Field in the DMG.OPT.*groupname* Reserved Tag.

  Each override field is positional. Therefore, you must enter a blank value (x'40') if you do not want to override one of the related Print Options.

  Several Print Options are available for each of these fields. You can use only one of the available Print Options to override the active Print Option in the Structure Rule. You enter the selected 3-character abbreviation in the field.

  The DMGRFMT subprogram verifies that the values entered in these fields are valid Print Options. If so, DMGRFMT places these override values into the DMG.OPT.*groupname* Reserved Tag in the VRF.

  Override Print Options in the DMG.OPT.*groupname* tag apply to all Groups that receive the form.

To Override a Print Option for a Single Group

1. Change the DTNs for the forms you want to override in the Structure Rule for the Group.

   Increment the DTNs by 1 if possible.
2. Use the EXP1 syntax to code each form with no DTN override and no Print Option override.
3. Use the EXP2 syntax to code each form with a DTN override and the Print Option overrides.

   Code the DTN override with the new DTN you assigned in step 1.

Because the forms have different DTNs only for the specific Group, the related Print Option overrides apply only to that Group. (The override DTNs must match the DTNs in the Structure Rule for the Group.)

The table "EXPLICIT-OPTIONS-OVERRIDE Fields" on page 231 describes the EXPLICIT-OPTIONS-OVERRIDE fields displayed in "Format of the Explicit Forms List (EXP2)" on page 229.

EXPLICIT-OPTIONS-OVERRIDE Fields			
Field	**Values**	**Print Options**	**Meaning**
EOO-DUP-SIM-TUM	DUP	"DUPlex" on page 173	The page parity that applies to all pages of the form when Documerge goes to a new page. Used with the EOO-ODD-EVN-ANY field.
	SIM	"SIMplex" on page 176	
	TUM	"TUMble" on page 177	If the PRINTDEF specifies SIMPLEX, this option is ignored and SIM is assumed. You cannot override these options if IMPDEF is coded for this form in the Structure Rule.
	blank	No override	
NOTE: You cannot override the IMPDEF (Imposition Definition) Print Option, nor can you override any other Print Option with IMPDEF.			
EOO-SEP-CON	SEP	"SEParate" on page 176	Indicates if this form is concatenated with the previous form or begins on a separate page.
	CON	"CONcatenate" on page 173	
	CKP	Concatenate and Keep with Previous form if possible, page 173	The **CKP** override keeps selected specific explicit forms together. After coding the first form, code CKP for the second and subsequent forms that must be kept on the same page.
	CNK	Concatenate NO Keep	The **CNK** override specifies that the print option not be changed to CKP, even if there is a KEEPLAST value specified for the form in the Rulebase.
	blank	No override	
EOO-MAI-AUX	MAI	"MAIn=" on page 176	For Xerox printers, selects the input tray for the paper. Documerge uses this value only if this form is the first form of a new sheet. **You cannot override these options if CLUSTER= or FEED= is specified for the DTN in the Structure Rule**.
	AUX	"AUXiliary" on page 172	
	SRC	"FEED=" on page 174 and "CLUSTER=" on page 172	Because SRC gets the name for the next cluster from the DMG.SRC.*Groupname* tag, and that name must agree with the name in the DMG.OPTS.*Groupname* tag, **you cannot change MAI or AUX to SRC, nor can you change SRC to MAI or AUX**.
	PRV	(Previous)	PRV causes the last cluster name specified to be reused. Because PRV, MAI, AUX don't use tag values, you can change MAI or AUX to PRV, and you can change PRV to MAI or AUX.
	blank	No override	
EOO-STA-OFF	STA	"STAck" on page 177	Indicates if the output is to be offset from the previous page. Documerge uses this value only if this is the first form of a new sheet.
	OFF	"OFFset" on page 176	
	blank	No override	

EXPLICIT-OPTIONS-OVERRIDE Fields			
Field	**Values**	**Print Options**	**Meaning**
EOO-ODD-EVN-ANY	ODD EVN ANY *blank*	"ODD" on page 176 "EVeN" on page 174 "ANY" on page 172 No override	Additional page-parity specifications for the form. Used with the EOO-DUP-SIM-TUM field.
	Combine one of these values with one of the EOO-DUP-SIM-TUM values, specifying the EOO-DUP-SIM-TUM value first. Valid combinations and their results are:		
	DUP-SIM-TUM	**ODD-EVN-ANY**	**Result**
	DUP	ODD	First page of form on front of a new sheet. If a multiple-page form, page 2 on back side of same sheet. Page 3 on front of a new sheet, etc.
		EVN	First page of form on first new back available, which can be a new sheet. If a multiple-page form, page 2 on front side of new sheet, page 3 on back side of same sheet, page 4 on front side of new sheet, etc.
		ANY	Next page (first available front or back). If a multiple-page form, remaining pages on next available pages.
	SIM	ODD	All pages on front side of new sheets.
		EVN	All pages on back sides of new sheets.
		ANY	All pages on front side of new sheets.
	TUM	ODD	Same result as DUP ODD, except pages on all back sides are flipped upside down.
		EVN	Same result as DUP EVN, except pages on all back sides are flipped upside down.
		ANY	Same result as DUP ANY, except pages on all back sides are flipped upside down.
EOO-POR-LAN	POR LAN *blank*	"PORtrait" on page 176 "LANdscape" on page 176 No override	Indicates page orientation, the direction of concatenation, and Overlay shifting. Generally, this is the same as the way the form itself was composed. Portrait means "Short edge is at the top and bottom".
EOO-ON-OFF-OVL	ON OFF	"OVL=( )" on page 176 Turns off OVerLay=( )	The current active DTN-level Overlays print on the same pages as this form. The current active DTN-level Overlays do not apply to this form.

EXPLICIT-OPTIONS-OVERRIDE Fields

Field	Values	Print Options	Meaning
		Note: Package-level Overlays print regardless of this ON/OFF value. Once a Document Package-level Overlay is defined, it remains active for the remainder of the Document Package. For further information about Overlays and Overlay coding options, see "Documerge Overlay Forms" on page 465.	
	abc	Each consecutive occurrence of this *abc* format adds this Overlay name to the active list of Overlays. When an ON or OFF option is found, the next *abc* format adds to the list of active Overlays This 3-character value is a combination of 1-character values. For each position in this value, several 1-character values are possible.	
		Position	**Description**
		a	For a **DTN-level** Overlay with the *appliesto* **PHYSICAL** print option, indicate the page parity with one of the following values: *O* Both front (odd-numbered) and back (even-numbered) pages **F** Front side (odd-numbered) only **B** Back side (even-numbered) only
			For a **DTN-level** Overlay with the *appliesto* **LOGICAL** print option, indicate the page parity with one of the following values: **A** Both front (odd-numbered) and back (even-numbered) pages **D** Front side (odd-numbered) only **C** Back side (even-numbered) only
			For a **PACKAGE-level** Overlay with the *appliesto* **PHYSICAL** print option, indicate the page parity with one of the following values: **P** Both front (odd-numbered) and back (even-numbered) pages **R** Front side (odd-numbered) only **Q** Back side (even-numbered) only

EXPLICIT-OPTIONS-OVERRIDE Fields			
Field	**Values**	**Print Options**	**Meaning**
			For a **PACKAGE-level** Overlay with the *appliesto* **LOGICAL** print option, indicate the page parity with one of the following values: **J** Both front (odd-numbered) and back (even-numbered) pages **M** Front side (odd-numbered) only **K** Back side (even-numbered) only
			For a **PACKAGE-level** Overlay with the *appliesto* **SHEET** print option, indicate the page parity with one of the following values: **S** Both front (odd-numbered) and back (even-numbered) pages **U** Front side (odd-numbered) only **T** Back side (even-numbered) only
		b	**V** Indicates this is an Overlay form

EXPLICIT-OPTIONS-OVERRIDE Fields			
Field	**Values**	**Print Options**	**Meaning**
		c	For the *when* and *where* print options, indicate when and where the Overlay prints with one of the following values: **B** ALWAYS (*when*) and FIRST (*where*) **C** ALWAYS (*when*) and LAST (*where*) **D** ALWAYS (*when*) and NOTFIRST (*where*) **E** ALWAYS (*when*) and NOTLAST (*where*) **F** ALWAYS (*when*) and MIDDLE (*where*) **A** ALWAYS (*when*) and ALL (*where*) **H** ONLYBLANK (*when*) and FIRST (*where*) **I** ONLYBLANK (*when*) and LAST (*where*) **J** ONLYBLANK (*when*) and NOTFIRST (*where*) **K** ONLYBLANK (*when*) and NOTLAST (*where*) **M** ONLYBLANK (*when*) and MIDDLE (*where*) **O** ONLYBLANK (*when*) and ALL (*where*) **Q** NONBLANK (*when*) and FIRST (*where*) **R** NONBLANK (*when*) and LAST (*where*) **S** NONBLANK (*when*) and NOTFIRST (*where*) **T** NONBLANK (*when*) and NOTLAST (*where*) **U** NONBLANK (*when*) and MIDDLE (*where*) **N** NONBLANK (*when*) and ALL (*where*) **W** DEFAULT (*when*) and FIRST (*where*) **X** DEFAULT (*when*) and LAST (*where*) **Y** DEFAULT (*when*) and NOTFIRST (*where*) **Z** DEFAULT (*when*) and NOTLAST (*where*) **G** DEFAULT (*when*) and MIDDLE (*where*) **L** DEFAULT (*when*) and ALL (*where*)
	blank	No override	

EXPLICIT-OPTIONS-OVERRIDE Fields			
Field	**Values**	**Print Options**	**Meaning**
EOO-USER1	US1	"USer1 to USer5" on page 177	Selects COPYGROUP1 (AFP) or COPYMOD1 (Metacode) from the PRINTDEF.
	XXX	Turns off USer1	
	blank	No override	
EOO-USER2	US2	"USer1 to USer5" on page 177	Selects COPYGROUP2 (AFP) or COPYMOD2 (Metacode) from the PRINTDEF.
	XXX	Turns off USer2	
	blank	No override	
EOO-USER3	US3	"USer1 to USer5" on page 177	Selects COPYGROUP3 (AFP) or COPYMOD3 (Metacode) from the PRINTDEF.
	XXX	Turns off USer3	
	blank	No override	
EOO-USER4	US4	"USer1 to USer5" on page 177	Selects COPYGROUP4 (AFP) or COPYMOD4 (Metacode) from the PRINTDEF.
	XXX	Turns off USer4	
	blank	No override	
EOO-USER5	US5	"USer1 to USer5" on page 177	Selects COPYGROUP5 (AFP) or COPYMOD5 (Metacode) from the PRINTDEF.
	XXX	Turns off USer5	
	blank	No override	
EOO-BSS-CSS	BSS	"ESS" on page 174	Begins or continues a subset.
	CSS	"ESS" on page 174	Although the Structure Rule uses "ESS" (for End-of-Subset), the actual options in the VRF use values that mean "begin a new subset" and "continue the current subset".
	blank	No override	
EOO-LOGICAL-PAGE	PLP	(See "LPGDEF Command" on page 154)	Forces a new SSI and continues the current LPGDEF values.
	blank	No override	If the current Option Field in the DMG.LPG.*groupname* Reserved Tag is not CLP, any value (valid or invalid) in this field is ignored. An error message is not generated.
			If this field contains an invalid value and the current Option Field is CLP, DMGMERGE generates an error message.

The Inline Forms List (INL1)

An Inline form is data in line-printer format that is passed directly from the VDR, rather than from an EDL. The INL1 (Inline forms) List is required if RFCB-COMMAND is set to MERGESET. The INL1 List is a table containing one or more fixed-length entries.

NOTE

Override is not available for DTNs and Print Options of Inline forms.

If the value of the RFCB-VERSION field in the DMGRFMT Control Block is **030000** or higher, a filler with a value of **INL1** must precede each entry.

If the RFCB-VERSION value is less than **030000**, the filler must not precede the entry and must be the third (positional) parameter. See "DMGRFMT Parameter Syntax" on page 217 for more information.

Format of the Inline Forms List

```
      ********************************************************
      *                                                      *
      *      Inline Forms parameter passed to DMGRFMT        *
      *                                                      *
      ********************************************************
       01  MERGE-SET-INLINE-LIST.
           03  FILLER                      PIC X(4) VALUE 'INL1'.
           03  INLINE-ENTRY
               OCCURS 10 TIMES.
               05  INLINE-TAG-NAME        PIC X(30).
               05  INLINE-FORM-DTN        PIC X(5).
               05  INLINE-LINEEND         PIC X.
```

NOTE

The bolded "10" is for example only. In fact, the table can occur from one time to as many times as your compiler and storage can handle.

The following fields must be included in each Inline Forms List:

- **INLINE-TAG-NAME**
  A unique, 30 character Documerge Tag name padded with blanks if necessary. The INLINE-TAG-NAME can be derived from one of two areas:
  - Written to the VRF by the user with a call to DMGVRFWR (see "The DMGVRFWR Subprogram" on page 240 for more information)
  - Written as a tag in the Rulebase with data in the MERGE-SET-RECORD area.

  For each Merge Set, all calls to DMGVRFWR must precede calls to DMGRFMT.
- **DOCUMENT-TYPE-NUMBER**
  A numeric field from 000 to 99999 that assigns a Document Type Number(DTN) to the inline form. The DTN may be identical to other (Implicit or Explicit) forms' DTNs, as long as it was specified in the Rule Base structure for the inline form. The DTN requires leading zeros and must be a decimal integer in character (EBCDIC) format.
- **LINE-END-CHAR**
  A logical line-end character that DMGMERGE uses to break the stream of tagged data into lines on a form. Anytime DMGMERGE detects this character during the merging process, it breaks the current line and begins a new one. Any character, including low and high values, that does not appear in the print lines may be designated a line-end character.

Inline Forms

An Inline form is data in line-printer format, with a single fixed-pitched font and carriage controls. (Font indexing lets you use multiple fonts, but this is an advanced and complicated technique.) You don't store an Inline form in an EDL. The VDR writes an Inline form to the VRF as a tag and data.

As with all VRF tags, you can code an Inline form in the Rulebase Library Tag Table or in a VDR call to the DMGVRFWR subprogram. Inline forms are optional. You can code as many as you want.

You must code each Inline form in the Inline Forms List (INL1) in the VDR (see page "Format of the Inline Forms List" on page 237). Each INL1 list contains:

- The name of the 30-character tag that contains the Inline form data (padded with spaces if necessary).

  Inline forms within a Merge Set must have unique names. If the same name is used, the effect is similar to omitting a DELETE=Y parameter from a repetitively used BPSD. (DELETE=Y is not implied with Inline forms.)

- The line end character for that form (can be any character that is not in the print data — try **** (backslash).
- The DTN (Document Type Number) of the form (required here since the form is not a member of an EDL).

  Oracle recommends that you give Inline forms their own DTNs.

The Inline Forms List ends with a High Value. An empty INL1 list contains only a High Value.

DMGRFMT looks up the DTNs in the forms tables and locates each Inline form in its proper sequence with the other forms.

You cannot use the following Documerge features with Inline forms:

- Concatenation (CON Print Option)
- Imposition printing (IMPDEF command)
- Tumble printing (TUM Print Option)

You can use Inline forms to insert dynamic (line printer) data that can change from one Document Package to another, such as the declaration pages of policies.

Format of Inline Forms

Inline Forms use the format of line printer data (regardless of the type of printer you are actually using). Line printer data consists of a single fixed-pitch font (EBCDIC) and carriage control characters.

The carriage control can be either or both of the two standard formats, ASA or Machine.

In addition, Inline Forms require a (user defined) line-end character.

Code the form according to the following structure:

```
carriage control character / data (print line one) / line end character
carriage control character / data (print line two) / line end character
carriage control character / data (print line three) / line end character
carriage control character / data (print line four) / line end character
etc.
```

Blanks following the line end character belong to the next line. (The first blank is interpreted as a single space carriage control.) You can include blanks *before* the line end characters, in order to construct a fixed length table.

Writing the Inline Form Tag

The VDR builds the Inline Form data and writes the data to the VRF. As with any other VRF tag, there are two ways to write the Inline Form tag:

(1) Using the Rulebase Library Tag Table

The VDR builds the Inline Form data in the Merge Set Record as it does for any other VRF tag.

(2) Using the DMGVRFWR subprogram

The VDR calls DMGVRFWR before calling DMGRFMT to write the Inline Form tagged data.

Different Groups and different Inline Form Table entries can use the same Inline Form tagged data. However, an Inline Form can be referenced only by its associated Merge Set. Therefore, if an Inline Form is used by several Merge Sets, the Inline Form tagged data must be written to the VRF once for each Merge Set.

Sequence of Forms

Documerge determines the sequence, or printing order, of the forms in a given Group by constructing an internal table of the eligible forms and their DTNs. Previous Documerge versions built the table of eligible forms according to a sequence based on the form type:

(1) Implicit

(2) Explicit

(3) Inline

Documerge 3.x offers increased flexibility in the form sequence. Now, the form sequence is determined by the order of the form lists in the VDR. EXP1 (Explicit Forms List), EXP2 (Explicit Forms List), and INL1 (Inline Forms List) can occur in any order.

After Documerge builds the internal table, it is no longer concerned with the types of the forms. Documerge now gets the appropriate Structure Rule from the Rulebase. For the first DTN in the Structure Rule, Documerge searches its entire table of forms, pulling out every one that matches. For the second DTN from the Structure Rule, Documerge pulls out each form with the same DTN, and so on.

The DMGVRFWR Subprogram

The DMGVRFWR subprogram writes the sequential VRF. It is usually called by the DMGRFMT subprogram, but must be called directly by your VDR when writing Inline Forms. Due to the structure of the VRF data, it is necessary that this subprogram be used to write the actual data records.

Using an optional parameter to DMGVRFWR, DDNAME, it is possible to specify variable DD names for output files. This parameter can be appended to any call to DMGVRFWR.

Also, you can specify the name of the VRF allocation file. Refer to "" on page 252.

Creating Tags with DMGVRFWR

You can use DMGVRFWR to bypass the Rulebase by writing tags, including inline forms, directly to the VRF. (Inline forms may also be included in a tag in the Rulebase.) Although using DMGVRFWR is *never* required, it might improve throughput efficiency with a complex VDR involving, for instance, optional tags.

NOTE

Using DMGVRFWR to write a tag to the VRF should be considered an advanced Documerge technique, one that requires a good working understanding of the format of a VRF. Most Documerge applications do not require using DMGVRFWR.

The procedure varies, depending on whether you use DMGRFMT.

To Write Tags with DMGRFMT (or DMGFORMT)

1. Before calling DMGRFMT, call DMGVRFWR as many times as needed to write out all tags not included in the Rulebase.
2. Call DMGRFMT to end the Merge Set.

To Write Tags with DMGRFMT (or DMGFORMT)

1. For each Merge Set, call DMGVRFWR for each tag in the Merge Set, including any Reserved Tags (especially DMG.FLST.*groupname* and DMG.OPT.*groupname*).
2. For each Merge Set, call DMGVRFWR to end the Merge Set.
3. At the end of the VDR, call DMGVRFWR to close all Merge Sets.

See "Bypassing the Rulebase Library" on page 244 for more information.

IMPORTANT!

The prefix DMG. identifies Documerge Reserved Tags. Oracle reserves this convention for current and future Documerge Reserved Tags.

Code the prefix **DMG.** for a tag name only if you are coding one of the Documerge Reserved Tags documented in Chapter 8.

Calling DMGVRFWR

A call to DMGVRFWR creates the tagged data entry for one tag. DMGVRFWR must be called using the subprogram ISICALL. Included in the Documerge LOADLIB, this Oracle-written program dynamically calls other subprograms. Call DMGVRFWR using the calling sequence below.

DMGVRFWR calling syntax

```
CALL 'ISICALL'
    USING DMGVRFWR
          DATA-TAG-NAME
          ACTUAL-DATA
          ACTUAL-DATA-LENGTH
          DDNAME.                (optional)
```

DMGVRFWR Parameters

Use the following parameters when calling DMGVRFWR:

- **DATA-TAG-NAME**
  The variable data tag name to be associated with the variable data that follows. This value has a maximum 30 characters and no blanks within the tag name. A blank terminates the tag name.
- **ACTUAL-DATA**
  The variable data associated with the Tag name. The maximum length is 65,535 bytes.
- **ACTUAL-DATA-LENGTH**
  Indicates the length of the data passed in ACTUAL-DATA. This is packed data with a length of 3 (i.e.: PIC S9(5) COMP-3). The range is zero to 65535.
- **DDNAME**
  An optional field you can use if your VDR creates multiple VRFs. You use this field to specify either:
  - The name of the VRF (from the JCL control statement).

    This field is 8 bytes long. The default VRF name is DMGVRF1.
  - The name of the VRF (from the JCL control statement) *and* the name of the VRF's DMGVRFA file.

    This field is 17 bytes long. It begins with a slash (*/*).

    The 8-byte VRF file name follows the slash. The default VRF name is the last valid VRF file name passed to DMGVRFWR. If no VRF file name was passed, the default is DMGVRF1.

    The 8-byte name of the VRF allocation file follows the VRF name. The default name is the last valid VRF allocation file name passed to DMGVRFWR. If no VRF allocation file name was passed, the default is DMGVRFA.

NOTE

Oracle recommends that you use a unique DMGVRFA file name for each VRF, and that you specify both when you call DMGRFMT and DMGVRFWR.

Refer to "" on page 252 for more information.

If the DDNAME field is omitted, DMGVRFWR uses the last specified DDNAME (which could be the last one passed by DMGRFMT). If no DDNAME has been specified, the program uses DMGVRF1.

If (and *only* if) you are bypassing the Rulebase Library (that is, you are not calling DMGRFMT or DMGFORMT), be sure to end the Merge Set ("Ending a Merge Set" on page 242) and close the VRF ("Closing a VRF" on page 242).

Ending a Merge Set

```
CALL 'ISICALL'
    USING 'DMGVRFWR'
          'END'
           DDNAME
```

If DDNAME is omitted, the last-referenced VRF will be ended (by default, this is DMGVRF1).

Closing a VRF

```
 CALL 'ISICALL'
     USING 'DMGVRFWR'
           'CLOSE'
           DDNAME.              (optional)
```

NOTE

If DDNAME is omitted, the last referenced VRF will be closed (by default, DMGVRF1). This could be the last VRF passed by DMGRFMT. If DDNAME = *ALL, all VRFs will be closed.

Multiple VRFs can be created by specifying different DD names in each call to DMGVRFWR.

IMPORTANT!

Use caution when issuing multiple DD names. If you send tagged data to a file that has already been closed, DMGVRFWR issues a Return and Reason Code, but still reopens the file and writes the data to the file. **Any information previously written to the file is lost**.

DMGVRFWR writes out the tag name and data for one tagged data element in a Merge Set. It is also necessary to call DMGVRFWR to signal that a Merge Set is complete.

To end a SET if you are bypassing the Rulebase Library, a special call must be made to DMGVRFWR, passing only one parameter: a three-byte field containing the word END. DMGRFMT performs this call; the user-written reformatter program should *never* pass END to DMGVRFWR directly since DMGRFMT performs this function. When using multiple files, you may specify the DDNAME as the second parameter. Do not do this if you are using the Rulebase Library (that is, you are calling DMGRFMT or DMGFORMT).

To close the VRF at the end of the program if you are bypassing the Rulebase Library, call DMGVRFWR passing one parameter: a five-byte field containing the word CLOSE. Again, note that DMGRFMT performs this call when your program passes it the CLOSE field. Your VDR program should not pass CLOSE to DMGVRFWR directly when also using DMGRFMT. Do not do this if you are using the Rulebase Library (that is, you are calling DMGRFMT or DMGFORMT).

When using multiple files, there are two additional methods of issuing the CLOSE command:

- Once for each open file, using the DDNAME as the second parameter
- Once to close all open files, using

```
*ALL
```

as the second parameter.

The ISIFLAST Subprogram

ISIFLAST is a Oracle subprogram that locates the last non-blank character in a character string. You can use ISIFLAST if you are calling DMGVRFWR to write a tag. DMGVRFWR always writes the exact number of data characters specified, including trailing blanks. By calling ISIFLAST first, you can have DMGVRFWR exclude the trailing blanks from the data.

You pass three parameters to ISIFLAST:

(1) The character string

(2) A 3-byte packed (PIC S9(5) COMP-3) field containing the length of the character string (including any trailing blanks). This field is set by the VDR.

(3) A 3-byte packed (PIC S9(5) COMP-3) field containing the length of the character string minus any trailing blanks. This field is set by ISIFLAST. This field is zero if the entire character string is blank. You can pass this value to DMGVRFWR for the data length.

You call ISIFLAST using ISICALL. The following sample shows the calling code in COBOL:

```
77  ISIFLAST    PIC X(8)   VALUE 'ISIFLAST'.
    CALL 'ISICALL' USING ISIFLAST, CHAR-STRING, STRING-LEN, TRUNC-LEN.
```

The following sample shows the COBOL code for calling ISIFLAST followed by DMGVRFWR:

```
77  ISIFLAST    PIC X(8)   VALUE 'ISIFLAST'.
77  DMGVRFWR    PIC X(8)   VALUE 'DMGVRFWR'.
77  STRING-LEN  PIC S9(5)  COMP-3   VALUE +100.
77  TRUNC-LEN   PIC S9(5)  COMP-3.
77  TAG-NAME    PIC X(30)  VALUE 'MYTAG'.
77  CHAR-STRING PIC X(100).
    ...Build value to CHAR-STRING here...
    CALL 'ISICALL' USING ISIFLAST, CHAR-STRING, STRING-LEN, TRUNC-LEN.
    CALL 'ISICALL' USING DMGVRFWR, TAG-NAME, CHAR-STRING, TRUNC-LEN.
```

Advanced Documerge Techniques

You can use the Documerge components to produce output that is different from your standard Document Packages. The purpose of this section is to provide you with ideas and procedures for using Documerge to produce unique output that may be required at your site.

Bypassing the Rulebase Library

The Documerge Rulebase Library File simplifies the process of selecting documents and print options for the Merge Sets in executing Documerge. The documents and the print options that are used to produce them are based on the Document Type Number (DTN). One of the print options available is concatenation: the process of printing two or more short forms on a single physical page. This option is available only for documents that are in concatenated sets of DTNs.

You cannot mix concatenation and non-concatenation within the same DTN when you are using the CON print option in the Rulebase Library. This also makes it impossible to force a page break at strategic points between concatenated documents.

only for documents that share the same DTN.

You can, however, perform this action using a special type of processing that does not call the Documerge Rulebase Library. This unique processing bypasses the standard Rulebase Library and formats the tags in the Variable Replacement File.

During processing, the VDR subprogram DMGRFMT accesses the Rulebase Library to generate several Documerge reserved tags for each Merge Set. When bypassing the Rulebase Library, you must write these tags manually. The tags must be written correctly or you risk the possibility of undesirable output from DMGMERGE and possibly an ABEND situation.

Writing the VDR to Bypass the Rulebase

"Calling DMGVRFWR" on page 241 explains the procedure required for calling DMGVRFWR. Using that information and the procedure listed below, you can write a VDR that bypasses the Documerge Rulebase Library and DMGRFMT.

To Write a VDR to Bypass the Rulebase

1. Call DMGVRFWR to write the tag names and their corresponding data to the VRF.

   You must execute DMGVRFWR once for each tag that appears in the Merge Set.

2. Call DMGVRFWR to write the Documerge Reserved Tags.

   You must execute DMGVRFWR once for each of the following Documerge Reserved Tags:

   - DMG.FLST.*groupname*
   - DMG.OPT.*groupname*
   - DMG.Merge Set.ID
   - DMG.SKEY.*groupname*
   - other Reserved Tags that control Imposition printing or Logical Page Definitions

   There may be multiple sets of these tags depending on the number of Groups to be processed by DMGMERGE, with one exception. The exception is the DOCUMERGE.ID.TAG which should be written only once in the VRF.

Each occurrence of the GROUPING, OPTIONS, SORTKEY and IMPDEF tags must include the user-specified 1- to 21-character Group name for the Group processed. The SORTKEY. tag is written only when a user-defined sort key is present in the Merge Sets being processed. The IMPDEF. tag is written only when the Merge Sets being process include documents that are to be imposed.

3 After each Merge Set is written, call DMGVRFWR to END the Merge Set.

When all Merge Sets have been written, call DMGVRFWR to CLOSE the VRF. Use the procedure outlined in "Calling DMGVRFWR" on page 241.

NOTE

The tags and their data, including the Documerge Reserved Tags, may be written in any sequence. The only instances when the order in which the tags and data appear is when the DELETE=Y or MULTI parameters are used. These tags will produce the variable data in the order in which they are written in the VRF.

Notice that DTNs are not specified when the Rulebase Library is not used for processing Documerge input. Each document is associated with the default DTN value, zero (0), for the DFXUTIL LOAD function.

DMGUSER2 Input File

```
CMT                ***** INPUT FILE TO DMGUSER2 *****
CMT There are two main sections to the input data.
CMT The first is the program globals section and the second
CMT is the Merge Set section.  These block sections may be
CMT repeated as needed to change globals (e.g. - to turn
CMT trace on, exhibit input lines, etc.) or to produce
CMT multiple Merge Sets.
CMT
CMT ********************************************************************
CMT THIS IS THE BEGINNING OF THE PROGRAM GLOBALS SECTION
CMT Valid entries are CMT, TBN, RCV, DIA, RTD and EPG (EPG must be last)
BPG              (begin program globals)
DIA   66VDRMSG
TBN   YYN - 1: match begin and end blocks (Y or N); 2: exhibit input
CMT             lines (Y or N); 3: turn on/off trace
RCV     20  16  12  08  04  00 (Override return code values, entered
CMT                             in the following order: P-level,
CMT                             F-level, E-level, C-level, W-level,
CMT                             and I-level; each has four numeric
CMT                             positions.)
CMT
CMT
CMT FOLLOWING ARE THE RECORD TYPE DEFINITIONS
CMT   TYPE
CMT   -  LENGTH
CMT   -  -    OCCURS
CMT   -  -    -   SEQUENCE    COMMENTS (TAG NAMES)
CMT   TTTLLLLLOOOOSSSS
RTD     A0002000010001        NAME
RTD   SPA0000900010002        (filler)
RTD     B0000300010003        AGE
RTD   SPB0000200010004        (filler)
RTD     C0000600010005        SEX
RTD   SPC0000400010006        (filler)
RTD     D0001000010007        POLICY.NUMBER
RTD   SPD0002600010008        (filler)
RTD     E0001200010009        POLICY.DATE
RTD   SPE0000700010010        (filler)
RTD     F0001000010011        SUM.INSURED
RTD   SPF0000500010012        (filler)
RTD     G0000800010013        PREMIUM.CLASS
RTD   SPG0000700010014        (filler)
RTD     H0002200010015        BENEFIT.DESC
RTD   SPH0000900010016        (filler)
RTD     I0000400010017        TERMINATION.PERIOD
RTD   SPI0000500010018        (filler)
RTD     J0000900010019        ANNUAL.MAX.1
RTD   SPJ0000100010020        (filler)
RTD     K0000900010021        ANNUAL.MAX.2
RTD   SPK0000100010022        (filler)
RTD     L0003500010023        PREMIUM.PERIOD
RTD   SPL0001600010024        (filler)
RTD     M0000900010025        ANNUAL.PERIOD
RTD   SPM0007100010026        (filler)
RTD     N0000200010027        DMG.GCPY.INSURED
RTD   SPN0000000010028        (filler)
RTD     Q0000200010029        DMG.GCPY.FILE
RTD   SPQ0000400010030        (filler)
RTD     R0000800010031        DDNAME.INSURED
RTD   SPR0000000010032        (filler)
RTD     S0000800010033        DDNAME.FILE
CMT
EPG              (end program globals)
CMT ********************************************************************
CMT
CMT
```

Continued on next page

Continued from previous page

```
CMT ************************************************************
CMT THIS IS THE BEGINNING OF MERGESET 1
CMT Valid entries are CMT, BCB, BEF, BVD, BIF and EMS (EMS must be last).
BMS   MERGE1     (begin Merge Set)
CMT
CMT             ******** CONTROL BLOCK SECTION *********
CMT The following is the control block section for MERGESET 1
CMT Valid sub-entries for the control block section are CMT, CBC,
CMT RBN, RBL, VRF, EFD, SSP, and ECB (ECB must be last).
CMT
BCB   CONTROL    (begin control block)
CBC   MERGESET - VALID ENTRIES ARE MERGESET AND CLOSEDD, DO NOT CLOSE-ALL
RBN   MT97                                      - Rulebase name
RBL   0001                                      - Rulebase revision level
VRF   DMGVRF1                                   - VRF DDNAME
EFD   111090                                    - RFCB effective date
SSP   N                                         - RFCB Suppress Sysprint
ECB   CONTROL    (end control block)
CMT             ******** CONTROL BLOCK SECTION *********
CMT
CMT             ******** EXPLICIT FORMS SECTION ********
CMT THIS IS THE BEGINNING OF THE EXPLICIT FORMS SECTION FOR MERGESET 1
CMT Valid sub-entries for the Explicit forms section are CMT, EFN
CMT and EEF (EEF must be last).
BEF   EXPLICIT   (begin Explicit forms)
EFN   LM.LIFE.ENDORSE1                 00001- EXPLICIT FORM NAME, REV.
EFN   LM.LIFE.ENDORSE2
EFN   LM.LIFE.ENDORSE3
EEF   EXPLICIT   (end Explicit forms)
CMT             ******** EXPLICIT FORMS SECTION ********
CMT
CMT
CMT             ******** VARIABLE DATA SECTION *********
CMT THIS IS THE BEGINNING OF THE VARIABLE DATA SECTION.
CMT Valid sub-entries for the variable data section are CMT, VDR
CMT and EVD (EVD must be last).
BVD   DATA1      (begin variable data)
CMT   TAG TYPE
CMT   —  TAG VALUE
CMT   —  |
VDR     ABrandon M. Williams
VDR     B27
VDR     CMale
VDR     D1-966-3452
VDR     EJan 18 1984
VDR     F$ 25,000
VDR     GStandard
VDR     HFlexible Whole Life
VDR     ILife
VDR     J$166.25
VDR     K$205.25
VDR     L3 Years Thereafter to Jan 17, 2048
VDR     M$166.25
VDR     N01
VDR     Q01
VDR     RINSURED
VDR     SFILE
EVD   DATA1      (end variable data)
EMS   MERGE1     (end Merge Set)
CMT ************************************************************
BMS   MERGE2
BEF   EXPL2
EFN   LM.LIFE.ENDORSE2
EEF   EXPL2
BVD   DATA2
```

Continued on next page

Continued from previous page

```
CMT   TAG TYPE
CMT   —  TAG VALUE
CMT   —  |
VDR     ARachel F. Scott
VDR     B37
VDR     CFemale
VDR     D1-085-9833
VDR     EDec 15 1983
VDR     F$ 20,000
VDR     GStandard
VDR     HFlexible Whole Life
VDR     ILife
VDR     J$134.25
VDR     K$176.50
VDR     L3 Years Thereafter to Dec 14, 2047
VDR     M$145.00
VDR     N01
VDR     Q01
VDR     RINSURED
VDR     SFILE
EVD   DATA2
EMS   MERGE2
CMT ************************************************************************
BMS   MERGE3
BEF   EXPL3
EFN   LM.LIFE.ENDORSE1
EFN   LM.LIFE.ENDORSE3
EEF   EXPL3
BVD   DATA3
CMT   TAG TYPE
CMT   —  TAG VALUE
CMT   —  |
VDR     AJustin Wakefield
VDR     B32
VDR     CMale
VDR     D2-478-9047
VDR     ESep 28 1983
VDR     F$ 25,000
VDR     GStandard
VDR     HFlexible Whole Life
VDR     ILife
VDR     J$177.25
VDR     K$200.00
VDR     L3 Years Thereafter to Sep 27, 2047
VDR     M$235.00
VDR     N01
VDR     Q02
VDR     RINSURED
VDR     SFILE
EVD   DATA3
EMS   MERGE3
CMT ************************************************************************
BMS   MERGE4
BEF   EXPL4
EFN   LM.LIFE.ENDORSE1
EFN   LM.LIFE.ENDORSE3
EEF   EXPL4
BVD   DATA4
CMT   TAG TYPE
CMT   —  TAG VALUE
CMT   —  |
VDR     ALauren D. Huffman
VDR     B23
VDR     CFemale
```

Continued on next page

Continued from previous page

```
VDR     D1-766-4361
VDR     EJul 13 1983
VDR     F$ 15,000
VDR     GStandard
VDR     HFlexible Whole Life
VDR     ILife
VDR     J$ 98.00
VDR     K$115.50
VDR     L3 Years Thereafter to Jul 12, 2047
VDR     M$ 98.00
VDR     N02
VDR     Q02
VDR     RINSURED
VDR     SFILE
EVD   DATA4
EMS   MERGE4
CMT ************************************************************************
BMS   MERGE5
BEF   EXPL5
EFN   LM.LIFE.ENDORSE1
EEF   EXPL5
BVD   DATA5
CMT   TAG TYPE
CMT   -  TAG VALUE
CMT   -  |
VDR     ADaniel J. Fairbanks
VDR     B65
VDR     CMale
VDR     D1-344-2605
VDR     EJun 28 1983
VDR     F$ 85,000
VDR     GStandard
VDR     HFlexible Whole Life
VDR     ILife
VDR     J$235.25
VDR     K$280.00
VDR     L3 Years Thereafter to Jun 27, 2047
VDR     M$240.00
VDR     N01
VDR     Q00
VDR     RINSURED
VDR     SFILE
EVD   DATA5
EMS   MERGE5
CMT ************************************************************************
BMS   MERGE6
BEF   EXPL6
EFN   LM.LIFE.ENDORSE1
EFN   LM.LIFE.ENDORSE2
EEF   EXPL6
BVD   DATA6
CMT   TAG TYPE
CMT   -  TAG VALUE
CMT   -  |
VDR     ABryan B. Masters
VDR     B55
VDR     CMale
VDR     D1-566-3423
VDR     EOct  3 1983
VDR     F$ 55,000
VDR     GStandard
VDR     HFlexible Whole Life
VDR     ILife
VDR     J$235.00
```

Continued on next page

Continued from previous page

```
VDR     K$280.25
VDR     L3 Years Thereafter to Oct  2, 2047
VDR     M$235.00
VDR     N02
VDR     Q02
VDR     RINSURED
VDR     SFILE
EVD   DATA6
EMS   MERGE6
CMT ************************************************************************
```

The Variable Replacement File (VRF)

The Variable Replacement File (VRF) contains the reformatted variable data that is processed by DMGMERGE, the forms list, and the Print Options for the forms.

The VRF is created by the DMGVRFWR program and obtains the information that is written to the VRF from the VDR and the DMGRFMT program.

The program DMGRFMT calls DMGVRFWR to write the tags from the Rulebase Library. VDRs may optionally call DMGVRFWR directly to write additional tags and data not from the Rulebase. See "The DMGVRFWR Subprogram" on page 240 for more information.

The VRF has a record format of UNDEFINED and a maximum block size of 32767. Use the following block sizes for optimum space utilization:

- 23,476 for 3380 devices
- 19,069 for 3350 devices
- 32,760 for tape
- 2 blocks per track for other DASD devices

The VRF Allocation (DMGVRFA) File

A VRF Allocation (DMGVRFA) file increases Documerge efficiency by allocating exactly enough buffer space for Documerge programs to read a VRF. This precise allocation ensures that memory is not wasted and eliminates disk overflow files.

NOTE

Storage defined by a DMGVRFA file is allocated in 31-bit address space if possible. See "31-Bit Addressing" on page 192 for more information.

Each separate DMGVRFA file relates to one VRF. The DMGVRFA file defines:

- the number of tags in the largest Merge Set in the related VRF
- the amount of storage needed for the largest Merge Set in the related VRF.

IMPORTANT!

If you keep (catalog) the VRF, you should keep the VRF allocation file also. If you keep the VRF in a generation data set, keep the VRF allocation file in a corresponding generation data set. If you subsequently delete a generation from the VRF, delete the corresponding generation from the VRF allocation file.

Failure to maintain the VRF allocation file in this way can cause over- or under-allocation when the VRF is read in. Under-allocation causes significant degradation of processing speed.

To ensure that enough storage is allocated for efficient processing of tags and data from the VRF, we recommend that you NOT specify either the TAGBUFF nor the DATABUFF EXEC parameter, and that you use a unique DMGVRFA file for each VRF file.

If you run DMGMERGE with the EXEC parameter "DMGVRFA=N" (bypassing the DMGVRFA allocation file), then your JCL needs to specify the TAGBUFF and the DATABUFF parameters with enough space to read in the largest Merge Set.

To ensure that you are not under-allocating TAGBUFF or DATABUFF, you should also look for the DMGMRG183I message generated by DMGMERGE. Note that DMGMERGE will run successfully if TAGBUFF or DATABUFF is under-allocated, but considerably less efficiently.

The message appears in the ERRDDN file. If the MERGE control card ERRMSG=Y was coded, the message is in the MESSAGE file also. Refer to *Documerge Error Messages* for more information.

Creating DMGVRFA Files

You should create a separate DMGVRFA file for each VRF. Two Documerge programs can generate DMGVRFA files:

Documerge 3.x VDR

In a Documerge 3.1 or later version VDR, you code the DMGVRFA file name in the DMGRFMT control block field

```
RFCB-VRF-ALLOC-DDNAME
```

The DMGRFMT subprogram calls the DMGVRFWR subprogram, which writes this DMGVRFA file automatically when it closes the VRF. Refer to "RFCB-VRF- ALLOC-DDNAME" on page 222 for more information.

If you code a direct call to the DMGVRFWR subprogram, code the DMGVRFA file name in the DDNAME parameter. Refer to "DDNAME An optional field you can use if your VDR creates multiple VRFs. You use this field to specify either:" on page 241 for more information.

If your Documerge 3.x VDR creates multiple VRFs, verify that each DMGVRFA file name matches the related DMGVRF1 file name. A mismatch can result in over allocation or under allocation. Documerge does not warn you if these two file names do not match.

DMGBLDVA

You use the **DMGBLDVA** program to create the DMGVRFA file for a VRF created before Documerge 3.x.

DMGBLDVA also creates the DMGVRFA file for a VRF from a program other than DMGVRFWR (for example, Docusolve).

See "The DMGBLDVA program" on page 255 for more information.

Using DMGVRFA Files

Whenever you code a VRF as input to a Documerge program, code the VRF's related DMGVRFA file also.

If you use JCL file concatenation to combine multiple VRFs into one input VRF, concatenate the related DMGVRFA files also. For example:

```
//DMGVRF1   DD   DSN=FIRSTVRF,DISP=SHR
//          DD   DSN=SECONDVRF,DISP=SHR
//DMGVRFA   DD   DSN=FIRSTVRFA,DISP=SHR
//          DD   DSN=SECONDVRFA,DISP=SHR
```

The EXEC parameter DMGVRFA=N coded for any Documerge program turns off DMGVRFA processing for that job step.

The following Documerge programs can read a VRF and its DMGVRFA file:

- DMGDMPTG
- DMGMERGE
- DMGSORT

  DMGSORT changes the order of a VRF's Merge Sets, but it does not change the VRF's contents. Therefore, DMGSORT does not affect VRF allocation.

  When you code a sorted VRF as input to DMGMERGE, use the same DMGVRFA file for DMGMERGE that you used for DMGSORT.

For example:

- If the DMGSORT JCL is

```
//DMGVRF1   DD   DSN=YOURVRF,DISP=SHR
//DMGVRFA   DD   DSN=YOURVRFA,DISP=SHR
//DMGVRFS   DD   DSN=SORTEDVRF,DISP=(NEW,CATLG),...
```

- Then the DMGMERGE JCL is

```
//DMGVRF1   DD   DSN=SORTEDVRF,DISP=SHR
//DMGVRFA   DD   DSN=YOURVRFA,DISP=SHR
```

If you use JCL file concatenation to create the sorted VRF, use the same DMGVRFA files for DMGMERGE that you used for DMGSORT.

For example:

- If the DMGSORT JCL is

```
//DMGVRF1   DD   DSN=FIRSTVRF,DISP=SHR
//          DD   DSN=SECONDVRF,DISP=SHR
//DMGVRFA   DD   DSN=FIRSTVRFA,DISP=SHR
//          DD   DSN=SECONDVRFA,DISP=SHR
//DMGVRFS   DD   DSN=SORTEDVRF,DISP=(NEW,CATLG),...
```

- Then the DMGMERGE JCL is

```
//DMGVRF1   DD   DSN=SORTEDVRF,DISP=SHR
//DMGVRFA   DD   DSN=FIRSTVRFA,DISP=SHR
//          DD   DSN=SECONDVRFA,DISP=SHR
```

- DMGVDRG
- VRFDEBLD

DMGVRFA File Format

- LRECL=

  Minimum 29 (minimum 33 if variable-length records)

- BLKSIZE=

  User-selected

- RECFM=

  User-selected

- The DMGVRFA file contains only one record:

Position	Length	Description
01-07	07	Constant "DMGVRFA"
08	01	Blank (x'40')
09-14	06	Creation date (MMDDYY)
15-20	06	Creation time (HHMMSS)
21-24	04	Maximum number of tags in one Merge Set (fullword)
25-28	04	Maximum data storage needed for one Merge Set (fullword)
29	01	Constant ***** (asterisk)

The DMGBLDVA Program

The **DMGBLDVA** program creates DMGVRFA files for:

- VRFs created before Documerge 3.0
- VRFs created by programs other than DMGVRFWR (for example, Docusolve).

If the DMGBLDVA program creates DMGVRFA files for these VRFs, Documerge programs that use the DMGVRFA file allocate exactly enough VRF storage for maximum efficiency. See "" on page 252 for more information.

The Documerge program DMGVRFRD reads the DMGVRFA file. In turn, the DMGVRFRD program is called by the following programs:

- DMGMERGE
- DMGSORT
- DMGDMPTG
- DMGVDRG
- VRFDEBLD

DMGBLDVA JCL Example

```
//DMGBLDVA  ** put your job card here **
//*
//* ********************************************************************
//* ** DMGBLDVA prgoram creates DMGVRFA files for VRFs created       **
//* ** before 3.0, or created by other programs, such as Docusolve   **
//* **                                                               **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//BLDVA    EXEC PGM=DMGBLDVA,REGION=2M,
//MESSAGE  DD SYSOUT=*,
//            DCB=(RECFM=FBM,LRECL=151,BLKSIZE=1510)
//DMGVRF1  DD DSN=documerg.v03r02.vrf,DISP=SHR
//DMGVRFA  DD DSN=documerg.v03r02.vrfa,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,1),
//            DCB=(RECFM=F,LRECL=29,BLKSIZE=29)
//*
//
```

DMGBLDVA Files

The following are the DMGBLDVA files in "DMGBLDVA JCL Example" on page 255.

- **DMGVRF1** — the input VRF, usually from a VDR.

- **DMGVRFA** — defines the exact amount of storage required for the input VRF. DMGBLDVA writes only one record to the DMGVRFA file.
  - LRECL=
  - Minimum 29 (minimum 33 if variable-length records)
  - BLKSIZE=
  - User-selected
  - RECFM=
  - User-selected
- **MESSAGE** — contains messages generated by DMGBLDVA. Can be any valid block size, with LRECL at least 100, and variable or fixed, but the following format is recommended:
  - BLKSIZE=1510
  - LRECL=151
  - RECFM=FBM

The DMGSORT Program

The VRF records can be sorted by the DMGSORT program (this is optional). The sort is performed based upon the value contained in the DMG.SKEY.*Groupname* Reserved Tag in ascending or descending order. The variable data assigned to the DMG.SKEY.*Groupname* is defined per Group in the GROUP table, by the SORT= parameter. See "DMG.SKEY.Groupname" on page 321 for more information. DMGSORT invokes IBM sort or a compatible sort product.

DMGSORT JCL Example

```
//DMGSORT   ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* ** DOCUMERGE V. 3.2 - SORT A VRF BASED ON A DMG.SKEY.groupname TAG **
//* **                                                                **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGSORT  EXEC PGM=DMGSORT,REGION=4M
//SYSPRINT DD SYSOUT=*,
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSUDUMP DD SYSOUT=*
//ISIWTL   DD SYSOUT=*
//ISIWTO   DD SYSOUT=*
//DMGVRF1  DD DSN=documerg.v03r02.vrf,
//            DISP=SHR
//DMGVRFA  DD DSN=documerg.v03r02.vrfa,
//            DISP=SHR
//DMGVRFS  DD DSN=documerge.v03r02.sorted.vrf,
//            DISP=(NEW,CATLG),
//            UNIT=sysda,
//            SPACE=(TRK,(15,15),RLSE),
//            DCB=BLKSIZE=23476                    HALF TRACK
//WRKFIL   DD DSN=&&WRKFIL,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                    HALF TRACK
//SORTWK01 DD DSN=&&SORTWK1,
//            DISP=(NEW,DELETE,DELETE),
//            SPACE=(CYL,(10)),
//            UNIT=sysda,
//CTLCARDS DD *
* ------------------------------------------------------------------
* NOTE:
* CONTROL CARDS MUST BE MOVED OVER 1 SPACE FROM DMG 1.7
*    COLUMNS 02-31 = SORT KEY NAME
*    COLUMNS 33-37 = MAXIMUM SORT RECORD SIZE
*    COLUMN  39    = A/D - ASCENDING/DESCENDING
*    COLUMN  43-44 = AMODE FOR SORT MODULE (24 OR 31)
* ------------------------------------------------------------------
* REPLACE THE FOLLOWING CONTROL CARD WITH YOUR TAG NAME, ETC.
* ------------------------------------------------------------------
 DMG.SKEY.groupname             23464 a   31
/*
//
```

DMGSORT EXEC Parameters

Parameter	Value
DATABUFF=	Defines the amount of storage for the data in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the DATABUFF= value: ■ If the DMGVRFA= value is N, DATABUFF= defines the exact amount of storage. The default value is 500 K. ■ If the DMGVRFA= value is not N, DATABUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tag data combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is not N, omit the DATABUFF= parameter. **NOTE:** This parameter is optional. You can use it to limit the storage defined by the DMGVRFA file. Do not use this parameter if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.
DMGVRFA=	Indicates whether DMGMERGE uses a DMGVRFA file. See "" on page 252. **N** DMGMERGE does not use a DMGVRFA file. **Y** DMGMERGE uses a DMGVRFA file. The default value is Y. **NOTE:** If you specify a value other than N, DMGMERGE uses a DMGVRFA file. (DMGMERGE reads only the first character of this value.) Therefore, omit DMGVRFA= if you want the DMGVRFA file.
NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. The memory allocated for LM/MM is the number you specify multiplied by the block size of the WRKFIL. (See "WRKFIL" on page 378.) Specify a number from **2** to **2048**. You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)
TAGBUFF=	Defines the amount of storage for the tags in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the TAGBUFF= value: ■ If the DMGVRFA= value is N, TAGBUFF= defines the exact amount of storage. The default value is 500 K. ■ If the DMGVRFA= value is not N, TAGBUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tags combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is not N, omit the TAGBUFF= parameter. **NOTE:** This parameter is optional. You can use it to limit the storage defined by the DMGVRFA file. Do not use this parameter if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.

Parameter	Value
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. If the memory allocation is smaller than the amount of data, LM/MM uses the WRKFIL for secondary space. (See "WRKFIL" on page 378.) The WORKBUFF= value must be at least twice the value of the WRKFIL block size. The WORKBUFF= value must not exceed the WRKFIL block size multiplied by 2048. Valid values are: **nnnK** nnn units of 1024 bytes **nnnM** nnn units of 1,048,576 bytes (1024 — 1024) You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)

DMGSORT Files

The following are the DMGSORT files in "DMGSORT JCL Example" on page 257:

- **SYSPRINT** — Contains DMGSORT messages.
- **DMGVRF1** — The input VRF, usually from a VDR.
- **DMGVRFA** — The input VRF Allocation file, from a VDR or the DMGBLDVA program. When reading in the sorted VRF (in DMGMERGE or other programs), use the same DMGVRFA input to this DMGSORT (and output by your VDR).

  DMGSORT usually requires the DMGVRFA file, but it does not create the DMGVRFA file. If no DMGVRFA file exists, do either of the following:

  - Run DMGSORT with the EXEC parameter DMGVRFA=NO. See "DMGVRFA=" on page 258.
  - Run the DMGBLDVA program to create a DMGVRFA file. See "The DMGBLDVA program" on page 255.
- ISIWTL — Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTL is intended for programmer and application user error messages. Using Oracle standards, any program that has written to ISIWTL has encountered a fatal error.
- **ISIWTO** — Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTO is intended for computer operator messages. Using Oracle standards, any program that has written to ISIWTO has encountered a fatal error.

The contents of the DMGVRFA file are not affected by sorting the Merge Sets. DMGSORT rearranges the sequence of the Merge Sets in the VRF, but does not change anything inside of a Merge Set.

- **DMGVRFS** — Indicates the name of the file containing the sorted VRF output. The file's physical record format is undefined.
- **WRKFIL** — Overflow file, used by the List Manager/Memory Manager (LM/MM) for internal storage management.
- **CTLCARDS** — Designates the DMG.SKEY.*Groupname* Reserved Tag used for sorting purposes. *This must begin in column 2*. A maximum sort record size can be defined in columns 33 through 37 (optional). The X in column 39 indicates the order in which the sort is placed. A 'D' in the column 39 indicates descending order. 'A' or a blank indicates ascending order.

DMGSORT Options

DMGSORT invokes a load module named SORT, which is expected to be the standard IBM SORT program or a vendor equivalent. The SORT program may allow for various options which can fine-tune the sorting operations. Consult your sort program's user manual to review any sort options to select the sorting technique best suited to your DMGSORT run. Note that sometimes experimentation is the only way to determine the most efficient sorting options.

DMGSORT Control Card Format

The SYSIN sort control card is positional.

Position	Value
Column 1	Valid values are: ***** (asterisk) This is a comment record and is ignored. blank The remainder of the record is a sort control card. You can have any number of comments, but only one DMGSORT control card.
Column 2	DMG.SKEY.*Groupname* "*Groupname*" is the name of the Group to be sorted.
Column 32	blank
Column 33-37	Maximum sort record size, numeric, including leading zeros.
Column 38	blank
Column 39	Valid values are: **A** Ascending **D** Descending
Column 40	blank
Column 41	The number of sort work files. Not used in MVS.
Column 42	blank
Column 43-44	**31** DMGSORT runs the SORT program in 31-bit addressing mode. See "31-Bit Addressing" on page 192. **NOTE:** Using 31, run some tests to see if your SORT program supports 31-bit addressing. **Any other value** DMGSORT runs the SORT program in 24-bit addressing mode.
Column 45	blank
Column 46-49	(Optional) **OMIT** If you code the **OMIT** value in the DMGSORT control card for a Merge Set, any Merge Set that doesn't contain the requested sort key is omitted (e.g., DMGSORT doesn't place the Merge Set in the sorted output file). Omitting a Merge Set from the sort output file can reduce the sort time because fewer records have to be sorted. **IMPORTANT:** Because a sorted VRF can omit some Merge Sets that might be kept by other sorts. always use the original unsorted VRF as input to DMGSORT.
Column 50-80	These positions are ignored.

DMGSORT Sort Work Block Size

The block size selected for the sort work files may make a significant difference in the DMGSORT run time. A bigger block size often produces a more efficient DMGSORT run.

The maximum block size varies somewhat depending on the sort program used. Generally the maximum block size is 12 bytes less than the track capacity for the DASD device used for sort work files. However, your sort program may impose other limits on the block size, up to a maximum of 32767.

NOTE

DMGSORT generates variable-length records for sorting by the invoked sort program even though the VRF itself contains undefined records.

The DMGDMPTG Program

The program DMGDMPTG generates a printout of the tags contained in a VRF.

DMGDMPTG Sample JCL

```
//DMGDMPTG ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* **       DOCUMERGE V. 3.2 VRF TAG REPORTER                        **
//* **                                                                **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DUMPTAG  EXEC PGM=DMGDMPTG,PARM='SORT=YES WORKBUFF=500K'
//SYSPRINT DD SYSOUT=*
//DMGVRFA  DD DSN=your.documerg.v03r02.vrfa,
//            DISP=SHR
//DMGVRF1  DD DSN=your.documerg.v03r02.vrf,
//            DISP=SHR
//WRKFIL   DD DSN=&&WRKFIL,
//            UNIT=sysda,
//            DISP=(NEW,DELETE),
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                HALF TRACK
//
```

DMGDMPTG EXEC Parameters

Parameter	Value
DATABUFF=	Defines the amount of storage for the data in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the DATABUFF= value: ▪ If the DMGVRFA= value is N, DATABUFF= defines the exact amount of storage. The default value is 500 K. ▪ If the DMGVRFA= value is not N, DATABUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tag data combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is not N, omit the DATABUFF= parameter. **NOTE:** This parameter is optional. You can use it to limit the storage defined by the DMGVRFA file. Do not use this parameter if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.
DMGVRFA=	Indicates whether DMGMERGE uses a DMGVRFA file. Refer to "" on page 252. **N** DMGMERGE does not use a DMGVRFA file. **Y** DMGMERGE uses a DMGVRFA file. The default value is Y. **NOTE:** If you specify any value other than N, DMGMERGE uses a DMGVRFA file. (DMGMERGE reads only the first character of this value.) Therefore, omit the DMGVRFA= parameter if you want the DMGVRFA file.

Parameter	Value
NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. The memory allocated for LM/MM is the number you specify multiplied by the block size of the WRKFIL. (See "WRKFIL" on page 378.) Specify a number from **2** to **2048**. You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)
SORT=	Indicates whether DMGDMPTG should sort the tags prior to verifying and reporting on them. **YES** DMGDMPTG will sort the tags. **NO** DMGDMPTG will not sort the tags. The default value is no.
TAGBUFF=	Defines the amount of storage for the tags in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the TAGBUFF= value: ■ If the DMGVRFA= value is N, TAGBUFF= defines the exact amount of storage. The default value is 500 K. ■ If the DMGVRFA= value is not N, TAGBUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tags combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is not N, omit the TAGBUFF= parameter. **NOTE:** This parameter is optional. You can use it to limit the storage defined by the DMGVRFA file. Do not use this parameter if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. If the memory allocation is smaller than the amount of data, LM/MM uses the WRKFIL for secondary space. (See "WRKFIL" on page 378.) The WORKBUFF= value must be at least twice the value of the WRKFIL block size. The WORKBUFF= value must not exceed the WRKFIL block size multiplied by 2048. Valid values are: **nnnK** nnn units of 1024 bytes **nnnM** nnn units of 1,048,576 bytes (1024 — 1024) You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)

DMGDMPTG Files

- **SYSPRINT** — Indicates the output class or dataset name where the tag list is written.
- **DMGVRF1** — Indicates the dataset name that contains the VRF, sorted or unsorted.
- **DMGVRFA** — The input VRF Allocation file, from a VDR or the DMGBLDVA program.
- **WRKFIL** — Overflow file, used by the List Manager/Memory Manager (LM/MM) for internal storage management.

DMGDMPTG Report Fields

The following table lists each field in the Tag Dump Utility Report:

DMGDMPTG Report Fields

Field Name	Description
SET #	The sequence number of the Merge Set within the VRF.
TAG #	The sequence number of the tag within the Merge Set.
TAG NAME	The name assigned to the tag in the VRF.
DATLEN	The length of the data in the VRF for the tag.
TAG DATA	The variable data in the VRF for the tag. The data is truncated after 70 bytes on the report.

DMGDMPTG report format

```
D O C U M E R G E - V03 R02 L00
Date: 11/11/98                        Tag Dump Utility                    Page 1
Time: 16:56:41
SET #    Tag #   Tag Name               Datlen   Tag Data
-------  -----   ------------------     ------   -----------------
      1      1   TAG.NAME.1                  2   40
      1      2   TAG.NAME.2                  7   $966.25
      1      3   TAG.NAME.3                  7   $805.25
      1      4   TAG.NAME.4                  7   $166.25
      1      5   TAG.NAME.5                 19   Flexible Whole Life
      1      6   TAG.NAME.6                  0
```

DMGDMPTG Return Codes

The following list defines the DMGDMPTG return codes.

0	Normal.
8	User Error: invalid value in SORT field in EXEC PARM.
20	Error closing SYSPRINT output file.
100+rc	Error opening SYSPRINT output file. **rc** is the ISISEQIO return code. If rc=124, then verify that you have a valid DD for SYSPRINT.
200+rc	Error writing to SYSPRINT output file. **rc** is the ISISEQIO return code.
300+rc	Error reading the VRF. "rc" is the DMGVRFRD return code.

The DMGVERFY Program

The DMGVERFY program is a utility for testing and debugging. For example, you can use DMGVERFY to inspect the contents of a tag that has generated an incorrect count, or to reveal unprintable data in the VRF when a printing problem has occurred.

DMGVERFY examines the tags contained in a VRF and performs the following operations:

- Edits Documerge Reserved tags for correct format.
- Verifies that DMG.C. and DMG.CHECKPOINT tags are in the correct format.
- Checks valid tags for invalid (high or low) values.
- Issues a warning message for each invalid tag.
- Reports the use of invalid data or invalid tag names.
- Reports the invalid occurrence in the VRF of a Documerge internal reserved tag.

DMGVERFY Sample JCL

```
//DMGVERFY  ** put your job card here **
//*
//* ****************************************************************
//* **                                                            **
//* **     DOCUMERGE V. 3.2 VERIFY VRF TAGS UTILITY               **
//* **                                                            **
//* ****************************************************************
//*
//*
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGVERFY EXEC PGM=DMGVERFY,PARM='SORT=YES WORKBUFF=500K'
//SYSPRINT DD SYSOUT=*
//DMGVRFA  DD DSN=your.documerg.v03r02.vrfa,
//            DISP=SHR
//DMGVRF1  DD DSN=your.documerg.v03r02.vrf,
//            DISP=SHR
//WRKFIL   DD DSN=&&WRKFIL,
//            UNIT=sysda,
//            DISP=(NEW,DELETE),
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                HALF TRACK
//
```

DMGVERFY EXEC Parameters

Parameter	Value
DATABUFF=	(Optional) defines the amount of storage for the data in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the DATABUFF= value: ■ If the DMGVRFA= value is **N**, DATABUFF= defines the exact amount of storage. The default value is 500 K. ■ If the DMGVRFA= value is **Y**, DATABUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tag data combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is **Y**, omit the DATABUFF= parameter. **NOTE:** Do not use this parameter if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.

Parameter	Value
DMGVRFA=	Indicates whether DMGMERGE uses a DMGVRFA file. Refer to "" on page 252. **N** DMGMERGE does not use a DMGVRFA file. **Y** DMGMERGE uses a DMGVRFA file. The default value is Y. **NOTE:** If you specify any value other than N, DMGMERGE uses a DMGVRFA file. (DMGMERGE reads only the first character of this value.) Therefore, omit the DMGVRFA= parameter if you want the DMGVRFA file.
SORT=	Indicates whether DMGVERFY should sort the tags prior to verifying and reporting on them. **Y** or **YES** DMGVERFY will sort the tags. **N** or **NO** DMGVERFYwill not sort the tags. The default value is N.
NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. The memory allocated for LM/MM is the number you specify multiplied by the block size of the WRKFIL. (See "WRKFIL" on page 378.) Specify a number from **2** to **2048**. You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)
TAGBUFF=	Defines the amount of storage for the tags in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the TAGBUFF= value: ▪ If the DMGVRFA= value is N, TAGBUFF= defines the exact amount of storage. The default value is 500 K. ▪ If the DMGVRFA= value is not N, TAGBUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tags combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is not N, omit the TAGBUFF= parameter. **NOTE:** This parameter is optional. You can use it to limit the storage defined by the DMGVRFA file. Do not use this parameter if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. If the memory allocation is smaller than the amount of data, LM/MM uses the WRKFIL for secondary space. (See "WRKFIL" on page 378.) The WORKBUFF= value must be at least twice the value of the WRKFIL block size. The WORKBUFF= value must not exceed the WRKFIL block size multiplied by 2048. Valid values are: **nnnK** nnn units of 1024 bytes **nnnM** nnn units of 1,048,576 bytes (1024 x 1024) You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.)

DMGVERFY Files

- **SYSPRINT** — Indicates the output class or dataset name where the tag list is written.
- **DMGVRF1** — Indicates the dataset name that contains the VRF, sorted or unsorted.
- **DMGVRFA** — The input VRF Allocation file, from a VDR or the DMGBLDVA program.
- **WRKFIL** — Overflow file, used by the List Manager/Memory Manager (LM/MM) for internal storage management.

DMGVERFY Messages

The first column of following table lists the DMGVERFY messages, and the second column lists the names of the tags that can evoke each message:

DMGVERFY Messages

Message	Tag Name
Invalid form name for:	DMG.BNR.*Groupname* DMG.TLR.*Groupname*
Invalid file name for:	DMG.DD.*Groupname* DMG.ERRDD.*Groupname* DMG.ERROR.VRF
Invalid forms list for:	DMG.FLST.*Groupname*
Invalid print options for:	DMG.OPT.*Groupname*
Invalid PEDEF name for:	DMG.FDEF.*Groupname* DMG.MDEF.*Groupname* DMG.PDEF.*Groupname*
Invalid DMG.C. tag	DMG.C. tags DMG.CHECKPOINT. tags
Invalid data for:	Any tag with data of high or low values
Documerge Internal Reserved tag found in VRF and will override DMG value	DMG.ALT.CURRENT.DATE DMG.CURRENT.xxx DMG.DATE DMG.END.OF.SET DMG.BENCHMARK DMG.HASHMARK DMG.ITEM.LINE.COUNT DMG.LINE.COUNT DMG.PAGE.NUMBER DMG.POL.COUNT DMG.PRODUCT DMG.SET.NUMBER DMG.SHEET.COUNT DMG.SHEET.NUMBER DMG.SSI.COUNT DMG.TIME DMG.TOTAL.PAGES DMG.VERSION

DMGVERFY Report Format

Following is a sample DMGVERFY report. Its detail lists the number, type, and name of each tag that DMGVERFY has validated.

DMGVERFY report format

```
Run Date:  06/22/1998           D O C U M E R G E - V03 R02 L00     Page    1
Run Time:  14:06:44                  VRF Verify Messages

Merge Set ID Tag:   AB123456789
Merge Set Number:   2
Tag #
-------
      12  DMGVERFY - Documerge Internal Reserved tag: DMG.MERGESET.ID
      13  DMGVERFY - Documerge Internal Reserved tag: DMG.OPT.FILE
      14  DMGVERFY - Documerge Internal Reserved tag: DMG.OPT.INSURED
      15  DMGVERFY - Documerge Internal Reserved tag: DMG.SKEY.FILE
      16  DMGVERFY - Documerge Internal Reserved tag: DMG.SKEY.INSURED

Number of merge sets processed:         2
```

DMGVERFY Return Codes

The following list defines the DMGVERFY return codes.

0	Normal
4	A warning message was called on a tag in the VRF.
8	User Error: invalid value in SORT field in EXEC PARM.
20	Error closing SYSPRINT output file.
100+rc	Error opening SYSPRINT output file. **rc** is the ISISEQIO return code. If rc=124, then verify that you have a valid DD for SYSPRINT.
200+rc	Error writing to SYSPRINT output file. **rc** is the ISISEQIO return code.
300+rc	Error reading the VRF. "rc" is the DMGVRFRD return code.

The VRFBUILD / VRFDEBLD Programs

VRFBUILD and VRFDEBLD are Documerge utility programs that convert a VRF into a flat file which is easier to read and update.

- **VRFDEBLD** reads a VRF (and an optional DMGVRFA file) and converts it into a flat file — in a *control card* format. We call these records *cards*, because you can edit them like you edit a card file.

  Use VRFDEBLD to convert the VRF into a format that is easy to read and debug, and easy to change should data need to be changed. You can edit a flat file using TSO or other similar text editors.

- **VRFBUILD** reads the (optionally edited) flat file and writes a new VRF (and optional DMGVRFA file).

  Use VRFBUILD to build a VRF from the optionally edited flat file, or use it to make changes to print options for testing and debugging.

Flat File Format

The flat file created by VRFDEBLD, and used as input to VRFBUILD, has a special format. This special format

- Allows for tag data lengths to exceed the LRECL of the flat file.
- Makes the DMG.FLST (form names) and DMG.OPT (print options) more readable.

Here are the specifications to use when formatting a VRFDEBUILD flat file:

- Fixed or variable length format.

NOTE

If you want to read the flat file with a COBOL program, use fixed-length records, because COBOL does not give you the record length for variable length records.

- Maximum LRECL of 200 (204 for variable length). Oracle recommends that you use the maximum length.
- Minimum LRECL for Documerge 3.x VRFs is 96 for fixed length, 100 for variable length.

  The minimum LRECL might increase with later releases of Documerge because the minimum LRECL is based on the number of print options in the DMG.OPT.*Groupname* tag, and future releases might increase the number of print options.

In the flat file, column 1 of each record indicates the type of data record. The following table summarizes the valid values that can appear in column 1 of each data record.

*****	(Asterisk) comment, ignore flat file record.
O	(Alphabetic O) set run time options.
F	Set the DMGVRF1 and optional DMGVRFA output file name.
L	Begin a new Group. The form names and print options will follow.
T	Define a tag name and its data.
E	End this merge set.
blank	Continuation data for this type of record (for L and T records).

The following topics address each type of data record.

Comment card — Asterisk in Column 1

Any flat file record that contains an asterisk (*) in column 1 is ignored.

Use to

- Place comments to document any changes you have made to the flat file.
- Remove a form from the DMG.FLST and associated DMG.OPT tag without actually deleting the data record from the flat file. Placing an asterisk in column 1 effectively removes the form.

Syntax

Column	Description
1	Asterisk (*)
2-end	Ignored.

Coding Examples:

```
*This record is ignored
* So is this record
```

Notes

VRFDEBLD places a comment record between each merge set as follows:

```
*----- BEGIN MERGE SET NUMBER nnnnnnn
```

Where nnnnnnn is the sequential number if this merge set, starting with 0000001.

Set Runtime Options — O in Column 1

VRFDEBLD allows some runtime options. These are set with this O (options) card.

Syntax

Column	Description
1	Alphabetic O
2	blank
3-end	The runtime option desired.

The valid options are:

- **FOLD**Convert some values to uppercase.
- **NOFOLD**Do not convert values to uppercase.

FOLD converts the following values from lowercase to uppercase characters:

- Tag names (from the T card).
- Group names (from the L card).
- Form names and print options (continuation data following the L card).

Here are coding rules for FOLD or NOFOLD:

- FOLD is the default.
- The words FOLD and NOFOLD must be in uppercase characters.
- The FOLD or NOFOLD option takes affect with the next data record in the flat file.
- If you have form names with lowercase characters, use the NOFOLD option.
- The data for a tag is never converted to uppercase.

Coding Examples:

```
O FOLD
O NOFOLD
```

Select Output File Names — F in Column 1

The F card allows you to select the output file (DD) names for the VRF and VRF allocation file. This card is optional; the default names are DMGVRF1 (VRF) and DMGVRFA. (VRF allocation).

Syntax

Column	Description
1	Constant F (upper or lower case)
2	blank
3-19	One of two formats: The VRF file name in columns 3-10; columns 11-19 are blank; this assigns the VRF filename but not the allocation file. The allocation file defaults to the last named allocation file from a previous F card, or to filename "DMGVRFA" of no F card has specified a VRF allocation file name. A slash (/) in column 3, the VRF file name in columns 4-11, and the VRF allocation file name in columns 12-19. This format is valid for Documerge 3.0 and higher only. Note that the above two formats are the same ones used by DMGVRFWR.
20-end	Ignored.

Coding Examples

```
F VRFFILE              Set VRF name to VRFFILE; leave VRFA name unchanged
F /VRFFILE VRFAFILE    Set VRF name to VRFFILE and VRFA name to VRFAFILE
```

Notes

Most of the time you won't need this F card, and you will use the default names of DMGVRF1 and DMGVRFA.

Group Cards — L in column 1

Use this type of data record to code the Group name and its forms and print options. The L card is followed by one or more "continuation" cards — cards with column 1 blank. These continuation cards contain the actual form names and print options for this Group. The Group ends with the next non-blank and non-comment record (that is, a card with other than a blank or an asterisk in column 1).

The L card and the continuation cards correspond to the DMG.FLST.*Groupname* and DMG.OPT.*Groupname* tags in the VRF.

Syntax

L Card Syntax:

Column	Description
1	constant L (either uppercase or lowercase).
2	blank.
3-23	the 21-character Group name (folded to uppercase if using runtime option FOLD).
24-end	ignored.

Continuation Card Syntax — Cards Following the L Card

Column	Description
1	blank
2	blank
3-34	VLAM member name (32 characters, folded if runtime option is FOLD)
35-39	VLAM member revision (5 digits, leading zeros required)
40-44	blank
45-end	Data for DMG.OPT tag for this form. This must be coded exactly as it is to appear in the DMG.OPT; for example: **DUP SEP MAI STA ...**

Coding Examples

Here are the Group cards for the standard Documerge 2.1 Metacode demo for the first merge set:

```
L INSURED
  LM.LIFE.DEC                    00000      DUP SEP MAI STA ODD POR OFF
  LM.LIFE.OVERLAY                00001      DUP SEP MAI STA ANY POR OVL
  LM.LIFE.CONTRACT                00000      DUP SEP MAI STA ANY POR ON
  LM.LIFE.ENDORSE1                00001      DUP CON MAI STA ANY POR ON
  LM.LIFE.ENDORSE2               00001      DUP CON MAI STA ANY POR ON
 LM.LIFE.ENDORSE3             00001     DUP CON MAI STA ANY POR ON
L FILE
  LM.LIFE.DEC                    00000      SIM SEP MAI STA ODD POR OFF
  LM.LIFE.CONTRACT              00000      SIM SEP MAI STA ODD POR OFF
  LM.LIFE.ENDORSE1                00001      SIM SEP MAI STA ANY POR OFF
  LM.LIFE.ENDORSE2               00001      SIM SEP MAI STA ANY POR OFF
  LM.LIFE.ENDORSE3                00001      SIM SEP MAI STA ANY POR OFF
```

L Card Notes and Cautions

You can place an asterisk in column 1 of a continuation card to delete a form without physically deleting it from the flat file.

CAUTION!

In Documerge 3.x, the continuation cards for the forms and options (following the L card) will usually exceed 80 characters due to the number of print options.

Take care when editing print options (in a continuation card following an L card), because

- The record often exceeds 80 characters, so if you use TSO or other editor that does not display more than 80 characters, be careful not to delete or unintentionally change data that does not display.
  If your editor has a wrap feature that lets you see more than 80 columns, Oracle suggests you turn on this feature.
- Print options are column dependent. If you change an option, make sure you don't shift any other options out of their column-dependent locations.

For details about print option overrides, see "EXPLICIT-OPTIONS-OVERRIDE" on page 230.

Tag name — T Card

The T card is where you code the data for a BPSD. Also, you can code data tags used by the exit program here. Code all data except for DMG.FLST and DMG.OPT on a T card. You can even code a DMG.FLST and DMG.OPT tag with a T card, but the L card is much easier to code for these.

Syntax

Column	Description
1	Constant T (lower or uppercase)
2	blank
3-32	Tag name (30 chars maximum, no imbedded blanks; folded to uppercase if runtime option is FOLD)
33-34	blank
35-end	Data for this tag; never folded to uppercase; see rules about trailing blanks and hyphen

Continuation Card Syntax — Cards Following the T Card

If the data does not fit into one record, then continue the data with one or more continuation cards (blank in column 1), as follows:

Column	Description
1	blank
2	blank
3-end	Continue the data.

You can code as many continuation cards as needed, but the total length of the data cannot exceed 65535. The data in column 3 of the continuation card butts up to the truncated data from the previous card.

Coding Examples

The following T cards are from the standard Documerge 2.1 Metacode installation demo VRF:

Metacode installation demo VRF with T cards example

```
T NAME                          Brandon M. Williams
T AGE                            27
T SEX                            Male
T POLICY.NUMBER                   1-966-3452
T DMG.MERGESET.ID                 1-966-3452
T POLICY.DATE                     Jan 18 1984
T SUM.INSURED                     $ 25,000
T PREMIUM.CLASS                   Standard
T BENEFIT.DESC                   Flexible Whole Life
T TERMINATION.PERIOD              Life
T ANNUAL.MAX.1                    $166.25
T ANNUAL.MAX.2                    $205.25
T PREMIUM.PERIOD                 3 Years Thereafter to Jan 17, 2048
T ANNUAL.PERIOD                   $166.25
T DMG.GCPY.INSURED                01
T DMG.GCPY.FILE                   01
T DMG.DD.INSURED                  INSURED
T DMG.DD.FILE                     FILE
T DMG.SKEY.INSURED                01Brandon M. Williams -
T DMG.SKEY.FILE                      021-966-3452
```

We will show a T card with continuation cards after we discuss truncating trailing blanks and a trailing hyphen in the next section.

Truncating Trailing Blanks and a Trailing Hyphen in T Card Data

In a T card, and any continuation cards for a T card:

- Trailing blanks are always truncated (removed) from the data.
- If the data ends with a hyphen, that hyphen is ignored, but blanks to the left of the hyphen are not truncated.
- Use a hyphen to force blanks to be kept instead of truncated.

This is best explained with several examples. Assume we want tag name "MYTAG" with data "ABC". We could code this in several ways. The simplest is:

```
T MYTAG                                ABC
```

However, the following would all give the same results (the characters "ABC" for the data value):

```
T MYTAG
  ABC

T MYTAG
  A
  B
  C

T MYTAG                                A
  B
  C

T MYTAG
  A
  BC

T MYTAG

  ABC
```

The previous example has a totally blank card before the data "ABC". We would still get "ABC" for the data because the trailing blanks are truncated from each data card, so a totally blank card is ignored when processing a tag continuation card.

Remember: On the T card, the data starts in column 35; on the continuation card, the data starts in column 3. And, in a continuation card, the data in column 3 butts up to the last character from the data in the previous card after trailing blank and trailing hyphen truncation.

Suppose you want trailing blanks in the data. Then code a hyphen following the last blank in the data. If a data card ends with a hyphen (-), the hyphen is ignored (not placed in the data), but data to the left of the hyphen is not truncated. Suppose we want the data to be "A B C" (blanks between each character). Then we could code one of several T cards:

```
T MYTAG                                A B C

T MYTAG                                A -
  B -
  C

T MYTAG
  A B -
  C
```

If we wanted the data to be "A B C — (trailing blank after the C), we could code:

```
T MYTAG                                  A B C -
```

So the hyphen is NOT a continuation character (like we normally use in SYSIN control cards). Instead it allows us to keep trailing blanks in the data. It is OK to code the hyphen even if the data does not have blanks. The following gives a data value of "ABC".

```
T MYTAG                                  ABC-
```

Because the VRFBUILD ignores the trailing hyphen, the previous code is the same as

```
T MYTAG                                  ABC
```

Suppose you need a trailing hyphen in the data. Then code two hyphenes. The following gives a data value of "ABC-" (a trailing hyphen in the data):

```
T MYTAG                                  ABC --
```

The rightmost hyphen is the "special hyphen", the one that is ignored. The hyphen to the left is treated as part of the data value.

Other than a blank (hex 40) and a hyphen, all other characters are treated as data characters, including any binary data (such as the DMG.LPG tag).

Program VRFDEBLD generates a trailing hyphen in the data record only to prevent needed blanks from being truncated, or to place a trailing hyphen in the data (in which case 2 hyphenes are coded as described above).

The following shows a T card with continuation cards.

```
T LARGE.TAG                        This is a large tag with lots of data.  Th
  e data continues onto this continuation card and is so large that sev
  eral continuation cards are needed.  Notice that words "wrap" – are spli
  – across two cards.  When VRFBUILD creates the VRF, the data in col
  umn 3 in a continuation card will immediately follow the truncated data from
   the previous card.  Notice the leading blank on this one; this puts a space
   between the words that are on two different
   cards.  This is because trailing blanks are truncated (ignored) in each
   T card and continuation cards.  Another way to put spaces between words -
  that are on two cards is to end the first card with a "blank-hyphen", because
-
  the ending hyphen is ignored, but data to the left of the hyphen is not -
  truncated.  Note that most of the time your data will probably fit onto -
  one T card with no continuation cards, but you still need to learn the -
  rules for truncating trailing blanks and a trailing hyphen.  Also,
   if I want a data record to end with with a hyphen (where this hyphen is –
   actually part of the tagged data), I must code two hyphenes.
   As an example, here I code two hyphenes --—  The rightmost (2nd) hyphen is
   the special hyphen that is ignored by program
   VRFBUILD because it is a trailing hyphen; that is, because the hyphen is the
   last non-blank character in the card.  The first hyphen is kept, as is all
   data
   to the left of this special trailing hyphen.
```

End This MergeSet — E card.

The E card terminates the merge set. The E card is implied at the end of the file; if you don't code a final E card, one is assumed. Therefore, you need the E card only if you have more than one merge set.

Syntax

Column	Description
1	Constant E
2-end	ignored.

Coding Example

```
E   The Merge Set is ended; these comments are ignored.
```

VRFBUILD MVS JCL

Input files:

SYSIN The flat file input.

Output files:

DMGVRF1	VRF input (similar to VDR step)
DMGVRFA	VRF allocation file (similar to VDR step)
ISIWTL	Error message(s) from VRFBUILD. Contains message only if return code is not zero. Empty if return code is zero

Sample MVS JCL to Run VRFBUILD

```
//*VRFBUILD ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* ** VRFBUILD will read the optionally edited flat file created by **
//* ** VRFDEBLD, and write a new VRF.                                 **
//* **                                                                **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//VRFBUILD EXEC  PGM=VRFBUILD
//DMGVRF1  DD  DSN=documerg.v03r02.vrf,
//             DISP=(NEW,CATLG,DELETE),
//             UNIT=sysda,
//             DCB=BLKSIZE=23476,
//             SPACE=(TRK,(30,30),RLSE)
//DMGVRFA  DD  DSN=documerg.v03r02.vrfa,
//             DISP=(NEW,CATLG,DELETE),
//             UNIT=sysda,
//             DCB=(LRECL=80,BLKSIZE=80,RECFM=F),
//             SPACE=(TRK,1)
//SYSIN    DD  DSN=documerg.v03r02.vrfdebld,DISP=SHR
//ISIWTL   DD SYSOUT=*
//
```

VRFDEBLD MVS JCL

Input files:

DMGVRF1	VRF input (similar to DMGMERGE step)
DMGVRFA	VRF allocation file (similar to DMGMERGE step)

Output files:

OUTPUT The flat file output.

Sample MVS JCL to Run VRFDEBLD

```
//VRFDEBLD  ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* ** VRFDEBLD will read a VRF and convert it into a flat file.     **
//* ** This flat file is in a format which is easier to read and     **
//* ** edit, and lists form names with their associated print options**
//* ** for easier debugging.  The maximum RECL is 200 (204 if VB).   **
//* **                                                                **
//* ** After editing the flat file run VRFBUILD to crerate a new VRF.**
//* **                                                                **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//VRFDEBLD EXEC  PGM=VRFDEBLD
//DMGVRF1  DD  DSN=documerg.v03r02.vrf,DISP=SHR
//DMGVRFA  DD  DSN=documerg.v03r02.vrfa,DISP=SHR
//OUTPUT   DD  DSN=documerg.v03r02.vrfdebld,
//             DISP=(NEW,CATLG,DELETE),
//             UNIT=sysda,
//             DCB=(RECFM=FB,LRECL=200,BLKSIZE=2000),
//             SPACE=(TRK,(150,150),RLSE)
//WRKFIL   DD  DSN=&&WRKFIL,
//             DISP=(NEW,DELETE),
//             UNIT=sysda,
//             DCB=BLKSIZE=0,          OK to use BLKSIZE=0 for WRKFIL
//             SPACE=(TRK,(1,30))
//ISIWTL   DD SYSOUT=*
//
```

Sample flat file

Here is a completed sample flat file, running VRFDEBLD against the VRF from the standard Documerge installation Metacode demo.

(Note the "BSS" and "CSS" options for subset support; these are optional and default to "CSS". There are other options also. See "Format of the Explicit Forms List (EXP2)" on page 229 for details about each print option value.)

Flat file example

```
* — — - BEGIN MERGE SET NUMBER 0000001
L INSURED
  LM.LIFE.DEC                    00000      DUP SEP MAI STA ODD POR OFF                    BSS
  LM.LIFE.OVERLAY                00001      DUP SEP MAI STA ANY POR OVL                    CSS
  LM.LIFE.CONTRACT               00000      DUP SEP MAI STA ANY POR ON                     CSS
  LM.LIFE.ENDORSE1               00001      DUP CON MAI STA ANY POR ON                     CSS
  LM.LIFE.ENDORSE2               00001      DUP CON MAI STA ANY POR ON                     CSS
  LM.LIFE.ENDORSE3               00001      DUP CON MAI STA ANY POR ON                     CSS
L FILE
  LM.LIFE.DEC                    00000      SIM SEP MAI STA ODD POR OFF                    BSS
  LM.LIFE.CONTRACT               00000      SIM SEP MAI STA ODD POR OFF                    CSS
  LM.LIFE.ENDORSE1               00001      SIM SEP MAI STA ANY POR OFF                    CSS
  LM.LIFE.ENDORSE2               00001      SIM SEP MAI STA ANY POR OFF                    CSS
  LM.LIFE.ENDORSE3               00001      SIM SEP MAI STA ANY POR OFF                    CSS
T NAME                           Brandon M. Williams
T AGE                            27
T SEX                            Male
T POLICY.NUMBER                  1-966-3452
T DMG.MERGESET.ID                1-966-3452
T POLICY.DATE                    Jan 18 1984
T SUM.INSURED                    $ 25,000
T PREMIUM.CLASS                  Standard
T BENEFIT.DESC                   Flexible Whole Life
T TERMINATION.PERIOD             Life
T ANNUAL.MAX.1                   $166.25
T ANNUAL.MAX.2                   $205.25
T PREMIUM.PERIOD                 3 Years Thereafter to Jan 17, 2048
T ANNUAL.PERIOD                  $166.25
T DMG.GCPY.INSURED               01
T DMG.GCPY.FILE                  01
T DMG.DD.INSURED                 INSURED
T DMG.DD.FILE                    FILE
T DMG.SKEY.INSURED               01Brandon M. Williams -
T DMG.SKEY.FILE                  021-966-3452
E
* -  - - BEGIN MERGE SET NUMBER 0000002
L INSURED
  LM.LIFE.DEC                    00000      DUP SEP MAI STA ODD POR OFF                    BSS
  LM.LIFE.OVERLAY                00001      DUP SEP MAI STA ANY POR OVL                    CSS
  LM.LIFE.CONTRACT               00000      DUP SEP MAI STA ANY POR ON                     CSS
  LM.LIFE.ENDORSE2               00001      DUP CON MAI STA ANY POR ON                     CSS
L FILE
  LM.LIFE.DEC                    00000      SIM SEP MAI STA ODD POR OFF                    BSS
  LM.LIFE.CONTRACT               00000      SIM SEP MAI STA ODD POR OFF                    CSS
  LM.LIFE.ENDORSE2               00001      SIM SEP MAI STA ANY POR OFF                    CSS
T NAME                           Rachel F. Scott
T AGE                            37
T SEX                            Female
T POLICY.NUMBER                  1-085-9833
T DMG.MERGESET.ID                1-085-9833
T POLICY.DATE                    Dec 15 1983
T SUM.INSURED                    $ 20,000
T PREMIUM.CLASS                  Standard
T BENEFIT.DESC                   Flexible Whole Life
T TERMINATION.PERIOD             Life
T ANNUAL.MAX.1                   $134.25
T ANNUAL.MAX.2                   $176.50
T PREMIUM.PERIOD                 3 Years Thereafter to Dec 14, 2047
T ANNUAL.PERIOD                  $145.00
T DMG.GCPY.INSURED               01
T DMG.GCPY.FILE                  01
T DMG.DD.INSURED                 INSURED
T DMG.DD.FILE                    FILE
T DMG.SKEY.INSURED               01Rachel F. Scott     -
T DMG.SKEY.FILE                  021-085-9833
E
```

Return Codes

Return codes from VRFDEBLD

VRFDEBLD sets the following return codes. If you get a non-zero return code, it means that VRFDEBLD failed for some reason. You should fix the problem and rerun VRFDEBLD; do not use the flat file if you get a non-zero return code from VRFDEBLD.

0	All is OK; the flat file is good
40	Data length mismatch between DMG.FLST and DMG.OPT for a Group name. The input VRF is bad. Determine how this VRF was created; especially, how were the DMG.FLST and DMG.OPT tags created. Normally, DMG.FLST and DMG.OPT tags should be created only by the DMGRFMT program called by a VDR.
1nn	(Return code range 100 to 199.) Error opening the OUTPUT file. Program ISISEQIO issued return code nn trying to open the OUTPUT file. Verify the JCL for the OUTPUT file.
2nn	(Return code range 200 to 299.) Error processing the VRF. Program DMGVRFRD issued return code nn trying to process the VRF. Verify the JCL for the DMGVRF1 and DMGVRFA files. If no DMGVRFA file, code **`DMGVRFA=NO`** in the EXEC PARM.
3nn	(Return code range 300 to 399.) Error writing to the OUTPUT file. Program ISISEQIO issued return code nn trying to write to the OUTPUT file. If return code 308, this is a data truncation error; increase the LRECL of the OUTPUT file. If return code 320, then the OUTPUT file is full; allocate more disk space.

Return Codes from VRFBUILD

Do not use the generated VRF or VRFA file if you get a non-zero return code from VRFBUILD. If you get a non-zero return code, then VRFBUILD failed for some reason. Fix the error and rerun VRFBUILD.

VRFBUILD sets the following return codes:

0	All is OK; DMGVRF1 file and optional DMGVRFA file created.
12	Error in SYSIN flat file input; see ISIWTL file for error message.
16	Error writing to file DMGVRF1 or DMGVRFA. See ISIWTL file for error message.

Notes and TIPS

- If you use too small an LRECL, VRFDEBLD will quit with return code 308. If this happens, rerun with an LRECL of at least 96 (100 if variable length).
- Use fixed length records (RECFM=F or FB) if reading the flat file with a COBOL program. If you use variable length records instead, you might process garbage because COBOL does not give you the size of the record.
- Program VRFDEBLD generates the T records as follows:
  - The T record itself contains the first 38 characters of the data.
  - If there are more than 38 characters, then continuation records contain the remaining data, 70 characters per record.
  - A hyphen is generated only if the data ends with two hyphens.

  The maximum number of characters-per-card is 200. However, we recommend you not exceed 80 characters for T cards and their continuation cards, to make it easier to view and edit these cards online (with TSO or other editors).

NOTE

In Documerge 3.x, the continuation cards for the forms and options (following the L card) will exceed 80 characters due to the number of print options.

- Take care when editing print options (in a continuation card following an L card), because:
  - The record often exceeds 80 characters, so there might be data that is not displayed on your 80-character-wide screen.
  - Print options are positional. If you change an option, make sure you don't shift any other options out-of-position.
- Take care when making mass-changes with your editor.
  - For example, if you want to change all Overlay options from "ON" to "OFF", do not simply issue a mass change from "ON" to "OFF", because this would change any characters "ON" to "OFF", not just for the Overlay option. (Also, in this example, you would not want to change a 2-character option "ON" into a 3-character option "OFF", because this would shift the subsequent options to the right; instead, change "ON "to "OFF" — put a blank after "ON". Remember that all options are 3 characters long followed by a blank.)
  - Remember that a trailing hyphen in a T record or its continuation record is not the same as a hyphen in the DMGMERGE SYSIN or other SYSIN records. The trailing hyphen in the T record marks the end of the data for this record (but not necessarily the end of the data for the tag), so data to the left of the trailing hyphen is not truncated. The trailing hyphen is needed only if the data record requires trailing blanks or a trailing hyphen.
  - Ensure that automatic numbering is turned off in your TSO or other edit program. VRFBUILD uses all columns of the SYSIN input. So, editors cannot put any sequence numbers or other information into columns 73-80 or other columns.
- Generally, the sequence of tags is not important. However, it is important for:
  - Tags used by BPSDs that specify DELETE=Y.
  - Some "DMG." special tags, such as DMG.SRC and DMG.LPG.
- VRFBUILD creates only the print options listed in the continuation cards following the L card. If you don't code any of the print options (if these values are blank), then the resulting VRF won't have them either. This is important if you want to use VRFBUILD to create a 2.0 or 2.1 VRF, which of course don't support the 3.x print options. Often, the last print option you need is the Overlay option.
- Some print options can be changed only if you change corresponding special tags. Examples of this are options SRC (uses the DMG.SRC tag), BMP (used the DMG.IDEF tag) and BLP (uses the DMG.LPG tag). In general, if you don't understand all of the rules about a print option, then don't change it. Otherwise, DMGMERGE could abend. or do other undesirable stuff.
- If you get a DMGMERGE P-Level error using a VRF created with VRFBUILD, review the special tags — the ones that begin with "DMG". If these are wrong, this can cause DMGMERGE to abend.

Documerge Reserved Tags

Documerge predefines the names of special tags, called Reserved Tags, for specific purposes. Some Reserved Tags give you information that helps you audit and verify the Documerge run. Others tell the DMGMERGE program to perform certain tasks at specific points in the run.

All Reserved Tag names begin with the **DMG.** prefix.

IMPORTANT!

Oracle reserves the **DMG.** prefix for Documerge Reserved Tags. Do not code the **DMG.** prefix for any tag name, or use a **DMG.** tag for any purpose other than its documented use.

Oracle reserves the right to use any tag that begins with **DMG.** in any future release of Documerge (Documaker FP).

The values of some Reserved Tags give you processing information when they're printed in the Documerge output. The names of these Reserved Tags are intended to be coded in Boilerplate Space Definitions (BPSDs) during form composition. DMGMERGE inserts their values into the Boilerplate Spaces defined by the BPSDs.

NOTE

For more information about the hierarchy of the tag lookup sequence, see the list on page 32.

For example, assume that you designated a certain page of a form as an audit page, and that you coded a BPSD specifying the DMG.TOTAL.SHEETS Reserved Tag on that page. When DMGMERGE reads this Reserved Tag it calculates the total number of sheets in the Document Package and inserts that number into the Boilerplate Space. When the Document Package is printed this number appears on the audit page, and can be used by finishing equipment or other post-production processes.

Other Reserved Tag values are processing instructions for DMGMERGE. These values are intended for program control; there is no practical reason for them to appear in printed Documerge output. Therefore, these Reserved Tag names are not intended to be coded in BPSDs.

For example, the value of DMG.FLST.*Groupname* is a complete list of the forms that you want DMGMERGE to include in a Document Package for a Group. Although you could print that value on a form, you probably wouldn't want to; usually it would appear only in the VRF.

Reserved Tags are useful diagnostic tools for troubleshooting. For example, suppose that you included the DMG.ERROR.VRF Reserved Tag in a Merge Set. If any Group in that Merge Set had an error DMGMERGE would write the entire Merge Set to the file you selected as the value of DMG.ERROR.VRF. Then you could view that file to diagnose and correct the error.

There are two types of Reserved Tags:

- User-selected
- Internal (program-generated)

User-Selected Reserved Tags

A user-selected Reserved Tag is a Reserved Tag with a value you determine. The DMGVRFWR subprogram writes these Reserved Tags by Merge Set to the VRF. You include a user-selected Reserved Tag in a Merge Set by specifying the Reserved Tag's name and value in either

- A Rulebase Library Tag Table
- A VDR call to DMGVRFWR

A user-selected Reserved Tag's value is a string of variable data in the VDR Merge Set Record. You can assign a unique data string to the Reserved Tag. Or, you can assign a data string that is assigned also to another tag in the Rulebase Library Tag Table.

For example, suppose that you wanted to include the DMG.MERGESET.ID Reserved Tag in a particular Merge Set as a cross-reference to the Merge Set's printed Document Packages. Also suppose that you wanted to use the policy number as the Merge Set's identification number.

A regular Documerge tag called `POLICY.NUMBER` is coded in the Tag Table:

```
TAG NAME=POLICY.NUMBER                    POS=45       LEN=10
```

This tag's value (the policy number) is specified as a 10-character data string beginning in position 45 of the VDR Merge Set Record. You could give the Merge Set's identification number this same value by coding DMG.MERGESET.ID as the first tag in the Tag Table and selecting the same parameters as those of `POLICY.NUMBER`:

```
TAG NAME=DMG.MERGESET.ID                  POS=45       LEN=10
TAG NAME=POLICY.NUMBER                    POS=45       LEN=10
```

DMGRFMT generates DMG.MERGESET.ID with the same value as that of `POLICY.NUMBER`, and DMGVRFWR writes it to the VRF. Then DMGMERGE inserts the policy number into the Boilerplate Space defined by the BPSD that specifies DMG.MERGESET.ID. This value in a printed Document Package helps you find the Document Package's Merge Set in the VRF — to diagnose an error, for example.

Internal Reserved Tags

An internal Reserved Tag is a Reserved Tag whose value is generated automatically by one of the following Documerge programs:

- The DMGRFMT subprogram
- The DMGMERGE program

> **NOTE**
>
> Knowledge of internal Reserved Tags is necessary if you plan to write your own VRF in order to bypass the Rulebase Library.

Internal Reserved Tags Generated by DMGRFMT

DMGRFMT automatically generates some Reserved Tag values that are instructions to DMGMERGE. The DMGVRFWR subprogram writes them by Merge Set to the VRF.

> **IMPORTANT!**
>
> Do not write these Reserved Tags to the VRF yourself, either through the Rulebase Library or in VDR calls to DMGVRFWR, unless you are intentionally bypassing the Rulebase Library by eliminating VDR calls to DMGRFMT.
>
> You should not code the names of these Reserved Tags in BPSDs.

For each Merge Set, DMGRFMT automatically generates Reserved Tags that give DMGMERGE

- The names of implicit, explicit, and inline forms in each Document Package
- The Print Options for the implicit and explicit forms
- A list of any implicit and explicit forms that are specified for the Document Package but missing from the EDL
- A list of the Groups receiving each Document Package
- Imposition Definitions for Groups
- Logical Page Definitions for Groups
- Sort keys for the DMGSORT program

Internal Reserved Tags Generated by DMGMERGE

DMGMERGE computes some Reserved Tag values dynamically. These values should exist only in memory; they should not be written to the VRF. Therefore, you should not code them in Rulebase Library Tag Tables or in VDR calls to DMGVRFWR.

IMPORTANT!

If an internal Reserved Tag generated by DMGMERGE exists also in the VRF, the value in the VRF overrides the value in memory.

You can code these Reserved Tag names in BPSDs to print the values in the DMGMERGE output. These values give you information that you can use to audit processing, such as the current date or the number of sheets in a Document Package.

See "Reserved Tag Quick-Reference Table" on page 293 to determine the names of the Reserved Tags.

BPSD Coding for Reserved Tags

The following example shows the standard BPSD coding syntax for a Documerge Reserved Tag. Assume that the Reserved Tag's value is 10.

The BPSD coding

```
:BPSD NAME='DMG.TOTAL.SHEETS' LENGTH=5 CHAR=@
```

generates the following Boilerplate Space and Replacement Characters:

```
@@@@@
```

After the DMGMERGE step, the printed appearance of the Reserved Tag's value in the Boilerplate Space depends on the following conditions:

- If the RESERVEDRT control card of the MERGE command in DMGMERGE is not specified the value is printed as

```
10
```

(The tag data is left justified, with three trailing spaces.)

- If the RESERVEDRT control card is specified, the value is printed as

```
00010
```

(The tag data is right justified, with three leading zeros.)

Refer to "RESERVEDRT" on page 439 for more information.

IMPORTANT!

In a Boilerplate Space that receives the value of a Reserved Tag, the string of Replacement Characters must not be broken; other characters or Metacode or AFP control information must not exist within the Replacement Character string. Otherwise, each resulting sub-string of Replacement Characters receives the entire value.

Do not code a Reserved Tag on any form requiring imposition printing or on a Metacode form requiring tumble printing.

See "Reserved Tag Quick-Reference Table" on page 293 to determine the names and characteristics of Reserved Tags.

Using the DMG.CHECKPOINT Reserved Tags

While processing, the DMGMERGE 3.x program will query certain Reserved Tags before or after certain events or *checkpoints*. All of these checkpoint tags start with **DMG.CHECKPOINT...**, and this guide provides detailed explanations of all of them in "Checkpoint Reserved Tag Processing Rules" on page 287.

You can use the checkpoint Reserved Tags to

- Dynamically capture data about certain events during DMGMERGE processing.

  For example, as alternative to using the STATSFILE parameter, you could create a log file for the forms used in a DMGMERGE run based on the values of counters you programmed for the DMG.CHECKPOINT.START.FORM tag.

- HOldFurther process stored counts or switch values.

  For example, you could program a DMG.CHECKPOINT.END.FORM tag to call a DMG.C.xxx user exit program after DMGMERGE has read the EOFs (end-of-files) for all forms processed for a Document Package.

NOTE

Checkpoint Reserved Tags are also used for internal debugging by Oracle personnel. DMGMERGE uses the DMGLMMM program as its list manager/memory manager. Oracle offers a DMGLMDMP program for dumping the data generated by DMGLMMM.

You can use the TAG command to execute this dump.

For example:

TAG=(DMG.CHECKPOINT.END.MERGEDAT 'CDMGLMDMP')

causes DMGLMMM to display its data after merging in the BPSDs.

The DMGLMDMP program requires the specification of an output file that has the DD name ***LMTRACE***. Because this can be a large file, ***LMTRACE*** should be written to disk.

Generally, the contents of the LMTRACE file are not meaningful except to Oracle programmers. Therefore, unless requested by Oracle, you should not use this tracing facility.

Checkpoint Reserved Tag Processing Rules

Here are the checkpoint tag processing rules:

- For the checkpoint tags to work, the applicable Document Package must contain at least one form. Forms do not have to have any BPSDs, or be imposed or tumbled, or contain any sheet Overlays.
- If the Document Package being processed does not cause DMGMERGE to perform certain steps, it will skip the look ups of the checkpoint tags associated with those steps.

  For example, if it does not process any sheet Overlays, DMGMERGE does not look up the DMG.CHECKPOINT.START.SHEETOVL tag.

NOTE

If GLOBAL ALLERROR=Y (or ALLERROR=YES), all the tags up to and including DMG.CHECKPOINT.START.COMBNRN are looked up for each Group in the Merge Set. Then, all following tags are looked up for each Group in the Merge Set.

- As it processes a Document Package, DMGMERGE queries each tag in the following sequence:

Tag Name	Value
DMG.CHECKPOINT.START.PACKAGE	Used as a Document Package (a Group within a Merge Set) is just starting to process. This is also the start of the COLLATION process. This tag is always looked up. **NOTE:** If the Group copy count (DMG.GCPY.groupname tag) is zero, then this tag (nor any DMG.CHECKPOINT tags) is looked up for this Group.
DMG.CHECKPOINT.START.FORM	Used before getting a new form (e.g., EDL form, Overlay, or Dynacomp or DocuWord exit form) for processing. **TIP:** Use item DMG.CURRENT.FORM to determine the form name. This tag is looked up once per form. For Overlays, this tag is looked up each time the Overlays is placed on the page.
DMG.CHECKPOINT.END.FORM	Used as the EOF (end-of-file) for each form (e.g., EDL form, Overlay, or Dynacomp or DocuWord exit form) occurs. **TIP:** Use item DMG.CURRENT.FORM to determine the form name. This tag is looked up once per form. For Overlays, this tag is looked up each time the Overlays are placed on the page.
DMG.CHECKPOINT.END.COLLATION	Used after all forms have been assembled in the proper sequence. This tag is always looked up for each Document Package. At this point, the following tags are complete and available: ▪ DMG.TOTAL.SHEETS ▪ DMG.TOTAL.PAGES
DMG.CHECKPOINT.START.MERGEDAT	Used before BPSD tag processing starts (applies also to BPSD tags filled for sheet Overlays, but does not apply for VSD tags because they fill at collation time). This tag is looked up only if at least one form had a BPSD. **NOTE:** The MERGEDAT process can be repeated during sheet Overlay processing.

Tag Name	Value
DMG.CHECKPOINT.END.MERGEDAT	Used as BPSD tag processing finishes. This tag is looked up only if at least one form had a BPSD.
DMG.CHECKPOINT.START.IMPOSE	(Metacode and AFP only) Used before starting imposition processing. This tag is looked up only if the Document Package has any imposition requests (IMPOSE in the Rulebase).
DMG.CHECKPOINT.END.IMPOSE	(Metacode and AFP only) Used as imposition processing finishes. This tag is looked up only if the Document Package has any imposition requests (IMPOSE in the Rulebase).
DMG.CHECKPOINT.START.SHEETOVL	(Metacode and AFP only) Used at the start of sheet Overlay processing, if any sheet Overlays were requested. Sheet Overlay processing repeats the COLLATE and MERGEDAT process. With any sheet Overlays, the following tags will be looked up again: ■ DMG.CHECKPOINT.START.FORM (for each sheet Overlay form) ■ DMG.CHECKPOINT.END.FORM (for each Overlay form) ■ DMG.CHECKPOINT.START.MERGEDAT (called only once, if any sheet Overlays had a BPSD) ■ DMG.CHECKPOINT.END.MERGEDAT (called only once, if any sheet Overlays had a BPSD)
DMG.CHECKPOINT.END.SHEETOVL	(Metacode and AFP only) Used as all sheet Overlay processing ends.
DMG.CHECKPOINT.START.TUMBLE	(Metacode only) Used before tumble print processing starts. This tag is looked up if any sheets need to be tumbled. Tumbling can be requested directly (TUM option) or indirectly via landscape imposition.
DMG.CHECKPOINT.END.TUMBLE	(Metacode only) Used as tumble print processing finishes. This tag is looked up if any sheets need to be tumbled.
DMG.CHECKPOINT.START.COMBNRN	(AFP only) Used at the start of a routine that combines multiple AFP pages into one page. This tag is always looked up for AFP processing.
DMG.CHECKPOINT.START.WRITEOUT	Used before writing the print records to the output file. This tag is always looked up. **TIP:** When GLOBAL ALLERROR=Y, it is possible to set item CALLRETC to minus one to ignore a Document Package for a Group based on information from a subsequent Group in the Merge Set determined from one of the above checkpoint exits. For example, a user exit for DMG.CHECKPOINT.END.COLLATE might capture the number of sheets of paper for GROUP-2, save this value, and then later decide to ignore GROUP-1 based on this value.

Tag Name	Value
DMG.CHECKPOINT.END.PACKAGE	Populated after a Document Package finishes processing and has been written to the selected output file. **CAUTION:** If there is any error such as truncation of the **T** command value, or a non-zero return code from a user exit program, it will not affect the disposition of the package. However, you may be able to detect errors in the package using one or more of the following: ■ If ERRMSG=YES in the DMGMERGE SYSIN (which Oracle strongly recommends), the error message will show in the MESSAGE file , but will not show in the ERRDDN file. However, it is too late to set the special all hex 'FF' return code to tell DMGMERGE to ignore the package. ■ Check DMG.HIGHEST.ERROR.LEVEL (at other checkpoints, processing is not complete for the package and more errors could still follow). However, setting the return code (CALLRECT parameter) to **-1** has no affect because the package has already been written to the output file.

The DMGMERGE Process and Flow for Reserved Tags

DMGMERGE calls the appropriate DMGPRNT program (DMGPRNTX=Metacode, DMGPRNTA=AFP, DMGPRNT2=line printer) once for each copy for each Group in a Merge Set. We refer to each of the copies for a Group as a ***Document Package***, for which the following process occurs:

(1) Start Document Package. Look up tag DMG.CHECKPOINT.START.PACKAGE.

(2) Start collation. No checkpoint tag is looked up; use DMG.CHECKPOINT.START.PACKAGE as the "start collation" checkpoint.

(3) Place a form (or part of a form if it spans multiple pages) on the page. At the start of the form, tag DMG.CHECKPOINT.START.FORM is looked up. At the end of the form, tag DMG.CHECKPOINT.END.FORM is looked up.

If there is an Overlay form to process, DMG.CHECKPOINT.START.FORM can be looked up more than once before the DMG.CHECKPOINT.END.FORM is looked up. This is because Overlays "interrupt" the current form's process.

If there is a dynamic form to process, several of the count-reporting reserved tags might contain incorrect data because DMGMERGE processes a dynamic form using the first eligible clean file. For more details about this behavior, see "Limitations on the Use of Reserved Tags when Processing Dynamic Forms" on page 291.

Repeat step (3) for all forms in the Document Package.

(4) End collation. Look up tag DMG.CHECKPOINT.END.COLLATION.

At this point, DMGMERGE has determined the total pages, total sheets, and final clean DD file name for the Document Package.

(5) If any forms so far had a BPSD, then do the MERGEDAT routine:

- Look up tag DMG.CHECKPOINT.START.MERGEDAT
- Perform all merging
- Look up tag DMG.CHECKPOINT.END.MERGEDAT

(6) For Metacode and AFP, if any imposition is requested:

- Look up tag DMG.CHECKPOINT.START.IMPOSE
- Perform the imposition
- Look up tag DMG.CHECKPOINT.END.IMPOSE

(7) For Metacode and AFP, if any sheet Overlays are requested:

- Look up tag DMG.CHECKPOINT.START.SHEETOVL
- Place each sheet Overlay on the proper page. For each sheet Overlay, look up DMG.CHECKPOINT.START.FORM and DMG.CHECKPOINT.END.FORM.
- If any sheet Overlays had any BPSD, then do the MERGEDAT routine (again):
  - Look up tag DMG.CHECKPOINT.START.MERGEDAT
  - Perform all merging
  - Look up tag DMG.CHECKPOINT.END.MERGEDAT
- Look up tag DMG.CHECKPOINT.END.SHEETOVL

(8) For Metacode, if any tumbling, then:

- Look up tag DMG.CHECKPOINT.START.TUMBLE
- Perform the tumbling. If the PRINTDEF does not indicate that the INVERT DJDE is supported, call program DMGTUMBL to do the actual tumbling. If the INVERT DJDE is supported, generate the proper INVERT= DJDE parameters.
- Look up tag DMG.CHECKPOINT.END.TUMBLE

(9) For AFP:

- Look up tag DMG.CHECKPOINT.START.COMBNRTN
- Combine multiple pages into one page. An example of a multiple page is concatenation.
- Loop up tag DMG.CHECKPOINT.END.COMBNRTN

(10) Write to the desired output file:

- Write to the desired output file. Look up tag DMG.CHECKPOINT.START.WRITEOUT.
- End processing this Document Package. Look up tag DMG.CHECKPOINT.END.PACKAGE.

NOTE

If GLOBAL ALLERROR=Y, steps 1 through 9 are done for each Group in the Merge Set, then step 10 is done for each Group.

Limitations on the Use of Reserved Tags when Processing Dynamic Forms

DMGMERGE takes TAG parameters defined in FILEDEFs from the final clean file it selects for BPSD processing. However, when processing dynamic forms such as Docuword or Dynacomp forms with Variable Space Definitions (VSDs), DMGMERGE uses the first eligible clean file because dynamic forms are built during the collate step, when the final clean file is not yet known.

Therefore, those Documerge reserved tags that report line, page, single-sided-image (SSI), and sheet counts might not contain accurate data. These "count" reserved tags include the following:

- DMG.CURRENT.SHEET.COUNT
- DMG.LINE.COUNT
- DMG.SHEET.COUNT
- DMG.SHEET.NUMBER
- DMG.SSI.COUNT
- DMG.TOTAL.PAGES
- DMG.TOTAL.SHEETS

Additionally, when a dynamic form is processed for a Document Package

- The DMG.CURRENT.*ddname* reserved tag will contain the name of the first (not last) eligible clean file.
- Tag parameters defined in the FILEDEF for the first eligible clean file will be used.

The exceptions are *dynamic sheet Overlays and dynamic forms used as sheet Overlays. Because these are rendered later in the process*, DMGMERGE will use TAG parameters from the final clean file, and all "count" reserved tags will contain accurate data.

Reserved Tags Quick Reference

The table beginning on page 293 presents a quick reference to Reserved Tag information. The table does not contain full descriptions, but it includes a cross-reference to that information for each Reserved Tag. The following list explains the table headings:

Reserved Tag Name	The name of the reserved tag and the page on which you can find a full description.
BPSD	Indicates whether the Reserved Tag name is intended for coding in BPSDs so that its value appears in printed Documerge output. **YES** Intended for BPSD coding. The value gives you processing information when printed. **NO** Not intended for BPSD coding. The value is program control information. This value can be printed; however, it has no practical purpose when printed. **X** Must not be coded in a BPSD. This is a hexadecimal value and is unprintable.
Type	Indicates whether the Reserved Tag is user-selected or internal.
Generated By	The Documerge program that generates the value of the reserved tag: **DMGMERGE** The value is generated automatically during the DMGMERGE step. **VDR** The value is generated only if you code the Reserved Tag in a Rulebase Library Tag Table or if your VDR calls the DMGVRFWR subprogram to write the Reserved Tag. **DMGRFMT** The value is generated automatically by the Documerge Reformatter (DMGRFMT). You should not see this value in any printed output from Documerge.
Audit	Indicates whether the Reserved Tag value gives you auditing information, such as the number of Document Packages, sheets and lines for a particular output file.
Dash Code	Indicates whether the Reserved Tag value can be generated as dash codes to provide information to an optical scanner. See "Reserved Tags and Dash Codes" on page 326 for more information.
RESERVEDRT	Causes data in the specified Reserved Tag to be right justified, with leading zeros and no commas (some finishing equipment requires this format), when merged into the BPSD on the form. See "RESERVEDRT" on page 439 for more information.

Reserved Tag Quick-Reference Table

Reserved Tag Name	BPSD	Type*	Generated By *	Audit	Dash Code	RESERVEDRT
"DMG.ALT.CURRENT.DATE" **on page 295**	Yes	U	VDR	—	—	—
"DMG.BENCHMARK" **on page 295**	Yes	U	*	—	Y	—
"DMG.BNR.Groupname" **on page 296**	No	U	VDR	—	—	—
"DMG.BTEXT.SEQ" **on page 295**	Yes	I	*	—	—	—
"DMG.C.xxx" **on page 296**	Yes	U	VDR	—	—	—
"DMG.CHECKPOINT.END.COLLATE" **on page 296**	No	U	VDR	—	—	—
TIP: Use DMG.CHECKPOINT.START.PACKAGE to initialize the collation start (there is no DMG.CHECKPOINT.START.COLLATE tag).						
"DMG.CHECKPOINT.END.FORM" **on page 296**	No	U	*	Y	—	—
"DMG.CHECKPOINT.END.IMPOSE" **on page 297**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.END.MERGEDAT" **on page 297**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.END.PACKAGE" **on page 297**	No	U	*	Y	—	—
"DMG.CHECKPOINT.END.SHEETOVL" **on page 299**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.END.TUMBLE" **on page 299**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.START.FORM" **on page 299**	No	U	*	Y	—	—
"DMG.CHECKPOINT.START.IMPOSE" **on page 300**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.START.MERGEDAT" **on page 300**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.START.OUTPUT.FILE" **on page 300** **TIP:** Use DMG.CHECKPOINT.END.PACKAGE to get ending counts for an output file (there is no DMG.CHECKPOINT.END.OUTPUT.FILE tag).	No	U	*	Y	—	—
"DMG.CHECKPOINT.START.PACKAGE" **on page 301**	No	U	*	Y	—	—
"DMG.CHECKPOINT.START.SHEETOVL" **on page 302**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.START.TUMBLE" **on page 303**	No	U	VDR	—	—	—
"DMG.CHECKPOINT.START.WRITEOUT" **on page 303**	No	U	VDR	—	—	—
"DMG.CURRENT.DDNAME" **on page 303**	No	I	*	—	—	—
"DMG.CURRENT.FORM" **on page 304**	No	U	*	Y	—	—
"DMG.CURRENT.GROUP" **on page 304**	No	I	*	—	—	—

Reserved Tag Name	BPSD	Type*	Generated By *	Audit	Dash Code	RESERVEDRT
"DMG.CURRENT.SHEET.COUNT" on page 305	Yes	I	*	—	Y	Y
"DMG.DATE" on page 305	Yes	I	*	—	—	—
"DMG.DD.Groupname" on page 305	No	U	VDR	—	—	—
"DMG.END.OF.SET" on page 306	Yes	I	*	—	—	—
"DMG.END.OF.SUBSET" on page 306	Yes	I	*	—	—	—
"DMG.ERDD.Groupname" on page 306	No	U	VDR	—	—	—
"DMG.ERROR.VRF" on page 307	No	U	VDR	—	—	—
"DMG.FDEF.Groupname" on page 308	No	U	VDR	—	—	—
"DMG.FLST.Groupname" on page 309	No	I	*	—	—	—
"DMG.FONT.SUBSTITUTION.AFP" on page 309	No	U	VDR	—	—	—
"DMG.FONT.SUBSTITUTION.META" on page 312	No	U	VDR	—	—	—
"DMG.GCPY.Groupname" on page 314	Yes	U	VDR	—	—	—
"DMG.HASHMARK" on page 314	Yes	I	*	—	Y	—
"DMG.HEX8BCC.COUNT" on page 314	Yes	I	*	—	—	Y
"DMG.HIGHEST.ERROR.LEVEL" on page 314	No	U	*	Y	—	—
"DMG.IDEF.Groupname" on page 315	X	I	DMGRFMT	—	—	—
"DMG.ITEM.COUNT.VERIFY" on page 315	Yes	I	*	—	Y	Y
"DMG.LINE.COUNT" on page 315	Yes	I	*	Y	—	Y
"DMG.LPG.Groupname" on page 316	X	I	DMGRFMT	—	—	—
"DMG.MDEF.Groupname" on page 316	No	U	VDR	—	—	—
"DMG.MERGESET.ID" on page 316	Yes	U	VDR	—	—	—
"DMG.MISSING.FORMS" on page 316	No	I	DMGRFMT	—	—	—
"DMG.OPT.Groupname" on page 317	No	I	DMGRFMT	—	—	—
"DMG.PAGE.NUMBER" on page 319	Yes	I	*	—	—	—
"DMG.PDEF.Groupname" on page 319	No	U	VDR	—	—	—
"DMG.POL.COUNT" on page 319	Yes	I	*	Y	—	Y
"DMG.PRA" on page 320	Yes	U	VDR	—	—	—
"DMG.PRA.Groupname" on page 320	Yes	U	VDR	—	—	—
"DMG.PRODUCT" on page 320	Yes	I	*	—	—	—
"DMG.SET.NUMBER" on page 321	Yes	I	*	—	Y	—
"DMG.SHEET.COUNT" on page 321	Yes	I	*	Y	Y	Y
"DMG.SHEET.NUMBER" on page 321	Yes	I	*	—	Y	Y
"DMG.SKEY.Groupname" on page 321	No	U	VDR	—	—	—
"DMG.SRC.Groupname" on page 322	X	I	DMGRFMT	—	—	—

Reserved Tag Name	BPSD	Type*	Generated By *	Audit	Dash Code	RESERVEDRT
"DMG.SSI.COUNT" on page 323	Yes	I	*	Y	Y	Y
"DMG.TIME" on page 323	Yes	I	*	Y	—	—
"DMG.TLR.Groupname" on page 323	No	U	VDR	—	—	—
"DMG.TOTAL.PAGES" on page 323	Yes	I	*	Y	—	—
"DMG.TOTAL.SHEETS" on page 323	Yes	I	*	Y	Y	Y
"DMG.TXT" on page 324	Yes	U	VDR	—	—	—
"DMG.TXT.Groupname" on page 324	Yes	U	VDR	—	—	—
"DMG.VDR.ERRORS" on page 324	No	U	VDR	—	—	—
"DMG.VERSION" on page 326	Yes	I	*	Y	—	—

TIP

In the Type column, “I” represents an Internal reserved tag, while “U” represents a User-selected tag.

In the Generated By column, the Reserved Tag is generated by DMGMERGE unless specified otherwise.

Reserved Tag Descriptions

This section describes the Documerge Reserved Tags. Refer to "BPSD Coding for Reserved Tags" on page 285 and to *Composing Forms for Documerge* for the coding syntax of Reserved Tags.

DMG.ALT.CURRENT.DATE

(PROGRAM CONTROL; USER-SELECTED)

Contains the current date in **mm/dd/yyyy** format for Year 2000 support. Documerge reports will continue to have 2-digit years in their headings.

DMG.BENCHMARK

(PRINT RELATED; USER-SELECTED)

Documerge places a dash code, or hash mark, in every occurrence of this tag in a Document Package except the first and last.

This tag works with Bell & Howell AIM equipment. It gives the same results as DMG.HASHMARK.

DMG.BTEXT.SEQ

(PRINT RELATED; INTERNAL DMGMERGE)

Contains the SEQ count that is generated for BTEXT processing. DMGMERGE adds 1 to the SEQ sequence counter for each clean Document Package, and then generates a DJDE BTEXT SEQ record that contains the SEQ counter total.

When using the DMG.BTEXT.SEQ reserved tag, follow these guidelines:

- The minimum LENGTH specification is 1 (it must not be zero), and the maximum length specification is 5 with a Xerox specified maximum value of 65534.
- Compose the forms to use for BTEXT processing so that the DMG.BTEXT.SEQ reserved tag prints in the correct position with the correct font for processing by the Xerox 4635 printer bar code scanner. The BTEXT form or forms can be regular forms or Overlays.

DMG.BNR.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

Contains the user-specified EDL form name and revision level to be used instead of the Documerge-generated banner page. This tag generates a form at the beginning of the **clean output file** selected by DMGMERGE. The revision level is optional, with a default of the highest revision level at the time of the DMGMERGE step.

Maximum length of tag value is

- 32-character EDL member name
- 5-character revision level
- 5 blanks

Values can be generated by

- The EDL member name and revision level
- DMGMERGE BANNER= control card. The EDL member name is coded as a Merge Group control card. Example:

```
MERGE GROUP=XXXXXXXXX -
BANNER='VLAM2.EDL.MEMBER.NAME'(REV#)
```

The BANNER= control card value overrides any VDR-generated value. The value in the DMGMERGE control card CHAIN= overrides all other values. This Reserved Tag is optional with a default of the Documerge banner sheet.

DMG.C.xxx

(PRINT RELATED; USER-SELECTED)

The DMG.C.*xxx* VRF value tells DMGMERGE how to build a character string that replaces a Boilerplate Space. DMGMERGE builds this character string dynamically.

See "DMG.C.xxx Reserved Tag Processing" on page 331 for more information.

DMG.CHECKPOINT.END.COLLATE

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up when all forms have been assembled in the proper sequence. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.END.FORM

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag used to call a DMG.C.xxx user exit program when DMGMERGE reads the EOF (end-of-file) for each form (e.g., EDL form, Overlay, or Dynacomp or DocuWord exit form) being processed for a Document Package. For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

The value of DMG.CHECKPOINT.END.FORM must be in DMG.C.xxx tag format, and the first character of the value must be a valid DMG.C.xxx command (T, L, or C). For details about DMG.C.xxx processing, see "DMG.C.xxx Value" on page 332.

Use the DMG.CHECKPOINT.END.FORM tag to report form usage statistics. For example, you could generate the following value for DMG.CHECKPOINT.END.FORM tag:

```
|...+....1....+....2....+....3....+....4..
T DMG.CURRENT.FORM              TAGLNOTLLY
```

This value in this example includes

(1) A DMG.C.xxx Tag command field (`T` in position 1) and the other fields (positions 32–41) required specify the processing of the current form name by the exit program (for details, see "The TAG Command" on page 335).

(2) The DMG.CURRENT.FORM reserved tag, which contains the current form name (for details, see "DMG.CURRENT.FORM" on page 304).

After processing the value, the exit program could then generate statistics about forms usage — including the form names used.

Following are some guidelines for using the DMG.CHECKPOINT.END.FORM reserved tag:

- The exit program associated with DMG.CHECKPOINT.END.FORM can set the CALLRETC return code to minus-one to stop tag processing and cancel writing of the Merge Set (for details about CALLRETC, see "Using the CALLRETC Return Code to Control Document Package Processing" on page 348).
- For flexibility, consider specifying the value of DMG.CHECKPOINT.END.FORM with the TAG parameter of the MERGE command. You can easily add or delete TAG parameters (and hence audit statistical processing) as needed, without making VDR or Rulebase changes.
- The value of the DMG.CHECKPOINT.END.FORM tag is available for processing user-defined banner or trailer pages. (The current Merge Set tags are available for banner page processing, and the final Merge Set tags are available for trailer page processing.)

  To determine if banner or trailer processing has been specified, the exit program can access the DMG.MERGESET.ID value for banner or trailer processing:

```
%%%BANNER%%%
%%%TRAILER%%%
```

  For details about the DMG.MERGESET.ID tag, see "DMG.MERGESET.ID" on page 316.

DMG.CHECKPOINT.END.IMPOSE

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up when it has ended imposition processing. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.END.MERGEDAT

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up when it has ended BPSD tag processing. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.END.PACKAGE

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag used to call a DMG.C.xxx user exit program when a Document Package (a Group within a Merge Set) ends processing. For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

The value of DMG.CHECKPOINT.END.PACKAGE must be in DMG.C.xxx tag format. The first character of the value must be a DMG.C.xxx command (T, L, or C). For details about DMG.C.xxx processing, see "DMG.C.xxx Value" on page 332.

Use this tag to report statistical totals in the audit of each Document Package of a Merge Set. For example, you could use a DMG.CHECKPOINT.END.PACKAGE tag containing the following value:

```
|...+....1....+....2....+....3....+....4..
T DMG.SET.NUMBER                 TAGLNOTLLY
T DMG.MERGESET.ID                TAGLNOTLLY
T DMG.CURRENT.GROUP              TAGLNOTLLY
T DMG.CURRENT.DDNAME             TAGLNOTLLY
T DMG.SHEET.COUNT                TAGLNOTLLY
T DMG.SSI.COUNT                  TAGLNOTLLY
T DMG.HIGHEST.ERROR.LEVEL        TAGLNOTLLY
```

To generate the following audit items:

(1) Logical VRF record number for the Merge Set

(2) Merge Set identifier

(3) Group Name

(4) Output JCL file name

(5) Number of sheets of paper required

(6) Number of single-sided images produced

(7) Highest error message level (if messages where produced)

Following are some guidelines for using the DMG.CHECKPOINT.END.PACKAGE reserved tag:

- The associated exit program should generate a zero return code so that CALLRETC is not set. If CALLRETC contains a non-zero value, tag processing stops (for details about CALLRETC, see "Using the CALLRETC Return Code to Control Document Package Processing" on page 348). However, DMGMERGE generates no error because it has completed error processing before it detects the DMG.CHECKPOINT.END.PACKAGE tag.

  The exit program can access the DMG.HIGHEST.ERROR.LEVEL tag to determine if the Document Package had any errors (for details, see ""DMG.HIGHEST.ERROR.LEVEL" on page 314.
- For flexibility, consider specifying the value of DMG.CHECKPOINT.END.PACKAGE with the TAG parameter of the MERGE command. You can easily add or delete TAG parameters (and hence audit statistical processing) as needed, without making VDR or Rulebase changes.
- The value of the DMG.CHECKPOINT.END.PACKAGE tag is available for processing user-defined banner or trailer pages. (The current Merge Set tags are available for banner page processing, and the final Merge Set tags are available for trailer page processing.)

  To determine if banner or trailer processing has been specified, the exit program can access the DMG.MERGESET.ID value for banner or trailer processing:

```
%%%BANNER%%%
%%%TRAILER%%%
```

For details about the DMG.MERGESET.ID tag, see "DMG.MERGESET.ID" on page 316.

DMG.CHECKPOINT.END.SHEETOVL

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up when it has ended all sheet Overlay processing. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.END.TUMBLE

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up as it ends tumble print processing. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.START.FORM

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag used to call a DMG.C.xxx user exit program each time DMGMERGE requests a new form (e.g., EDL form, Overlay, or Dynacomp or DocuWord exit form) for processing. For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

The value of DMG.CHECKPOINT.START.FORM must be in DMG.C.xxx tag format, and the first character of the value must be a valid DMG.C.xxx command (T, L, or C). For details about DMG.C.xxx processing, see "DMG.C.xxx Value" on page 332.

Use the DMG.CHECKPOINT.START.FORM tag to report form usage statistics. For example, you could generate the following value for DMG.CHECKPOINT.START.FORM tag:

```
|...+....1....+....2....+....3....+....4..
T DMG.CURRENT.FORM              TAGLNOTLLY
```

This value in this example includes

(1) A DMG.C.xxx Tag command field (T in position 1) and the other fields (positions 32–41) required specify the processing of the current form name by the exit program (for details, see "The TAG Command" on page 335).

(2) The DMG.CURRENT.FORM reserved tag, which contains the current form name (for details, see "DMG.CURRENT.FORM" on page 304).

The exit program could access this value to maintain form usage statistics.

Following are some guidelines for using the DMG.CHECKPOINT.START.FORM reserved tag:

- The exit program associated with DMG.CHECKPOINT.START.FORM can set the CALLRETC return code to minus-one to stop tag processing and cancel writing of the Merge Set (for details about CALLRETC, see "Using the CALLRETC Return Code to Control Document Package Processing" on page 348).
- For flexibility, consider specifying the value of DMG.CHECKPOINT.START.FORM with the TAG parameter of the MERGE command. You can easily add or delete TAG parameters (and hence audit statistical processing) as needed, without making VDR or Rulebase changes.
- The value of the DMG.CHECKPOINT.START.FORM tag is available for processing user-defined banner or trailer pages. (The current Merge Set tags are available for banner page processing, and the final Merge Set tags are available for trailer page processing.)

To determine if banner or trailer processing has been specified, the exit program can access the DMG.MERGESET.ID value for banner or trailer processing:

```
%%%BANNER%%%
%%%TRAILER%%%
```

For details about the DMG.MERGESET.ID tag, see "DMG.MERGESET.ID" on page 316.

DMG.CHECKPOINT.START.IMPOSE

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up just as it starts imposition processing. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.START.MERGEDAT

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up just as it starts BPSD tag processing (applies also to BPSD tags filled for sheet Overlays, but does not apply for VSD tags, because they filled at collation time). You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.START.OUTPUT.FILE

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag used to call a DMG.C.xxx user exit program when a Document Package (a Group within a Merge Set) finishes processing and is ready to be written to the selected output file. For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

The value of DMG.CHECKPOINT.START.OUTPUT.FILE must be in DMG.C.xxx tag format. The first character of the value must be a DMG.C.xxx command (T, L, or C). For details about DMG.C.xxx processing, see "DMG.C.xxx Value" on page 332.

Use this tag to reset statistical totals in the audit of each Document Package in a Merge Set. For example, you could use a DMG.CHECKPOINT.START.OUTPUT.FILE tag containing the following value:

```
|...+....1....+....2....+....3....+....4..
T DMG.SET.NUMBER                 TAGLNOTLLY
T DMG.MERGESET.ID                TAGLNOTLLY
T DMG.CURRENT.GROUP              TAGLNOTLLY
T DMG.CURRENT.DDNAME             TAGLNOTLLY
T DMG.SHEET.COUNT                TAGLNOTLLY
T DMG.SSI.COUNT                  TAGLNOTLLY
T DMG.HIGHEST.ERROR.LEVEL        TAGLNOTLLY
```

to reset the following audit items:

(1) Logical VRF record number for the Merge Set

(2) Merge Set identifier

(3) Group Name

(4) Output JCL file name

(5) Number of sheets of paper required

(6) Number of single-sided images produced

(7) Highest error message level (if messages where produced)

Following are some guidelines for using the DMG.CHECKPOINT.START.OUTPUT.FILE reserved tag:

- The exit program associated with DMG.CHECKPOINT.START.OUTPUT.FILE has the last opportunity to set the CALLRETC return code to minus-one to stop tag processing and cancel writing of the Merge Set (for details about CALLRETC, see "Using the CALLRETC Return Code to Control Document Package Processing" on page 348).
- For flexibility, consider specifying the value of DMG.CHECKPOINT.START.OUTPUT.FILE with the TAG parameter of the MERGE command. You can easily add or delete TAG parameters (and hence audit statistical processing) as needed, without making VDR or Rulebase changes.
- The value of the DMG.CHECKPOINT.START.OUTPUT.FILE tag is available for processing user-defined banner or trailer pages. (The current Merge Set tags are available for banner page processing, and the final Merge Set tags are available for trailer page processing.)

  To determine if banner or trailer processing has been specified, the exit program can access the DMG.MERGESET.ID value for banner or trailer processing:

```
%%%BANNER%%%
%%%TRAILER%%%
```

  For details about the DMG.MERGESET.ID tag, see "DMG.MERGESET.ID" on page 316.

DMG.CHECKPOINT.START.PACKAGE

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag used to call a DMG.C.xxx user exit program when a Document Package (a Group within a Merge Set) is just starting to process. For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

The value of DMG.CHECKPOINT.START.PACKAGE must be in DMG.C.xxx tag format, and the first character of the value must be a valid DMG.C.xxx command (T, L, or C). For details about DMG.C.xxx processing, see "DMG.C.xxx Value" on page 332.

Use this tag to reset statistical counters in the audit of each Document Package in a Merge Set. For example, you could use a DMG.CHECKPOINT.START.PACKAGE tag containing the following value:

```
|...+....1....+....2....+....3....+....4..
T DMG.SET.NUMBER                TAGLNOTLLY
T DMG.MERGESET.ID               TAGLNOTLLY
T DMG.CURRENT.GROUP             TAGLNOTLLY
```

To reset the following audit items:

(1) Logical VRF record number for the Merge Set

(2) Merge Set identifier

(3) Group Name

Following are some guidelines for using the DMG.CHECKPOINT.START.PACKAGE reserved tag:

- Because DMGMERGE has not yet provided their values, do not use the DMG.CURRENT.DDNAME and DMG.CURRENT.ERRDDN reserved tags for the clean and error file names.
- For flexibility, consider specifying the value of DMG.CHECKPOINT.START.PACKAGE with the TAG parameter of the MERGE command. You can easily add or delete TAG parameters (and hence audit statistical processing) as needed, without making VDR or Rulebase changes.The clean and error file names, DMG.CURRENT.DDNAME and DMG.CURRENT.ERRDDN, are not set yet, do don't use them here.
- The value of the DMG.CHECKPOINT.START.PACKAGE tag is available for processing user-defined banner or trailer pages. (The current Merge Set tags are available for banner page processing, and the final Merge Set tags are available for trailer page processing.)

  To determine if banner or trailer processing has been specified, the exit program can access the DMG.MERGESET.ID value for banner or trailer processing:

```
%%%BANNER%%%
%%%TRAILER%%%
```

  For details about the DMG.MERGESET.ID tag, see "DMG.MERGESET.ID" on page 316.

DMG.CHECKPOINT.START.SHEETOVL

(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up just as it starts sheet Overlay processing. DMGMERGE places sheet Overlays *last* — on top of assembled pages, fills in any BPSDs in the sheet Overlays, and then also looks up the DMG.CHECKPOINT.START.- and -END.FORM tags for each sheet Overlay. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.START.TUMBLE
(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up just as it starts tumble print processing. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CHECKPOINT.START.WRITEOUT
(PROGRAM CONTROL; USER-SELECTED)

(Optional) a checkpoint tag that DMGMERGE looks up just as it starts writing the print records to the output file. You can populate this tag with a count or switch value, and check the contents of this tag for a count or flag value(s). For an explanation of the checkpoint tags, see "Using the DMG.CHECKPOINT Reserved Tags" on page 286.

DMG.CURRENT.BANNER
(PROGRAM CONTROL; INTERNAL/DMGMERGE)

Contains the current user-defined banner form for the current clean output file as specified in the BANNER= parameter in the DMGMERGE SYSIN (including any parentheses). The maximum value length is 39 characters.

You can use this tag to print the banner form name on a user banner page. If there is no BANNER= parameter, the value will be 39 hex zeroes, which can generate message DMGMRG540W (Tagged data contained one or more Metacodes) if the associated printer is a Metacode printer.

DMG.CURRENT.CHAIN
(INFORMATIONAL; INTERNAL/DMGMERGE)

Contains the data Chain for the current Document Package. The maximum value length is four characters.

DMG.CURRENT.COPY.NUMBER
(INFORMATIONAL; INTERNAL/DMGMERGE)

Contains the current Document Package copy number, left justified, with no leading zeros. This value will always be **1** unless either COPIES= or DMG.GCPY.*groupname* is set to a value larger than one. The maximum value length is eight characters.

DMG.CURRENT.DDNAME
(PROGRAM CONTROL; INTERNAL/DMGMERGE)

Contains the file name of the JCL control statement for the file to which the current Merge Set is being written by DMGMERGE.

The name of this Reserved Tag in Documerge 1.7 was DDNAME.CURRENT.

DMG.CURRENT.FGRPDEF
(INFORMATIONAL; INTERNAL/DMGMERGE)

Contains the font group definition name used for processing the current Document Package, not including the **DF** prefix. The maximum value length is six characters.

DMG.CURRENT.FORM

(PROGRAM CONTROL; USER-SELECTED)

Contains the member name, and other identification and processing information for the form that the DMGPRNT program is currently processing. This tag returns the following 60-character value:

Character Position	Description
1–32	VLAM member name
33	blank
34–38	Revision level
39	blank
40–43	Chain name
44	blank
45–52	VLAM (EDL) DD name
53	blank
54–60	DMG.FLST sequence number of this form in the DMG.FLST.*Groupname* tag.

Suggested uses for the DMG.CURRENT.FORM tag include the following:

- Code in a DMG.C.xxx tag to program a user exit that does conditional processing based on the form name or other identifying information.
- Code in a zero-length DMG.C.xxx tag to capture the name of the form for a report or log file.
- Code in a BPSD to provide form information in banner or trailer pages.

DMG.CURRENT.GROUP

(PROGRAM CONTROL; INTERNAL/DMGMERGE)

Contains the name of the Group currently being processed by DMGMERGE.

The name of this Reserved Tag in Documerge 1.7 was DMG.CURRENT.

DMG.CURRENT.MERGEDEF

(INFORMATIONAL; INTERNAL/DMGMERGE)

Contains the merge definition name used for processing the current Document Package, not including the **DM** prefix. The maximum value length is six characters.

DMG.CURRENT.PRINTDEF

(INFORMATIONAL; INTERNAL/DMGMERGE)

Contains the print definition name used for processing the current Document Package, not including the **DP** prefix. The maximum value length is six characters.

DMG.CURRENT.SHEET.COUNT

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the number of sheets of paper for the current clean DD file. One possible use involves finishing equipment. Using the tag would help ensure that there were no missing pages in a given DD file (stack of paper).

These restrictions apply to the DMG.CURRENT.SHEET.COUNT tag:

- The tag is valid only for error-free Merge Sets. If the Merge Set is in error, the value generated for this tag is meaningless.
- When using DMG.CURRENT.SHEET.COUNT, do not route clean and error Merge Sets to the same DD.

DMG.CURRENT.TRAILER

(PROGRAM CONTROL; INTERNAL/DMGMERGE)

Contains the current user-defined trailer form for the current clean output file as specified in the TRAILER= parameter in the DMGMERGE SYSIN (including any parentheses). The maximum value length is 39 characters.

You can use this tag to print the trailer form name on a user trailer page. If there is no TRAILER= parameter, the value will be 39 hex zeroes, which can generate message DMGMRG540W (Tagged data contained one or more Metacodes) if the associated printer is a Metacode printer.

DMG.DATE

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the date that the DMGMERGE step started. This value remains constant throughout a Documerge run and is generated by DMGMERGE.

DMG.DD.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

Holds the file name, or a list of file names for the JCL DD statement(s) used for clean output for the specified Group. DD JCL statement(s) for the file name(s) must be specified in the DMGMERGE JCL. This tag is used for independent routing, where the output Document Packages for the specified Group are directed to file(s) or SYSOUT classes named by the JCL control statement. This permits the use of multiple output devices on a Group basis.

For each Group, the requirement is to code either the DDNAME control card or the DMG.DD.*Groupname* tag in the VDR. If both the DMG.DD.*Groupname* tag and the DDNAME control card occur, Documerge uses the DDNAME control card value.

The maximum value length for MVS is eight characters. The value(s) for this tag can be generated by

- VDR file name value(s)

  For Documerge 3.x, instead of a single DDname, you can code a list of DDnames in sequence with at least one blank separating each DDname.

  For example, you could code the DMG.DD.*Groupname* tag to contain the following DD names:

```
INSC0050 INSC0100 INSCREST
```

- DMGMERGE control card(s). Code file names with the Merge Group control card DDNAME=. For example:

```
MERGE GROUP=XXXXXXXXX -
DDNAME=XXXXXXXX
```

The name of this Reserved Tag in Documerge 1.7 was DDNAME.*Groupname*.

DMG.END.OF.SET

(PRINT RELATED; INTERNAL/DMGMERGE)

Indicates whether DMGMERGE inserts a dash code for marking the last page of a Document Package. You can use this tag and the DMG.END.OF.SUBSET tag in the same form. Some finishing equipment, such as Bell & Howell AIM equipment, can read the dash code to verify that all forms in the subset have been processed. (See "Reserved Tags and Dash Codes" on page 326 for information about dash codes.)

Dash codes are controlled by two parameters of the MERGEDEF PEDEF:

- DASHCODE-ON=
- DASHCODE-OFF=

DMGMERGE assigns the DASHCODE-ON= value to the last DMG.END.OF.SET tag in the Document Package. DMGMERGE assigns the DASHCODE-OFF= value to all preceding DMG.END.OF.SET tags in the same Document Package. You determine the values of these parameters. You can use whatever characters that your finishing equipment requires.

DMG.END.OF.SUBSET

(PRINT RELATED; INTERNAL/DMGMERGE)

Indicates whether DMGMERGE inserts a dash code for marking the last page of a portion, or subset, of a Document Package.

You can code this tag and the DMG.END.OF.SET tag in the same form.

Some finishing equipment, such as Bell & Howell AIM equipment, can read the dash code to verify that all forms in the subset have been processed. (See "Reserved Tags and Dash Codes" on page 326 for information about dash codes.)

You define a Document Package subset with the ENDSUBset (ESS) Print Option in a Rulebase Library Structure Rule. (See "ESS" on page 174 for information about the ESS Print Option.)

Dash codes are controlled by two parameters of the MERGEDEF PEDEF:

- DASHCODE-ON=
- DASHCODE-OFF=

DMGMERGE assigns the DASHCODE-ON= value to the last DMG.END.OF.SUBSET tag in the subset. DMGMERGE assigns the DASHCODE-OFF= value to all preceding DMG.END.OF.SUBSET tags in the same subset. You determine the values of these parameters. You can use whatever characters that your finishing equipment requires.

DMG.ERDD.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

Holds the file name, or a list of file names for the JCL DD statement(s) used for output in error for the specified Group. DD JCL statement(s) for the file name(s) must be specified in the DMGMERGE JCL. For independent routing, the output Document Packages for the specified Group are directed to file(s) or SYSOUT classes named by JCL control statements, so that there can be multiple output devices for a Group.

For each Group, the requirement is to code either the ERRDDN control statement for the MERGE command or the DMG.ERDD.*Groupname* tag in the VDR. If both the DMG.ERDD.*Groupname* tag and the ERRDDN control card occur, Documerge uses the ERRDDN control card value.

The maximum length for each MVS file name is eight characters. The value(s) for this tag can be

- VDR file name(s). For Documerge 3.1 and later releases, instead of a single DDname, you can code a list of DDnames in sequence with at least one blank separating each DDname. For example, you could code the DMG.DD.*Groupname* tag to contain the following DD names:

```
INSC0050 INSC0100 INSCREST
```

- DMGMERGE control card(s). Code file names with the Merge Group control statement ERRDDN. For example:

```
MERGE GROUP=XXXXXXXX -
ERRDD=XXXXXXXX
```

The name of this Reserved Tag in Documerge 1.7 was ERRDDN.*Groupname*.

DMG.ERROR.VRF

(PROGRAM CONTROL; USER-SELECTED)

Indicates the names of two output files:

- The error VRF for the Merge Set
- The allocation (DMGVRFA) file for the error VRF

See "" on page 252 for more information.

This is an optional tag. It can occur in each Merge Set, and it applies to all Groups in a Merge Set.

This value must conform to the rules for MVS DD names. It can contain 1 to 16 characters:

Characters	Description

1-8	The name of the error VRF output file for the Merge Set.
	For example, ERRVFILE, in positions 19-26 below, is specified in characters 1-8 of the tag value:
	`01...5...10...15...20...25...30...35...40...45...50`
	`CHAR DMG.ERROR.VRF ERRVFILEALLVFILE DMG.IDEF.9700.F`
	`ZONE 441CDC4CDDDD4EDC10CDDECCDCCDDECCDC1CDC4CCCC4FFFF4C`
	`NUMR 001447B59969B59606499569351335693544 47B9456B970OB6`
	This DD name is overridden if you specify an ERRVRF parameter in the DMGMERGE GLOBAL command. Refer to "ERRVRF=" on page 390.
	DMGMERGE determines the name of the error VRF in this order:
	(1) If you specify an ERRVRF parameter, DMGMERGE writes *every* Merge Set that has an error to the file named in the ERRVRF parameter.
	(2) If you do not specify an ERRVRF parameter, DMGMERGE writes a Merge Set that has an error to the error file *only if* that Merge Set contains a valid DMG.ERROR.VRF tag. The error file is the DD named in characters 1-8.
	(3) If you do not specify either an ERRVRF parameter or a DMG.ERROR.VRF tag, DMGMERGE does not write Merge Set errors to any output file.

9-16	The name of the allocation (DMGVRFA) file for the error VRF.
	For example, ALLVFILE, in positions 31-40 below, is specified in characters 9-16 of the tag value:
	`01...5...10...15...20...25...30...35...40...45...50`
	`CHAR DMG.ERROR.VRF ERRVFILEALLVFILE DMG.IDEF.9700.F`
	`ZONE 441CDC4CDDDD4EDC10CDDECCDCCDDECCDC1CDC4CCCC4FFFF4C`
	`NUMR 001447B59969B59606499569351335693544 47B9456B970OB6`
	This DD name is overridden if you specify an ERRVRFA parameter in the DMGMERGE GLOBAL command. Refer to "ERRVRFA=" on page 390.
	DMGMERGE determines the name of the error VRF allocation file in this order:
	1 If you specify an ERRVRFA parameter, DMGMERGE uses the storage defined by the file named in the ERRVRFA parameter.
	2 If you do not specify an ERRVRFA parameter, DMGMERGE uses the storage defined by the file named in characters 9-16 of the DMG.ERROR.VRF tag.
	3 If you specify a DD name in characters 1-8 of the DMG.ERROR.VRF tag, but do not specify a DD name in characters 9-16, DMGMERGE uses the storage defined by the default file name ERRVRFA.
	4 If you do not specify a DD name in characters 1-8 of the DMG.ERROR.VRF tag, DMGMERGE does not write to an error VRF or use an allocation file.
	NOTE: You cannot use the DMGMERGE parameter TAG= to define DMG.ERROR.VRF, because this parameter is Group dependent. The TAG= parameter values are not used when writing the error VRF or in determining its DD name.

DMG.FDEF.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

This tag can contain a 1- to 6-character value that is the name of the FGRPDEF (Font Group Def) that Documerge will use to process the Group specified by *Groupname*.

You can use the **DMG.FDEF.***Groupname* tag instead of coding the FGRPDEF parameter in a MERGE or FILEDEF command. This enables dynamic selection of the FGRPDEF from the VDR. If you don't code **DMG.FDEF.***Groupname,* DMGMERGE uses the default FGRPDEF, if one is coded in the PRINTDEF.

If you use the DMG.FDEF.*Groupname* Reserved Tag, the following rules apply:

- If a FGRPDEF parameter is also specified in a MERGEDEF used in the MERGE or FILEDEF command, Documerge uses the value in the FGRPDEF parameter instead of the DMG.FDEF.*Groupname* value.
- If a MERGEDEF is not specified in either the MERGE or FILEDEF commands, the VRF must contain a DMG.MDEF.*Groupname* tag with a valid MERGEDEF value. Documerge then ignores any individual FGRPDEFs coded in the MERGE or FILEDEF commands for the Group.

DMG.FLST.*Groupname*

(PROGRAM CONTROL; INTERNAL/DMGRFMT)

Contains a complete list of forms that are included in the Document Package produced for the specified Group. This tag must be present in the VRF in order for merging to take place. DMGRFMT generates this tag with:

(1) The names of the Explicit and Inline forms in the VDR

(2) The names of the Implicit forms in the Rulebase Library Forms Table.

DMGRFMT generates this Reserved Tag automatically. Do not write this tag yourself through the Rulebase Library Tag Table or DMGVRFWR.

The maximum length of value is 42 bytes; 1-32 byte VLAM member name or inline form name (blank padded on right as needed), followed by a five byte revision level, followed by five bytes of blanks.

The highest revision level at the time of the DMGMERGE step is used (if no revision level is specified). If the chain is not specified, the default chain contained in MERGEDEF is used, if a CHAIN= value is specified in the MERGE control card this overrides any chain values specified in the MERGDEF.

The name of this Reserved Tag in Documerge 1.7 was GROUPING.*Groupname*.

DMG.FONT.SUBSTITUTION.AFP

(PRINT RELATED; USER-SELECTED)

DMG.FONT.SUBSTITUTION.AFP lets you specify AFP font substitutes for dynamic selection of a desired signature font or logo font. Font substitution applies to all forms used to process a Merge Set.

You can use this reserved tag to

- Eliminate the need to code all possible signature fonts in the font list, even though only one of them is really needed for a given Document Package.

  For example, if your application uses several different signatures for insurance policies, and therefore several different signature or logo fonts, you can program the VDR to specify the required logo or signature font name on a *just-in-time* basis — by using the DMG.FONT.SUBSTITUTION.AFP tag.

- Eliminate the need to program multiple BPSDs. If your application used several signatures that span several fonts, you have had to code several BPSDs that would overprint each other to get the desired signature.

  Use of the DMG.FONT.SUBSTITUTION.AFP tag eliminates the need for multiple overprinted BPSDs. Instead, you code only one BPSD using only one signature font. Then, using this reserved tag, you tell DMGMERGE to change the font name.

In the DMG.FONT.SUBSTITUTION.AFP tag, the format of each original font/font substitute 48-character entry is

Character Position	Description
1–24	The original font:
1–8	The original coded font name, which traditionally starts with **X**
9–16	The original code page name, which traditionally starts with **T**
17–24	The original font character set name, which traditionally starts with **C**
25–48	The new (substitute) font:
25–32	The substitute coded font name, which traditionally starts with **X**
33–40	The substitute code page name, which traditionally starts with **T**
41–48	The substitute font character set name, which traditionally starts with **C**

Following are the rules for using DMG.FONT.SUBSTITUTION.AFP.

- You can specify the font name, or the code page and font character set, but not both. Therefore, for a given original or substitute font, you can specify one of the following:
  - A coded font name and spaces only for the code page and font character set.
  - Both the code page and font character set, and spaces only for the name of the coded font.

  You don't have to specify the same fields for the original and new font (that is, you can specify a font name for the original font, and specify the code page and font character set for the new font, and vice-versa).

CAUTION!

Unless the original and substituted fonts have exactly the same font widths for all characters, font substitution should not be used to change a text font, because this could change margins or cause text to be shifted off the page.

Documerge does not check for the existence or validity of the new font. Therefore, it is possible to select a new font name that does not exist, and this could cause print errors.

Also, the Documerge calculations for vertical and horizontal size (e.g., for concatenation) are based on the original font. If you do not substitute a font similar in size to the original font, printing off the page might occur. Documerge does not warn if printing goes off the page because of font substitution.

- When specifying entries for DMG.FONT.SUBSTITUTION.AFP:
  - Because DMGMERGE does not allow hex FF in SYSIN input, you have to specify all blanks in an 8 character name entry to indicate it is null. DMGMERGE will change the spaces to all hex FF. You can also use all blanks when generating the DMG.FONT.SUBSTITUTION.AFP tag in the VRF to indicate a null name, although in this case, you can also specify hex FF as the first two bytes of the name.
  - If you repeat a font/font substitute specification in one or more DMG.FONT.SUBSTITUTION tags, Documerge ignores all occurrences of the entry but the first.
  - The overall limit for each Merge Set and device type (AFP or Metacode) is 256 entries of font/font substitutes (for each form, there is still a maximum number of 128 fonts).

    For example, you could not have two DMG.FONT.SUBSTITUTION.AFP tags, each with 130 entries, for the same Merge Set. This would make 260 entries total, and exceed the 256 maximum limit. Documerge would ignore the extra fonts and generate an error message.
  - If desired, the tag can be empty.
- You can specify DMG.FONT.SUBSTITUTION tags more than once, and the tags can come from different sources.

  For example, these tags can be coded in the Rulebase and also in the DMGMERGE MERGE TAG parameter. Documerge first searches for occurrences of in the MERGE command TAG parameter, before searching for occurrences of the tag in the VRF.
- You cannot propagate or *nest* font substitutes.

  For example, if a DMG.FONT.SUBSTITUTION tag specifies that Documerge replace original FONT1 with substitute FONT2, Documerge performs the substitution; but if a second entry specifies that FONT2 be replaced with FONT3, Documerge does not perform the substitution.

DMG.FONT.SUBSTITUTION.META

(PRINT RELATED; USER-SELECTED)

DMG.FONT.SUBSTITUTION.META lets you specify METACODE font substitutes for dynamic selection of a desired signature font or logo font. Font substitution applies to all forms used to process a Merge Set.

You can use this reserved tag to

- Eliminate the need to code all possible signature fonts in the font list, even though only one of them is really needed for a given Document Package.

  For example, if your application uses several different signatures for insurance policies, and therefore several different signature or logo fonts, you can program the VDR to specify the required logo or signature font name on a *just-in-time* basis — by using the DMG.FONT.SUBSTITUTION.META tag.

- Eliminate the need to program multiple BPSDs. If your application used several signatures that span several fonts, you have had to code several BPSDs that would overprint each other to get the desired signature.

  Use of the DMG.FONT.SUBSTITUTION.META tag eliminates the need for multiple overprinted BPSDs. Instead, you code only one BPSD using only one signature font. Then, using this reserved tag, you tell DMGMERGE to change the font name.

In the DMG.FONT.SUBSTITUTION.META tag, the format of each original font/font substitute 12-character entry is

Character Position	Description
1–6	The original font name
7–12	The substitute font name

Following are some rules and examples for using DMG.FONT.SUBSTITUTION.META reserved tag:

CAUTION!

Unless the original and substituted fonts have exactly the same font widths for all characters, font substitution should not be used to change a text font, because this could change margins or cause text to be shifted off the page.

Also, the Documerge calculations for vertical and horizontal size (e.g., for concatenation) are based on the original font. If you do not substitute a font similar in size to the original font, printing off the page might occur. **Documerge does not warn if printing goes off the page because of font substitution.**

- Because DMGMERGE performs font substitution in the final stages of processing, it does not change the %%%ISIFONTDEF%%% comment record, so the record still reports the original (unsubstituted) font name.

- When coding entries for DMG.FONT.SUBSTITUTION.META
  - If you repeat a font/font substitute specification in one or more DMG.FONT.SUBSTITUTION tags, Documerge ignores all occurrences of the entry but the first.
  - The overall limit for each Merge Set and device type (AFP or Metacode) is 256 entries of font/font substitutes (for each form, there is still a maximum number of 128 fonts).

    For example, you could **not** have two DMG.FONT.SUBSTITUTION.META tags, each with 130 entries, for the same Merge Set. This would make 260 entries total, and exceed the 256 maximum limit. Documerge would ignore the extra fonts and generates an error message.
  - If you use font substitution for forms that will be tumble printed, you must specify both the original upright font, and the tumbled font in the DMG.FONT.SUBSTITUTION.META tag; otherwise the tumbled font will not get replaced.
- You can specify DMG.FONT.SUBSTITUTION tags more than once, and the tags can come from different sources.

  For example, these tags can be coded in the Rulebase and also in the DMGMERGE MERGE TAG parameter. Documerge first searches for occurrences of in the MERGE command TAG parameter, before searching for occurrences of the tag in the VRF.
- You cannot propagate or *nest* font substitutes.

  For example, if a DMG.FONT.SUBSTITUTION tag specifies that Documerge replace original FONT1 with substitute FONT2, Documerge performs the substitution; but if a second entry specifies that FONT2 be replaced with FONT3, Documerge does not perform the substitution.
- Because Documerge cannot change the font list in an FRM, you cannot use font substitution for a Metacode form that contains a FRM reference (i.e., a FORMS= in the DJDE).
- When feasible, code the CDFROM=FGRPDEF parameter to specify the use of the same CODEDEF for the original font and the substitute font.

  Documerge then uses the code page for the original font, instead of the code page for the substitue font, and the CODEDEF used to convert the BPSD values from EBCDIC to ASCII will also be the CODEDEF for the original font, which Documerge has already used to compose the fixed text of the form.

DMG.GCPY.*Groupname*

(PRINT RELATED; USER-RELATED)

Contains the number of copies produced for the specified Group.

NOTE

The MAXCOPY parameter defined in the PRINTDEF must be equal to or higher than the value of the DMG.GCPY.*Groupname* tag.

The maximum value length is five characters with a maximum value of 32,767. Values for this tag can be generated by:

(1) The VDR numeric value.

(2) DMGMERGE COPIES= control card. The value is coded as a Merge Group control card. Example:

```
MERGE GROUP=XXXXXXXXX -
COPIES=XXXXXXXX
```

The COPIES= control card value overrides any VDR-generated value.

Optional. If not specified, one copy of the Group's output is produced. When output is in error, only one copy is produced, and this value is not used.

DMG.HASHMARK

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains a dash code, or hash mark, generated by DMGMERGE for every occurrence of the tag within a Document Package except the first and last.

This tag works with Bell & Howell AIM equipment. It gives the same results as DMG.BENCHMARK.

DMG.HEX8BCC.COUNT

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the total number of hex 8B carriage controls in the output file. Internally generated by DMGMERGE.

The tag contains the value up to, but not including

- The current package if GLOBAL ALLERROR=N
- The current mergeset if GLOBAL ALLERROR=Y

To right-justifiy and zero lead the data for DMG.LINE.COUNT use the MERGE command RESERVEDRT parameter (for details, see "RESERVEDRT" on page 439.)

NOTE

JES also reports the total number of hex 8B carriage controls, and you can compare the JES value with the DMGMERGE value.

DMG.HIGHEST.ERROR.LEVEL

(PROGRAM CONTROL; USER-SELECTED)

(Optional) reports the highest error message level produced during the processing of a Group (Document Package) in a Merge Set.

Use the DMG.HIGHEST.ERROR.LEVEL tag to

- Generate customized audit statistics when associated with the DMG.CHECKPOINT.END.PACKAGE tag.

- Program user exits with the DMG.CHECKPOINT.START.OUTPUT.FILE tag (for details, see "DMG.CHECKPOINT.START.OUTPUT.FILE" on page 300), and DMG.CHECKPOINT.END.PACKAGE tag (for details, see "DMG.CHECKPOINT.END.PACKAGE" on page 297).

The value generated for DMG.HIGHEST.ERROR.LEVEL can be one of the following:

N	No errors so far
W	W-level errror
C	C-level error
E	E-level error

F- and P-level errors automatically cause the current program to quit, so they don't apply. Document Package processing does not generate any I-level messages.

DMG.IDEF.*Groupname*

(HEXADECIMAL: CANNOT BE PRINTED; INTERNAL/DMGRFMY)

Contains Imposition printing information for the specified Group. The value for this tag generated by DMGRFMT based upon the IMPDEF (imposition definition) assigned to a DTN.

DMGRFMT generates this Reserved Tag automatically. Do not write this tag yourself through the Rulebase Library Tag Table or DMGVRFWR.

This is an optional tag.

The name of this Reserved Tag in Documerge 1.7 was IMPDEF.*Groupname*.

DMG.ITEM.COUNT.VERIFY

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the number of sheets in a specified Document Package. The sheets are counted from 1-15 then the count starts over again so that sheet number 16 is numbered as 1.

DMG.LINE.COUNT

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains audit statistic information of the number of lines written by DMGMERGE processing for each clean output file. The value for this tag is always generated by DMGMERGE. If the tag is present in the VRF the value is ignored and the DMGMERGE value is used.

Use of the GLOBAL command ALLERROR= parameter affects the value of DMG.LINE.COUNT. If the value is

- **YES** (write all Groups to error files if any Groups are erred), the value of DMG.LINE.COUNT is as-of the previous Merge Set because at the time of merging, Documerge did not write any data to output files.
- **NO** (write each Group based on its error status)the value of DMG.LINE.COUNT is as-of the previous Document Package for the previous Group in the Merge Set.

In either case, the current Document Package is not included in DMG.LINE.COUNT. For details about ALLERROR=, see "ALLERROR=" on page 386.

To right-justifiy and zero lead the data for DMG.LINE.COUNT use the MERGE command RESERVEDRT parameter (for details, see "RESERVEDRT" on page 439.)

DMG.LPG.*Groupname*

(HEXADECIMAL: CANNOT BE PRINTED; INTERNAL/DMGRFMT)

Contains the values of one LPGDEF command for each BLP (Begin Logical Page) Option Field in the DMG.OPT.*Groupname* tag.

When DMGMERGE finds a BLP option for a form, it gets the next LPGDEF values from the DMG.LPG.*Groupname* tag. DMGMERGE places the form in the area defined by the LPGDEF command values.

DMGRFMT generates this Reserved Tag automatically. Do not write this tag yourself through the Rulebase Library Tag Table or DMGVRFWR.

See "LPGDEF Command" on page 154, "DMG.OPT.Groupname" on page 317, and "DMG.FLST.Groupname" on page 309 for more information.

DMG.MDEF.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

This tag can contain a 1- to 6-character value that is the name of the MERGEDEF (Merge Definition) that Documerge will use to process the Group specified by *Groupname*.

You can use the **DMG.MDEF.***Groupname* tag instead of coding the MERGEDEF parameter in a MERGE or FILEDEF command. This enables dynamic selection of the MERGEDEF from the VDR.

If you use the DMG.MDEF.*Groupname* Reserved Tag, the following rules apply:

- If a MERGEDEF parameter is also specified in a MERGEDEF used in the MERGE or FILEDEF command, Documerge uses the value in the MERGEDEF parameter instead of the DMG.MDEF.*Groupname* value.
- If a MERGEDEF is not specified in either the MERGE or FILEDEF commands, the VRF must contain a DMG.MDEF.*Groupname* tag with a valid MERGEDEF value for each Group you request with the MERGE command. Documerge then ignores any individual FGRPDEFs or PRINTDEFs coded in the MERGE or FILEDEF commands for the Group.

DMG.MERGESET.ID

(PRINT RELATED; USER-SELECTED)

Contains a user-specified value that identifies a Merge Set. **The value of DMG.MERGESET.ID cannot exceed 35 characters**.

Oracle recommends that you define one DMG.MERGESET.ID Reserved Tag in the Rulebase Library Tag table for each Merge Set, with a value that is unique to the Merge Set.

DMG.MERGESET.ID should be the first tag coded in the Tag Table.

This tag is optional, and has no default value. The value for this tag is used by DMGRFMT and DMGMERGE to identify the Merge Set in error on the Documerge Error sheet. It is also placed on the comment record for each Merge Set when OUTCR=YES is specified in the MERGEDEF.

The name of this Reserved Tag in Documerge 1.7 was DOCUMERGE.ID.TAG.

DMG.MISSING.FORMS

(PROGRAM CONTROL; INTERNAL/DMGRFMT)

Contains EDL member names that were requested by DMGRFMT but were not found in the EDL.

DMGRFMT generates this Reserved Tag automatically. Do not write this tag yourself through the Rulebase Library Tag Table or calling DMGVRFWR from your VDR.

For Documerge 3.1 and later releases, if **all EDL forms** specified for a Group are missing, DMGMERGE will generate an error message and a Missing Forms report for all MERGE Groups because DMGRFMT cannot determine which Group a missing form belongs to.

This tag is optional. Each entry is a maximum of 42 bytes: a 32-character EDL member name, followed by four-character revision levels and five blanks.

In Documerge 1.7, this Reserved Tags name is DOCUMERGE.MISSING.FORMS.

DMG.OPT.*Groupname*

(PROGRAM CONTROL; INTERNAL/DMGRFMT)

Lists the Print Options for each document for the Group. This is an optional tag, with a maximum of 56 bytes per option and a maximum of 1000 options per tag.

DMGRFMT generates this Reserved Tag automatically. Do not write this tag yourself through the Rulebase Library Tag Table or DMGVRFWR.

The name of this Reserved Tag in Documerge 1.7 was OPTIONS.*Groupname*.

When you code a Rulebase Library Structure Rule for a Group, you associate Print Options with each DTN. The Print Options correspond to **Option Fields** in the DMG.OPT.*Groupname* tag. The DMGRFMT subprogram places the Print Option values into these Option Fields.

Depending on the value in the DMG.SRC.*Groupname* reserved tag, the DMGRFMT subprogram can also generate the following Metacode cluster name or bin number Print Option values:

PRV Indicates that no FEED= DJDE is to be generated for the cluster name or bin number of the paper tray to use for the DTN.

SRC Indicates that the DMG.SRC.*Groupname* reserved tag is to provide the cluster name or bin number of the paper tray to use for the DTN.

The DMG.OPT.*Groupname* tag contains 7 Option Fields for each DTN. Specific Print Options are assigned to each Option Field. The Option Fields produced in the DMG.OPT.*Groupname* tag for each document are illustrated in the following table. They are always produced in the sequence under the heading **Option Field #**.

DMG.OPT.*Groupname* Option Field Values

Option Field #	Abbreviation	Meaning
1	**DUP/SIM/TUM/IMP**	Duplex/Simplex/Tumble/Imposition
2	**SEP/CON/COF**	Separate/Concatenate/Concatenate force
3	**MAI/AUX/SRC/PRV**	Main/Auxiliary/Source/Previous
4	**STA/OFF**	Stack/Offset
5	**ODD/EVN/ANY**	Odd/Even/Any
6	**POR/LAN**	Portrait/Landscape
7	**ON/OFF/OVL/CKP**	On/Off/Overlay/Keep with previous
8	**BSS/CSS**	Begin Subset/Continue Subset
9	**BLP/CLP/NLP**	Begin Logical Page/Continue Logical Page/No Logical Page

With one exception, each Option Field consists of 4 bytes:

- A 3-character abbreviation for the option
- A 1-byte blank delimiter

The exception is Option Field 7, which relates only to DTN-scope (not Document Package-scope) Overlays (For an explanation, of Overlay scopes, see "Overlay Concepts — Scopes, Levels, and Sets" on page 465). When its value is ON, this Option Field uses two bytes for the abbreviation and a two-byte blank delimiter. Each document in the Group uses 28 bytes for the complete list of Print Options in the DMG.OPT.*Groupname* tag.

In the VRF, the DMG.OPT.*Groupname* Reserved Tag follows the DMG.FLST.*Groupname* Reserved Tag, which lists the forms for the specified Group. The DMG.OPT.*Groupname* tag lists the Print Option abbreviations in the same sequence in which the related form appears in the DMG.FLST.*Groupname* tag.

DMG.FLST.Groupname Example

```
CHAR                DMG.FLST.9700.INSURED yLM.LIFE.OVER
ZONE 444444444444441CDC4CDEE4FFFF4CDEEDCC0ADD4DCCC4DECD
NUMR 000000000000005447B6323B9700B95249540834B3965B6559
    01...5...10...15...20...25...30...35...40...45...50
CHAR LAY                  00001     LM.LIFE.DEC
ZONE DCE44444444444444444FFFFF44444DD4DCCC4CCC444444444
NUMR 318000000000000000000000010000034B3965B453000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR              00001     LM.LIFE.OVERLAY
ZONE 444444444444FFFFF44444DD4DCCC4DECDDCE4444444444444
NUMR 000000000000000010000034B3965B6559318000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR     00001     LM.LIFE.CONTRACT                 0000
ZONE 4444FFFFF44444DD4DCCC4CDDEDCCE4444444444444444FFFF
NUMR 0000000010000034B3965B365391330000000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
```

For example, the Print Option data for the Group named INSURED, which is to receive the documents LM.LIFE.DEC.OVERLAY.5, LM.LIFE.ENDORSE1.9700.5, and LM.LIFE.ENDORSE.2.9700.5, is produced in the VRF. for the document LM.LIFE.DEC.OVERLAY.5, followed by the Print Options for LM.LIFE.ENDORSE1.9700.5, and LM.LIFE.ENDORSE.2.9700.5.

The data produced for this tag appears in the VRF, in the format illustrated in "DMG.OPT.Groupname Example" on page 318. This example shows only the data produced for one document.

DMG.OPT.Groupname Example

```
CHAR 1      DMG.OPT.9700.INSURED {DUP SEP MAI STA ODD P
ZONE F444441CDC4DDE4FFFF4CDEEDCC0CCED4ECD4DCC4EEC4DCC4D
NUMR 1000004447B673B9700B952495400447025704190231064407
    01...5...10...15...20...25...30...35...40...45...50
CHAR OR OVL US1 US2 US3 US4 US5 DUP SEP MAI STA ODD POR
ZONE DD4DED4EEF4EEF4EEF4EEF4EEF4CED4ECD4DCC4EEC4DCC4DDD
NUMR 690653042104220423042404250447025704190231064407 69
    01...5...10...15...20...25...30...35...40...45...50
CHAR  ON  US1 US2 US3 US4 US5 DUP SEP AUX STA ODD POR O
ZONE 4DD44EEF4EEF4EEF4EEF4EEF4CED4ECD4CEE4EEC4DCC4DDD4D
NUMR 065004210422042304240425044702570147023106440769 06
    01...5...10...15...20...25...30...35...40...45...50
CHAR VL                     DUP SEP AUX STA ODD POR ON
ZONE ED44444444444444444444CED4ECD4CEE4EEC4DCC4DDD4DD4
NUMR 530000000000000000000000447025701470231064407690650
    01...5...10...15...20...25...30...35...40...45...50
```

The Print Options always appear in the same sequence, showing the values that apply to your documents. If Print Options were not issued with the STRUCTURE subcommand, the default values are produced. See "Print Options" on page 168 for the definitions of the Print Options and their abbreviations.

For further information about Overlays and Overlay coding options, see "Documerge Overlay Forms" on page 465.

DMG.PACKAGE.TYPE

(INFORMATIONAL; INTERNAL/DMGMERGE)

Use to determine if a user-defined banner or trailer form is being processed. The onene character values can be any of the following:.

B	processing a user banner form
T	processing a user trailer form
M	processing a standard Merge Set — not a banner or trailer

DMG.PAGE.NUMBER

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the page number of a single-sided image within a specific Document Package. The format is right justified with leading spaces. The value for this tag is generated by DMGMERGE.

The name of this Reserved Tag in Documerge 1.7 was PAGE.NUMBER.

DMG.PDEF.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

This tag can contain a 1- to 6-character value that is the name of the PRINTDEF (Print Definition) that Documerge will use to process the Group specified by *Groupname*.

You can use the **DMG.PDEF.***Groupname* tag instead of coding the PRINTDEF parameter in a MERGEDEF, or MERGE or FILEDEF command. This enables dynamic selection of the PRINTDEF from the VDR. If you don't code **DMG.PDEF.***Groupname,* DMGMERGE uses the default PRINTDEF, if one is coded in the MERGEDEF.

If you use the DMG.PDEF.*Groupname* Reserved Tag, the following rules apply:

- If a PRINTDEF parameter is also specified in a MERGEDEF used in the MERGE or FILEDEF command, Documerge uses the value in the PRINTDEF parameter instead of the DMG.PDEF.*Groupname* value.
- If a MERGEDEF is not specified in either the MERGE or FILEDEF commands, the VRF must contain a DMG.MDEF.*Groupname* tag with a valid MERGEDEF value. Documerge then ignores any individual PRINTDEFs coded in the MERGE or FILEDEF commands for the Group.

DMG.POL.COUNT

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains statistical information about the number of Document Packages produced per output destination (output DD). The value for this tag is always generated by DMGMERGE. If the tag is present in the VRF the value is ignored and the DMGMERGE value is used.

NOTE

You can use the RESERVEDRT parameter of the MERGE command to right justify the tag data with leading zeros and no commas. See "RESERVEDRT" on page 439 for more information.

DMG.PRA

(PRINT RELATED; USER-SELECTED)

Contains the value to use for BTEXT PRA (page reconciliation amount) processing of the Document Packages for all the Groups in a Merge Set. The contents of the DMG.PRA reserved tag must be generated by the VDR and written to the VRF so that DMGMERGE can generate a single PRA= value for each Document Package.

If the PRA values for each Document Package for each Group are the same, use DMG.PRA to avoid having to code a DMG.PRA.*Groupname* for each Group. If DMGMERGE finds both the DMG.PRA.*Groupname* and DMG.PRA reserved tags, it uses the value in DMG.PRA.*Groupname*.

DMG.PRA is optional, and has no default value. The value of the DMG.PRA reserved tag must be numeric with a maximum of 12 characters, and can consist only of 11 digits and one optional decimal point (period character). If the decimal point is used, it must be followed by the last 2 digits of the value. Here are some examples of valid and invalid DMG.PRA values:

Valid values	Invalid values
1	123.4 (if a decimal point is used, it must be followed by 2 digits)
123	123456789012 (too many digits)
123.00	123.456.789 (too many decimal points)
123456789.01	$123.45 (dollar signs, plus signs, commas, etc., are not allowed)

DMG.PRA.*Groupname*

(PRINT RELATED; USER-SELECTED)

Contains the value to use for BTEXT PRA (page reconciliation amount) processing of the Document Package for a single Group. The contents of the DMG.PRA.*Groupname* reserved tag must be generated by the VDR and written to the VRF so that DMGMERGE can generate a single PRA= value for each Document Package.

If the PRA values for each Document Package for each Group are the same, use DMG.PRA to avoid having to code a DMG.PRA.*Groupname* for each Group. If DMGMERGE finds both the DMG.PRA.*Groupname* and DMG.PRA reserved tags, it uses the value in DMG.PRA.*Groupname*.

DMG.PRA.*Groupname* is optional, and has no default value. The value of the DMG.PRA.*Groupname* reserved tag must be numeric with a maximum of 12 characters, and can consist only of 11 digits and one optional decimal point (period character). If the decimal point is used, it must be followed by the last 2 digits of the value. Here are some examples of valid and invalid DMG.PRA.*Groupname* values:

Valid values	Invalid values
1	123.4 (if a decimal point is used, it must be followed by 2 digits)
123	123456789012 (too many digits)
123.00	123.456.789 (too many decimal points)
123456789.01	$123.45 (dollar signs, plus signs, commas, etc., are not allowed)

DMG.PRODUCT

(PRINT RELATED; INTERNAL/DMGMERGE)

This tag describes

- The product name
- The version number

- The release number
- The level number

You can code this Reserved Tag on a Banner or Trailer page to identify the current Documerge system. For example:

```
D O C U M E R G E - V03 R02 L01
```

This tag contains 40 characters, including four leading spaces and five trailing spaces.

DMG.SET.NUMBER

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the logical VRF record number of the Merge Set currently being processed by DMGMERGE. The format is left-justified with trailing spaces. See "DMG.MERGESET.ID" on page 316 for more information.

If needed, you can use the command tag feature (i.e., DMG.C. tags) to reformat this tag. For details, see "DMG.C.xxx Reserved Tag Processing" on page 331.

The name of this Reserved Tag in Documerge 1.7 was DOCUMERGE.SET.NUMBER.

DMG.SHEET.COUNT

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains audit statistic information of the number of sheets of paper generated by DMGMERGE processing for each output file. The value for this tag is always generated by DMGMERGE. If the tag is present in the VRF the value is ignored and the DMGMERGE value is used.

NOTE

You can use the RESERVEDRT parameter of the MERGE command to right justify the tag data with leading zeros and no commas. See "RESERVEDRT" on page 439 for more information.

DMG.SHEET.NUMBER

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the number of the current physical sheets within a specific Document Package. The value for this tag is generated by DMGMERGE.

NOTE

You can use the RESERVEDRT parameter of the MERGE command to right justify the tag data with leading zeros and no commas. See "RESERVEDRT" on page 439 for more information.

DMG.SKEY.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

Optional. Contains variable data that the DMGSORT program uses to sort VRF records for a Group. DMGMERGE performs the sort in ascending order unless you specify **D** in column 39 of the DMGSORT control card.

The maximum length of this Reserved Tag's value is 252 characters. Like all other Documerge tags, DMG.SKEY.*Groupname* can be generated by one of the following:

- A VDR call to the DMGRFMT subprogram

This is the most common method of writing variable data to the VRF. DMGRFMT reads the names of Rulebase Library tags that are specified as sort fields in the Group Table. DMGRFMT passes the sort data to the DMGVRFWR subprogram, which writes the data to the VRF. Usually with this method DMG.SKEY.*Groupname* would not be coded in the Tag Table.

- A VDR call directly to the DMGVRFWR subprogram

  This method bypasses DMGRFMT and the Rulebase Library. You pass the sort data directly to DMGVRFWR, and DMGVRFWR writes the data to the VRF.

- A PC-based VDR, which creates the VRF through MRGUSER.DLL
- A DMG.SKEY.*Groupname* Reserved Tag coded directly in the Rulebase Library Tag Table

IMPORTANT!

DMGSORT ignores the first two characters of the DMG.SKEY.*Groupname* value; it reads the string that begins in the third character position. The maximum length of the actual sort key is 250 characters.

Therefore, a dummy character is required in the first and second position of the DMG.SKEY.*Groupname* value. These two dummy characters are inserted automatically if the DMGRFMT subprogram generates the value. However, the dummy characters are not inserted automatically if DMG.SKEY.*Groupname* is generated in any other way.

If you intend to sort a VRF that contains a DMG.SKEY.*Groupname* value not generated by DMGRFMT you must code the dummy characters in the value before you run DMGSORT. Otherwise, the VRF will not be sorted as you expect.

Oracle suggests that you code a blank in each of the two dummy positions. In the following example the dummy characters are blanks and the sort key begins in position 26:

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR DMG.SKEY.9700.INSURED    Brandon M. Williams 27 Ma
ZONE CDC4EDCE4FFFF4CDEEDCC0344C9898994D44E899889A4FF4D8
NUMR 447B2258B9700B952495405002915465O4B069339142027041
```

If the length of the DMG.SKEY.*Groupname* value is less than 252 characters DMGSORT pads the value internally with binary zeroes. If the length exceeds 252 characters errors occur in DMGSORT.

The name of this Reserved Tag in Documerge 1.7 was SORTKEY.*Groupname*.

DMG.SRC.*Groupname*

(HEXADECIMAL: CANNOT BE PRINTED; INTERNAL/DMGRFMT)

Tag created internally by Documerge, containing information on print options. Under normal operating conditions you should not encounter DMG.SRC.*Groupname*.

DMGRFMT generates this Reserved Tag automatically. Do not write this tag yourself through the Rulebase Library Tag Table or DMGVRFWR.

DMG.SSI.COUNT

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the audit statistic information of the number of single sided images generated by DMGMERGE processing for each output file. The value in this tag is the total of the number of single-sided images (page faces) printed. The value for this tag is always generated by DMGMERGE. If the tag is present in the VRF the value is ignored and the DMGMERGE value is used.

These processing rules apply for the DMG.SSI.COUNT tag:

- Duplexed (but not simplexed) blank pages are counted.
- The tag is valid only for error-free Merge Sets. If the Merge Set is in error, the value generated for this tag is meaningless.
- When using DMG.SSI.COUNT, do not route clean and error Merge Sets to the same DD.

NOTE

You can use the RESERVEDRT parameter of the MERGE command to right justify the tag data with leading zeros and no commas. See "RESERVEDRT" on page 439.

DMG.TIME

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the time the DMGMERGE step started. This value remains constant throughout a Documerge run and is generated by DMGMERGE.

DMG.TLR.*Groupname*

(PROGRAM CONTROL; USER-SELECTED)

Contains the user specified EDL form name and revision level used as a trailer page per Group. This tag generates a form at the end of **the clean output file** selected by DMGMERGE. The revision level is optional, with a default of the highest revision level at the time of the DMGMERGE step.

Values can be generated by

- The VDR EDL member name
- DMGMERGE TRAILER= control card. The EDL member name is coded as a Merge Group control card. Example:

```
MERGE GROUP=XXXXXXXXX -
TRAILER='VLAM2.EDL.MEMBER.NAME'(REV#)
```

The TRAILER= control card value overrides any VDR-generated value. This is an optional Reserved Tag; no trailer is printed if not specified.

DMG.TOTAL.PAGES

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the total number of pages (both sides of page) produced for a specific Document Package. The format is right justified with leading spaces. The value for this tag is generated by DMGMERGE.

The name of this Reserved Tag in Documerge 1.7 was TOTAL.PAGES.

DMG.TOTAL.SHEETS

(PRINT RELATED; INTERNAL/DMGMERGE)

Contains the total number of sheets of paper produced for a specified Document Package. The value for this tag is generated by DMGMERGE.

NOTE

You can use the RESERVEDRT parameter of the MERGE command to right justify the tag data with leading zeros and no commas. See "RESERVEDRT" on page 439 for more information.

DMG.TXT

(PRINT RELATED; USER-SELECTED)

Contains the string of characters to use as the value for the BTEXT TXT parameter for all the Document Packages for all the Groups in a Merge Set. You can use the TXT parameter to provide information thats helpful when reading the reconciliation report generated by BTEXT processing.

The contents of the DMG.TXT reserved tag must be generated by the VDR and written to the VRF so that DMGMERGE can generate a single TXT= value for each Document Package.

If the TXT value for each Document Package for each Group is the same, use DMG.TXT to avoid having to code a DMG.TXT.*Groupname* for each Group. If DMGMERGE finds both the DMG.TXT.*Groupname* and DMG.TXT reserved tags, it uses the value in DMG.TXT.*Groupname*.

The DMG.TXT tag is optional, and has no default value. The value of this tag cannot exceed 65 EBCDIC display characters (that is, normal text), and should not contain any BTEXT parameter keywords (e.g., RNA=). (To avoid prematurely ending the TXT string within the BTEXT DJDE record, Documerge changes any apostrophes or commas into blanks.)

DMG.TXT.*Groupname*

(PRINT RELATED; USER-SELECTED)

Contains the string of characters to use as the value for the BTEXT TXT parameter for a particular Group's Document Package. You can use the TXT parameter to provide information thats helpful when reading the reconciliation report generated by BTEXT processing.

The contents of the DMG.TXT.*Groupname* reserved tag must be generated by the VDR and written to the VRF so that DMGMERGE can generate a single TXT= value for each Document Package.

To vary the TXT value for each Document Package for each Group in a Merge Set, use a DMG.TXT.*Groupname* tag for each Group. Otherwise, use DMG.TXT to avoid having to code a DMG.TXT.*Groupname* for each Group. If DMGMERGE finds both the DMG.TXT.*Groupname* and DMG.TXT reserved tags, it uses the value in DMG.TXT.*Groupname*.

The DMG.TXT.*Groupname* tag is optional, and has no default value. The value of this tag cannot exceed 65 EBCDIC display characters (that is, normal text), and should not contain any BTEXT parameter keywords (e.g., RNA=). (To avoid prematurely ending the TXT string within the BTEXT DJDE record, Documerge changes any apostrophes or commas into blanks.)

DMG.VDR.ERRORS

(PROGRAM CONTROL; USER-SELECTED)

Provides a method for the user to force selected Merge Sets to the error DD at DMGMERGE time, with one or more user defined error messages printed on the error cover sheet for the Merge Set for each Group.

Each Group receives the same user-defined error messages (in addition, of course, to any other DMGMERGE messages for the Merge Set for that Group).

Use DMG.VDR.ERRORS any time you wish to flag a Merge Set for special handling, such as manual insertions, hand delivery, special signature, etc., as well as for any VDR detected errors.

As with most Documerge tags, you may insert DMG.VDR.ERRORS in the Rulebase or you may call DMGVRFWR to write it. If the tag is not present for a Merge Set, no user-defined error messages are generated. In addition, if the first position of the data included in the VDR is a blank (x'40'), no messages are generated (see below for further information about formatting your messages).

If the tag DMG.VDR.ERRORS is not present for a Merge Set, Documerge does not print any VDR-defined error messages.

Documerge writes message DMGMRG315W, reporting that the VDR generated certain messages, followed by message DMGMRG316I for each line of text in the VDR-defined message. Documerge cannot assign a severity level to your message. By default, the severity level is 4, because DMGMRG315 is a "W" level message.

NOTE

A VDR-defined error message causes each Group for the Merge Set to write to its respective error DD; each DD receives the same VDR-defined error message. Also, each DD receives any other DMGMERGE error messages.

Message Format for DMG.VDR.ERRORS

Code the error message(s) in the VDR. The minimum number of message lines is zero; that is, the data may contain no messages at all. The maximum number of message lines varies, depending on the size of each message, but is at least 648 (per Merge Set).

The very first position in the message data field defines the DELIMITER character. Use the Delimiter to separate multiple messages (or different lines of the same message text).

You choose the delimiter character for your application. The delimiter may be any hexadecimal character that does not appear in the message text itself, with the exception of blank (x'40'). Note that the delimiter need not be a printable character. High value (x'FF') is a suggested delimiter value.

Message lines immediately follow the first Delimiter character. Separate each message (or message line) with the Delimiter character. Include text only in the message lines; do not include carriage control characters. You do not need to make message lines all the same length, although if you do make them all the same length, you might find the VDR easier to code.

The maximum length of a message line is 100 characters.

Including consecutive Delimiter characters causes blank messages (blank lines), which you could use to make certain messages more prominent.

If the Delimiter character is blank (x'40'), then Documerge will recognize no VDR defined errors for that Merge Set.. Documerge ignores the rest of the data for the tag, although it is still written to the VRF.

Suggested COBOL Coding for DMG.VDR.ERRORS

Suggestions for COBOL coding of DMG.VDR.ERRORS tag data include:

- Set up a table of fixed-length entries.
- Each entry consists of the one character Delimiter followed by a fixed length message (maximum 100 characters).
- At the beginning of a new Merge Set, move SPACES to this table.

- To generate a message, set the Delimiter to HIGH-VALUE and build the desired error message text.

DMG.VERSION

(PRINT RELATED; INTERNAL/DMGMERGE)

A 6-character value that describes the following information about the Documerge system:

- The 2-digit version number
- The 2-digit release number
- The 2-digit level number

For example

```
030100
```

You can print this information on a Banner or Trailer page by coding the name of this Reserved Tag in a BPSD on that page.

Reserved Tags and Dash Codes

Dash codes are marks, such as the underscore character (_), that are placed on a page in an ordered series. Within a particular series, a combination of dash codes and line spaces provides specific information to an optical scanner. This information could include the total number of physical sheets required to produce a Merge Set, an end-of-set marker, or a staple or fold mark.

Dash codes are used commonly with automated finishing and mailing equipment.

NOTE

Finishing and mailing equipment has special requirements for dash codes. To correlate Documerge Reserved Tag values with this equipment, refer to the vendor's documentation.

Dash codes are produced by certain Documerge Reserved Tags. Some of these Reserved Tags produce dash codes exclusively. These are:

- DMG.BENCHMARK
- DMG.END.OF.SET
- DMG.END.OF.SUBSET
- DMG.HASHMARK

Other Reserved Tags produce numeric values only. (These Reserved Tags are listed on page 327.) You can use any of these numeric Reserved Tags to print dash codes by coding the BPSD in a special syntax. This syntax sets up the BPSD to accept the Reserved Tag's numeric data in binary format.

After you code the BPSD, DMGMERGE:

(1) Converts the Reserved Tag's numeric data into its binary equivalent

(2) Inserts the binary value into the BPSD

(3) Generates dash codes according to the binary value of the BPSD.

IMPORTANT!

Reserved Tags used to print dash codes must not appear on any form requiring Imposition printing or on a Metacode form requiring Tumble printing. If you attempt this, Documerge generates an error message and sends the Merge Set to the error dataset.

Binary Format for Reserved Tags

Certain Documerge Reserved Tags produce numeric values exclusively. Through a special coding syntax, the binary equivalents of these numeric Reserved Tag values can be specified in BPSDs. The binary value of each BPSD determines whether DMGMERGE inserts a dash code in that Boilerplate Space.

You can produce dash codes with the following Reserved Tags in binary format:

- DMG.CURRENT.SHEET.COUNT
- DMG.ITEM.COUNT.VERIFY
- DMG.LINE.COUNT
- DMG.POL.COUNT
- DMG.SET.NUMBER
- DMG.SHEET.COUNT
- DMG.SHEET.NUMBER
- DMG.SSI.COUNT
- DMG.TOTAL.PAGES
- DMG.TOTAL.SHEETS

When you code one of these numeric Reserved Tags in binary format, DMGMERGE converts the decimal value of the Reserved Tag's data into a 15-bit binary number. The bits are numbered from 1 to 15, beginning at the right.

For example, assume that the decimal value of a numeric Reserved Tag is 10. The binary values of the Reserved Tag's bits are:

Bit Identifier	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Binary Value	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

To produce dash codes from a numeric Reserved Tag, you code one BPSD for each bit of the Reserved Tag's binary value (not including leading zeros). In each NAME= parameter, you add the characters **.BIT.** and the bit identifier to the Reserved Tag's name:

```
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.1' LENGTH=5.
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.2' LENGTH=5.
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.3' LENGTH=5.
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.4' LENGTH=5.
```

The LENGTH= parameter specifies the number of times the bit's value is repeated in the BPSD. Therefore, the binary format of the previous example is:

```
00000        (bit 1)
11111        (bit 2)
00000        (bit 3)
11111        (bit 4)
```

The maximum bit identifier is 15. This range allows you to print numeric Reserved Tag values up to 32,767 in dash code format. If you code a higher value, Documerge resets dash code processing to zero.

You use two MERGEDEF parameters to specify the characters that DMGMERGE uses to generate dash codes:

- DASHCODE-OFF=
- DASHCODE-ON=

The character that DMGMERGE inserts in the Boilerplate Space depends on the value of the bit specified in the BPSD. The value of a bit is either 0 (off) or 1 (on). Therefore:

- If the bit value is 0, DMGMERGE inserts the character in the DASHCODE-OFF= parameter
- If the bit value is 1, DMGMERGE inserts the character in the DASHCODE-ON= parameter

The dash code character is inserted the number of times specified in the BPSD's LENGTH= parameter. Therefore, LENGTH= determines the length of the printed dash code.

NOTE

You specify the MERGEDEF in the MERGE control card, the FILEDEF control card, or the DMG.MDEF.*Groupname* tag in the VRF.

If you code the Reserved Tag in binary format but the MERGEDEF does not specify DASHCODE-ON= or DASHCODE-OFF=, DMGMERGE uses the following default values:

- DASHCODE-OFF=E' ' (a blank space in EBCDIC)
- DASHCODE-ON=E'_' (an underscore character in EBCDIC)

BPSD Coding for Binary Format

The following example shows the BPSD coding to produce dash codes with a numeric Reserved Tag. For this example, assume that:

- the Reserved Tag is DMG.TOTAL.SHEETS and that its value is 10
- the dash code characters specified in the MERGEDEF are
  - DASHCODE-OFF=E' '(a blank space in EBCDIC)
  - DASHCODE-ON=E'_'(an underscore character in EBCDIC)

The BPSD coding:

```
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.1' LENGTH=5 CHAR=@.
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.2' LENGTH=5 CHAR=@.
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.3' LENGTH=5 CHAR=@.
:BPSD NAME='DMG.TOTAL.SHEETS.BIT.4' LENGTH=5 CHAR=@.
```

generates the following Boilerplate Space and Replacement Characters:

```
@@@@@
@@@@@
@@@@@
@@@@@
```

The binary format of the tag data is:

```
00000        (bit 1)
11111        (bit 2)
00000        (bit 3)
11111        (bit 4)
```

(The binary value of each bit occupies the entire BPSD.)

Therefore, DMGMERGE generates the following dash code series:

```
     (BPSD binary value is 00000, resulting in five blanks)
     (BPSD binary value is 11111, resulting in five underscores)
     (BPSD binary value is 00000, resulting in five blanks)
     (BPSD binary value is 11111, resulting in five underscores)
```

Documerge Version 1.7 Reserved Tags

Some Documerge 1.7 Reserved Tag names did not have the **DMG.** prefix, which was added to all Reserved Tag names in Documerge 2.0. Other version 1.7 Reserved Tag names did use the DMG. prefix: Some of these kept that prefix but were changed in other ways; the rest were unchanged in version 2.0. The DMG. prefix and the version 2.0 Reserved Tag names apply to Documerge 3.x.

The original version 1.7 Reserved Tag names are also valid in Documerge 3.x. The DMGMERGE GLOBAL control card COMPATV1= indicates whether DMGMERGE uses these tag names as Documerge 3.x Reserved Tags or as ordinary tags. Therefore, you can use the original version 1.7 Reserved Tag names in existing VRFs and BPSDs without recoding. Refer to "COMPATV1=" on page 388 for more information.

Renamed Version 1.7 Reserved Tags

The renamed version 1.7 Reserved Tags and their Documerge 3.x equivalents are:

Documerge 1.7	Documerge 3.x	See page
DDNAME.CURRENT	DMG.CURRENT.DDNAME	303
DDNAME.*Groupname*	DMG.DD.*Groupname*	305
DMG.CURRENT	DMG.CURRENT.GROUP	304
DOCUMERGE.ID.TAG	DMG.MERGESET.ID	316
DOCUMERGE.MISSING.FORMS	DMG.MISSING.FORMS	316
DOCUMERGE.SET.NUMBER	DMG.SET.NUMBER	321
ERRDDN.*Groupname*	DMG.ERDD.*Groupname*	306
GROUPING.*Groupname*	DMG.FLST.*Groupname*	309
IMPDEF.*Groupname*	DMG.IDEF.*Groupname*	315
OPTIONS.*Groupname*	DMG.OPT.*Groupname*	317
PAGE.NUMBER	DMG.PAGE.NUMBER	319
SORTKEY.*Groupname*	DMG.SKEY.*Groupname*	321
TOTAL.PAGES	DMG.TOTAL.PAGES	323

DMGMERGE Processing Priorities

If you code one of the version 1.7 Reserved Tags and its Documerge 3.x equivalent in the same Merge Set, DMGMERGE uses only one of the two. The following table lists the conditions that determine which Reserved Tag DMGMERGE uses.

If	DMGMERGE uses
Both Reserved Tags exist in the VRF.	The Documerge 3.x Reserved Tag
▪ The version 1.7 Reserved Tag exists in the VRF. ▪ Its Documerge 3.x equivalent does not exist in the VRF. ▪ You code COMPATV1=YES (the default).	The version 1.7 Reserved Tag
Neither Reserved Tag exists in the VRF.	A default value, which depends on the purpose of the Reserved Tag This default value could cause an error that stops processing.

Unchanged Version 1.7 Reserved Tags

The following Reserved Tag names are identical in versions 1.7 and 3.x:

- "DMG.BNR.Groupname" on page 296
- "DMG.DATE" on page 305
- "DMG.END.OF.SET" on page 306
- "DMG.GCPY.Groupname" on page 314
- "DMG.ITEM.COUNT.VERIFY" on page 315
- "DMG.LINE.COUNT" on page 315
- "DMG.POL.COUNT" on page 319
- "DMG.SHEET.COUNT" on page 321
- "DMG.SSI.COUNT" on page 323
- "DMG.TIME" on page 323
- "DMG.TLR.Groupname" on page 323
- "DMG.TOTAL.SHEETS" on page 323

DMG.C.*xxx* Reserved Tag Processing

This section describes how to set up optional user-exit programs to work with

- **DMG.C.***xxx* Reserved Tags

  The DMG.C.*xxx* tag is a user-defined Reserved Tag. You define it as you do any other user-defined Reserved Tag. However, the DMG.C.*xxx* value is unique, and can be manipulated by an optional user-exit subprogram.

- **CTAGTRIGGER-**Defined Command Reserved Tags

  Documerge 3.1 and later releases let you modify other reserved tags with a 3-character prefix and use these tags for command tag processing. For an explanation of **CTAGTRIGGER-**Defined Command Reserved Tags, see "GLOBAL Parameters" on page 386.

The DMG.C.*xxx* Reserved Tag expands the relationship between Boilerplate Spaces and their corresponding VRF tags by:

- Combining multiple VRF tag values into one Boilerplate Space.
- Controlling the combined output from multiple VRF tags according to your printing requirements.
- Working with an **optional user-exit subprogram** to customize the Boilerplate Space replacement value further, such as by specifying a particular print format or converting the replacement value to a particular data format.

Oracle can supply this optional user-exit subprogram, or your company can write its own customized version. You can have more than one user-exit subprogram, even for the same DMG.C.*xxx* tag. For an example of a COBOL user-exit and special instructions to initialize MVS COBOL II environments, see "Optional Command Tag User-Exit Subprogram" on page 347.

A Boilerplate Space Definition (BPSD) marks a location in an electronic form where variable data is to be printed. Until the time for merging variable data with the form (the DMGMERGE step), Documerge inserts Replacement Characters in the location, or Boilerplate Space, marked by the BPSD. DMGMERGE replaces the Replacement Characters with variable data.

In previous levels of Documerge, the variable-data characters placed into one Boilerplate Space correspond exactly to the VRF value of one Documerge tag. In other words, one Boilerplate Space replacement value could come from only one tag.

In addition to the way you defined Boilerplate Spaces previously, the DMG.C.*xxx* Reserved Tag and the optional user-exit subprogram define extended Boilerplate Space replacement values. These values are created and manipulated dynamically.

Overview of DMG.C.*xxx*

DMG.C.*xxx Value*

The DMG.C.*xxx* VRF value tells DMGMERGE how to build a character string that replaces a Boilerplate Space. DMGMERGE builds this character string dynamically.

The VRF value of the DMG.C.*xxx* tag is the combined values of three DMG.C.*xxx* **commands**. You code these commands within a DMG.C.*xxx* tag in the VDR or a Rulebase Library Tag Table. DMGMERGE places the value of each command into an internal **command output area**. After DMGMERGE processes all commands in the tag, the command output area contains the Boilerplate Space replacement value.

Three DMG.C.*xxx* commands define the DMG.C.*xxx* tag value:

- **TAG** command

  Appends the value of another tag to the current character string in the command output area.

- **CALL** command

  Invokes an optional user-exit subprogram to manipulate the current character string in the command output area. This character string is the value produced by all DMG.C.*xxx* commands that precede a particular CALL command.

- **LITERAL** command

  Appends a constant value that you specify to the current character string in the command output area.

A Boilerplate Space that references a DMG.C.*xxx* tag receives the combined values of all DMG.C.*xxx* commands in the tag. DMGMERGE processes DMG.C.*xxx* commands in their consecutive sequence within the tag. For example, if you code two CALL commands within one DMG.C.*xxx* tag, the output from the first CALL command becomes the input to the second.

You can code as many of the three commands as you want in one DMG.C.*xxx* tag. However, like all tags that are written to the VRF, the DMG.C.*xxx* tag cannot contain more than 65,535 bytes.

You can mix the commands within one tag. You can create a DMG.C.*xxx* tag that uses only one command. You can create a tag that uses many commands of the same type. Or, you can omit commands from the DMG.C.*xxx* tag entirely; Documerge prints blanks in a Boilerplate Space that calls for a DMG.C.*xxx* tag containing no commands.

Documerge places no limit on the number of DMG.C.*xxx* tags in the VRF. All DMG.C.*xxx* tags must begin with the characters **DMG.C.**. (The character C stands for Command.) Each DMG.C.*xxx* tag can have a unique name. For *xxx*, you can substitute any 1- to 24-character name.

Command Output Area

The command output area receives input from all TAG, CALL, and LITERAL commands within a DMG.C.*xxx* tag. DMGMERGE processes these commands in the order you code them within the tag.

Before DMGMERGE processes any commands for a DMG.C.*xxx* tag, the initial value of the command output area is null. After DMGMERGE processes all commands in a DMG.C.*xxx* tag, the command output area contains a character string that is the combined value of all commands in that tag. DMGMERGE inserts this string into the Boilerplate Space marked by a BPSD that references the tag.

The output of the first command processed is appended to the initial null value. At this point, the command output area's current value is the same character string as the output of the first DMG.C.*xxx* command processed.

If the next DMG.C.*xxx* command is a TAG or LITERAL command, its output is appended to the current value of the command output area, beginning with the next available output position. Therefore, the value of the command and the current value of the command output area are combined, and this combined string becomes the new current value. If the next DMG.C.*xxx* command is a CALL command, its output replaces the entire current value of the command output area and becomes the new current value.

The output of each successive command within this DMG.C.*xxx* tag is appended to or replaces the current value of the command output area. After all commands in this DMG.C.*xxx* tag are processed, the command output area's current value is a character string that combines the output of all commands in this DMG.C.*xxx* tag. This final value in the command output area becomes the Boilerplate Space replacement value.

NOTE

The command output area can contain a maximum of 32,759 characters for one DMG.C.*xxx* tag. DMGMERGE generates this string in EBCDIC characters. DMGMERGE translates this string to ASCII characters for Metacode printers as it does for any BPSD.

BPSD Coding

The DMG.C.*xxx* tag requires no special BPSD coding, except that the tag name must begin with the characters **DMG.C.**. All BPSD options are available. You can use a BPSD that references a DMG.C.*xxx* tag on any Documerge form.

Like any other tag referenced by a BPSD, the DMG.C.*xxx* tag must exist in the VRF or be defined in the MERGE control card TAG=. And, the tag name specified in a BPSD must match the tag name in the VRF or the TAG= control card.

Unlike other tags, however, the BPSD length is not related directly to the LEN= value in a Tag Table. This is because the Boilerplate Space receives the combined value of all commands within a DMG.C.*xxx* tag. That value is the character string that remains in the command output area after the last command in the tag is processed.

The length of this character string depends on the function and value of each DMG.C.*xxx* command. Also, any called user-exit subprogram can replace the command output area with a shorter or longer character string. Therefore, the BPSD length depends on the DMG.C.*xxx* commands and user-exit subprograms. The BPSD length should be at least that of the final string in the command output area for the DMG.C.*xxx* tag.

Printer Types

The DMG.C.*xxx* tag works with AFP, Metacode, and line printers.

VDR Coding Example

For an illustration of coding the VDR and BPSD in a DMG.C.*xxx* tag, look at this example. Let's assume that:

- The tag name is DMG.C.EXAMPLE.
- This tag is a combination of:

  POLICY.NUMBER
  (length 10 digits; VRF length is also 10)

  DMG.CURRENT.SHEET.COUNT
  (length 5 digits, right justified, leading zeros).
- We want to maintain statistics on sheet usage, using subprogram SUB1. (Refer to "Optional Command Tag User-Exit Subprogram" on page 347 for more information.)
- The subprogram SUB1 returns the 5-character sheet count and drops the policy number.

Given these assumptions, the VRF value of the DMG.C.EXAMPLE tag is:

- a TAG command for POLICY.NUMBER, 10 output characters
- another TAG command for DMG.CURRENT.SHEET.COUNT, 5 output characters, right justified with leading zeros
- a CALL command for the subprogram SUB1.

You could code the BPSD command like this:

```
.bf font128
:bpsd name='dmg.c.example' length=5.
.pf
```

NOTE

The BPSD length is 5 because the resulting command output area after the final command (the CALL to SUB1) is 5 characters (the value of DMG.CURRENT.SHEET.COUNT).

Therefore, the BPSD length for a DMG.C.*xxx* tag is not necessarily the same as the LEN= value in a Rulebase Library Tag Table.

The TAG Command

The TAG command tells Documerge to append a specific value within a DMG.C.*xxx* tag to the current value of the command output area. The appended value is the value of a Documerge tag that is named in a TAG command. This appended value begins in the next available output position after the current value. Therefore, the value of the tag named in this TAG command and the current value of the command output area are combined, and this combined string becomes the new current value. The value of the tag named in each successive TAG command within this DMG.C.*xxx* tag is appended to the current value of the command output area.

The command output area also receives input from any CALL or LITERAL commands within this DMG.C.*xxx* tag. After all commands in this DMG.C.*xxx* tag are processed, the command output area's current value is a large string that combines the output of all commands in this DMG.C.*xxx* tag. This final value in the command output area becomes the BPSD replacement value.

The TAG command's input and output values are in EBCDIC. For Metacode printers, DMGMERGE translates the input and output to ASCII after it processes all commands in the DMG.C.*xxx* tag.

TAG Command Processing

The TAG command starts with its prefix character, T (position 1), which tells Documerge to begin TAG command processing. After the prefix comes the value of the TAG command; this value is the name of a Documerge tag that you select (positions 2-31). Documerge looks up this tag name's value, and it needs to know what to do if it does not find the value or if the value length is zero. You use the mandatory tag switch (position 37) to tell Documerge how to handle these conditions. Depending on the tag switch you select, Documerge stops processing the DMG.C.*xxx* tag and generates an error message or continues processing.

If processing continues, Documerge gets the value of the tag name. Then Documerge truncates that value according to the input truncation option you select (position 38). You can use this option to drop leading and trailing spaces and leading zeros from a TAG command's input string (the value of the tag name), or you can tell Documerge not to truncate. This optionally truncated input string becomes this TAG command's input value.

To append this value to the command output area, Documerge needs to know how many output positions you want the value to occupy. You specify this number in positions 32-36. You can use the number that is equal to the number of characters in the TAG command's input value after optional truncation. Or, depending on your needs, you can use a different number. In other words, the number of output positions you specify can be equal to, less than, or greater than a TAG command's input value after optional truncation.

If you specify **TAGLN** in positions 32-36, Documerge assigns the number of output positions that is equal to the number of input characters after optional truncation. This input string is appended to the current value of the command output area, beginning with the next available position, and occupies the assigned number of output positions. Processing of this TAG command is complete.

If you specify **BPSDL** in positions 32-36, Documerge assigns the number of output positions that is equal to the length of the corresponding BPSD. If you specify an exact 5-digit number in positions 32-36, Documerge assigns that number of output positions. In each case the number of output positions can be equal to the number of input characters after optional truncation. If so, the input string is appended to the current value of the command output area, beginning with the next available position, and occupies the assigned number of output positions; processing of this TAG command is complete. However, in each case the number of characters in the truncated input string can also be less than or greater than the number of output positions you specify.

If the input value after optional truncation is less than or greater than the number of output positions, you must indicate how you want Documerge to build the output string. This output string consists of the input characters and their format within the output string. For each condition, you have several options for defining a TAG command's output string.

For an input value that is less than the number of output positions, you select an output justification option (position 39). This option determines how the input characters are justified within the output string.

Documerge justifies the input characters by inserting leading spaces, trailing spaces, or leading zeros in the output string. These justification characters are inserted on the left, the right, or both sides of the input characters; they are inserted until the number of output positions (positions 32-36) is reached. This justified string is appended to the current value of the command output area, beginning with the first available output position, and occupies the assigned number of output positions. Processing of this TAG command is complete.

For an input value that is greater than the number of output positions, you select an output truncation option (position 40). This option determines which input characters are excluded from the output string; it also determines whether Documerge stops processing for this DMG.C.*xxx* tag and generates an error message.

Depending on the truncation option value in position 40, iInput characters are dropped from the left, the right, or both sides of the input string; they are dropped until the number of output positions (positions 32-36) is reached. This truncated string is appended to the current value of the command output area, beginning with the first available output position, and occupies the assigned number of output positions. Processing of this TAG command is complete.

Coding the TAG Command

The TAG command is coded within a DMG.C.*xxx* tag. To code a DMG.C.*xxx* tag with a TAG command, you specify the TAG command prefix, the name of the tag whose value you want appended to the command output area, and the number of positions the tag value occupies in the command output area after optional truncation. Also, you assign a mandatory tag switch, an input truncation option, an output justification option, and an output truncation option.

The TAG command is 42 characters long. The following table lists the position, length, and description of the value you specify for each field within a DMG.C.*xxx* TAG command:

Position	Length	Description
01	1	This field is required. Type **T**, the TAG command prefix.
02-31	30	This field is required. Type the name of the tag that you want to add to the end of the command output area. With Documerge 3.1 and later releases, you can use a DMG.*reserved tag name* or any other tag name that exists in the VRF.
32-36	5	This field is required. The number of positions this TAG command input value occupies in the command output area after optional truncation. Documerge gives you three ways to specify this number: ■ Type an exact, 5-digit number. Use leading zeros if necessary. ■ Type **TAGLN** to indicate that the number of output positions is equal to the number of characters in the tag's input string after optional truncation (position 38). TAGLN eliminates blanks and zeros between tags. TAGLN also ensures that the tag's output is not truncated. ■ Type **BPSDL** to indicate that the number of output positions is equal to the length of the BPSD. Normally, you use BPSDL only if a DMG.C.*xxx* tag's input string contains a single TAG command but includes optional truncation (position 38) or output format options (positions 39 and 40).

37	1	This field is required. Type one of the following mandatory tag switches: **Switch** — **Meaning** **M** — **M**andatory. DMGMERGE must find the tag value in the VRF before processing. If DMGMERGE does not find the value, it generates an error message and stops processing this DMG.C.*xxx* tag. **N** — **N**on-blank. DMGMERGE must find the tag value in the VRF before processing, and the value length cannot be zero. (That is, the value cannot be null.) If DMGMERGE does not find the value, it generates an error message and stops processing this DMG.C.*xxx* tag. If DMGMERGE finds the value but the value length is zero, it generates an error message and stops processing this DMG.C.*xxx* tag. **O** — **O**ptional. DMGMERGE does not have to find the tag value in the VRF. If DMGMERGE does not find the tag value, the value is null. No error message is generated, and processing of this DMG.C.*xxx* tag continues.
38	1	This field is required. The input truncation option. Type one of the following: **Option** — **Truncation** **A** — Truncate leading spaces, leading zeros, and trailing spaces **B** — Truncate leading and trailing spaces **L** — Truncate leading spaces **N** — Do not truncate **T** — Truncate trailing spaces **Z** — Truncate leading spaces and leading zeros
39	1	This field is required. The output justification option. This option applies if the value of the tag named in this TAG command after optional truncation is *less than* the number of output positions (positions 32-36). Type one of the following options: **Option** — **Output Format** **C** — Centered. Contains leading and trailing spaces. **L** — Left justified. Contains trailing spaces. **R** — Right justified. Contains leading spaces. **Z** — Right justified. Contains leading zeros. **NOTE:** The relationship between TAG command input values and number of output positions is based on character count, not pel size. If the corresponding BPSD uses a proportional font, you may not get the results you want.

40	1	This field is required. The output truncation option. This option applies if the value of the tag named in this TAG command after optional truncation is *greater than* the number of output positions (positions 32-36). Type one of the following options: **Option** — **Output Format** **C** — Centered. The tag value is truncated equally on the left and on the right. **E** — Error message 371 is written to the ERRDD file. Processing of this DMG.C.*xxx* tag stops. No output string is built. The BPSD is replaced with blanks. **L** — Left justified. The tag value is truncated on the right. **R** — Right justified. The tag value is truncated on the left. **NOTE:** The relationship between TAG command input values and number of output positions is based on character count, not pel size. If the corresponding BPSD uses a proportional font, you may not get the results you want.
41	1	(Optional) deletes or restores the tag value retrieved by the TAG command. Enter one of the following deletion codes: **Option** — **Action** **N** or blank — (Default) do not delete the tag value retrieved by the TAG command **Y** — Delete the tag value retrieved by the TAG command after use **1** — Restore the tag value to the first occurrence retrieved, and do not delete the value **R** — Restore the tag value to the first occurrence retrieved, and delete the value after use
42	1	Blank (**x'40'**). Reserved for future use.

TAG Command Coding Example

For example, to add a mandatory tag named TAG1 with the following attributes:

- 20 mandatory output positions
- no optional truncation
- left-justified output justification with trailing spaces if less than output positions
- left-justified output truncation on the right if more than output positions
- delete tag value after use

you would code:

```
|...+....1....+....2....+....3....+....4..
TTAG1                          00020MNLLY
```

All that the forms coder needs to do is specify the TAG1 BPSD tag, with any BPSD options.

The CALL Command

The CALL command tells Documerge to call a user-specified subprogram dynamically. Oracle can supply this subprogram, or your company can write its own customized version. This subprogram manipulates the current character string in the command output area. You can have more than one user-exit subprogram, even for the same DMG.C.*xxx* tag.

The input to the subprogram is a control block and the current value of the command output area. (Refer to "Optional Command Tag User-Exit Subprogram" on page 347 for more information.) The output from the subprogram is a character string. This string replaces the current value of the command output area for this DMG.C.*xxx* tag.

The command output area also receives input from any TAG or LITERAL commands within this DMG.C.*xxx* tag. After all commands in this DMG.C.*xxx* tag are processed, the command output area's current value is a character string that combines the values of all commands in this DMG.C.*xxx* tag. This final value in the command output area becomes the BPSD replacement value.

If a CALL command is the last command in this DMG.C.*xxx* tag, its output string becomes the BPSD replacement value. The replacement value is always the character string that remains in the command output area after the last command in the tag is processed.

The CALL command's input and output values are in EBCDIC. For Metacode printers, Documerge translates the input and output values to ASCII after it processes all statements in the DMG.C.*xxx* tag.

Coding the CALL Command

The CALL command is 20 characters long. The following table lists the position, length, and description of the value you assign for each field within a DMG.C.*xxx* CALL command:

Position	Length	Description
01	1	**C**, the CALL command prefix.
02-09	8	The name of the CALL subprogram MVS load module or VSE phase.
10-17	8	The value that CALL passes to the CALL subprogram. **NOTE:** The subprogram chooses whether and how to use this value.
18-20	3	Blank (**x'40'**). Reserved for future use.

CALL Command Coding Example

For example, to call a CALL subprogram named ABCDE and pass it the value 123, you code:

```
|...+....1....+....2
CABCDE   123
```

The LITERAL Command

The LITERAL command tells Documerge to append a constant character string that you define to the command output area. You can use this command instead of the TAG command to save Documerge from having to look up tags. Also, this command can keep you from generating an unnecessary tag.

The exact characters in the LITERAL string are appended to the current value of the command output area, beginning with the next available output position, and occupy the assigned number of output positions.

The command output area also receives input from any TAG or CALL commands within this DMG.C.*xxx* tag. After all commands in this DMG.C.*xxx* tag are processed, the command output area's current value is a character string that combines the values of all commands in this DMG.C.*xxx* tag. This final value in the command output area becomes the BPSD replacement value.

The LITERAL command's input and output values are in EBCDIC. For Metacode printers, DMGMERGE translates the input and output string to ASCII after it processes all commands in the DMG.C.*xxx* tag.

Coding the LITERAL Command

The length of the LITERAL character string varies, depending on the number of characters you specify. The maximum number of characters for one LITERAL string is 999. If you need a LITERAL string longer than 999, you can code consecutive LITERAL commands with lengths of 999 or less within one DMG.C.*xxx* tag.

The following table lists the position, length, and description of the value you assign for each field within a DMG.C.*xxx* LITERAL command:

Position	Length	Description
01	1	Type **L**, the LITERAL command prefix.
02-04	3	Type the length of the LITERAL string, using three digits and leading zeros.
05-end	(varies)	Type the LITERAL character string. **NOTE:** If you want a DMG.C.*xxx* tag that consists only of LITERAL strings, you don't have to code it; you can get the same results by using regular Documerge processing. The LITERAL strings can be the value of a regular (non-DMG.C.*xxx*) tag in the VRF, and the BPSD can reference this regular tag.

LITERAL Command Coding Example

To code a LITERAL command with the value CONSTANTLITERALSTRING, you code:

```
|...+....1....+....2....+....3....+....4....+....5...
L021CONSTANTLITERALSTRING
```

DMGTAGL

DMGTAGL is a Documerge tag lookup and modification program that you can call from a DMG.C.xxx user exit or other exit program.

You can use DMGTAGL to

- Look up more than one tag value.
- Change the values of certain reserved tags for succeeding Groups in the same Merge Set.
- Change the values of the DMG.CURRENT.DDNAME and DMG.CURRENT.ERRDDN reserved tags for the clean and error file names.

DMGTAGL can perform the following operations:

- Read the VRF sequentially
- Look up the value of a specific tag name
- Change the value of a tag.
- Free the storage for a tag

If you change a tag value in the VRF, it remains changed for subsequent Groups in the same Merge Set.

DMGTAGL Tag Lookup Control Block (TLCB)

Program DMGTAGL expects one parameter, a "tag lookup control block", or TLCB for short. The DMG.C.xxx user exit passes the following control block of fields to DMGTAGL. Here is the layout of the TLCB, with recommended item names in the exit program:

Field Name	Position	(Length)	Description
TLCBID	001-004	(4)	"TLCB" Control block ID -- must be "TLCB".
TLCBSIZE	005-008	(4)	Binary fullword size of this control block; must be decimal 84 (hex 54).
TLCBVRM	009-014	(6)	Version/release/mod of this control block. Value is ignored, but we recommend you code "030100".
TLCBFILL	015-016	(2)	Filler for alignment; set to hex zeros.
TLCCOMND	017	(1)	Command:
			G Get the value of a tag
			V Change the value of a tag. There are other commands used internally by Documerge not documented here.
TLCTGNAM	018-047	(30)	Tag name.
	048	(1)	Filler for alignment; set to hex zero.
TLCOCNUM	049-052	(4)	Tag occurence number, when same tag name occurs more than once in the Merge Set:
			Zero (0) First non-deleted occurrence **Greater than zero** Specific occurence number (even if deleted) **Less than zero** Invalid
TLCBPSDL	053-056	(4)	Binary fullword length of current BPSD, used by some functions of DMGTAGL. However, user exits can set this to all hex zeros.

Field Name	Position	(Length)	Description
TLCDATAD	057-060	(4)	Binary fullword address of the data value for this tag. Command:
			G value is set by DMGTAGL; points to the current value. **V** value is set by the user exit program; points to the new value.
TLCDATLN	061-064	(4)	Binary fullword length of the data value for this tag. Command:
			G value set by DMGTAGL; contains the length of current value. **V** value set by the user exit program; contains the length of current value.
TLCTAGDA	065-068	(4)	Binary fullword address of Tag table entry for this tag. As a rule, always set this field to binary zeros before any call to DMGTAGL. DMGTAGL will set this field following most calls, so **you must code an instruction to reset it to binary zeros before each call to DMGTAGL**. Failure to do this can result in a return code 12 or other undesirable results.
TLCVRFTN	069-072	(4)	Binary fullword tag sequence number. Set by DMGTAGL for "G" command (get a value) to the occurence number of this tag in the VRF, or zero if tag is not in the VRF (such as an internal tag or from a TAG= parameter in the DMGMERGE SYSIN). The value is ignored for other commands.
TLCFLAGO	073	(1)	Binary bit flag (one byte/8 bits) set by DMGTAGL. Values are:
			10000000 Frees this tag when done. This indicates that not enough TAGBUFF and/or FORMSBUFF was allocated, and that some or all of this tag is in temporary storage. Generally user exit programs can ignore this. However, for best effeciency, we always recommend using the DMGVRFA file to insure sufficient TAGBUFF and DATABUFF; then this bit would never be set on. **01000000** Tag is currently deleted. This can be set when requesting a specific occurence in the TLCOCNUM field, and the tag is currently marked as "deleted". Both bits can be on at the same time; for example '11000000' means "should free" and "deleted". Other bits are currently not used.
TLCFLAGI	074	(1)	Binary bit flag (one byte/8 bits) set by user exit before calling DMGTAGL. This sets various options. To set more than one option, add the values. For example, "11000000" says "delete this tag after lookup", and also "process only VRF tags". Some options are mutually exclusive. It would not be logical to set "00001000" (force command processing) and also "00000100" (do not do command processing). Values are:

Field Name	Position	(Length)	Description
			10000000 Delete this tag after lookup; used by "G" command only. **01000000** Process only VRF tags; used by both "G" command and "V" command. If this bit is set on, internal tags and tags from TAG= parameters in the DMGMERGE SYSIN are bypassed. **00100000** Not used by "G" and "V" commands. **00010000** Undelete this tag. If it is currently deleted, mark it as not deleted. This undeletes **all** occurences of this tag, not just the current occurrence. Ok to use even if no occurences are currently deleted. **00001000** Force command processing. This forces command processing even if the tag name does not begin with "DMG.C." and the tag value does not begin with the GLOBAL CTAGTRIGGER value. User exits should not set this bit on; otherwise DMGTAGL will likely end with a bad return code.
			00000100 Do not do command processing. Command processing is bypassed even if the tag name does not begin with "DMG.C." and the tag value does not begin with the GLOBAL CTAGTRIGGER value. Use this to get the value of a command tag "exactly as coded" (aka "as-is"). **00000010** This is a BPSD tag. DMGTAGL uses this to update the STATSFILE properly. User exits should not set this bit on, since by definition the lookup is not done directly by a BPSD (even if a BPSD trigged the user exit program in the first place). **00000001** Not used yet; set this to zero.
	075-076	(2)	Filler for alignment; set to hex zeros.
TLCRETC	077-080	(4)	Binary fullword return code set by DMGTAGL. Also passed in register 15 (standard return code setting). See TLCRETC return/reason codes for values.
TLCRETC	081-084	(4)	Binary fullword reason code set by DMGTAGL. See TLCRETC return/reason codes for values.

To Get the Value of a Tag

1. Set **TLCCCOMND=** to G.
2. Set **TLCOCNUM=** to zero to get next undeleted occurrence, or set to greater than zero to get a specific occurrence. Zero is normal.
3. Set **TLCTAGDA =** to zero.
4. Set **TLCFLAGI =** to the desired options; zero is normal.
5. Call **DMGTAGL** passing the TLCB as the only parameter.
6. Test the return code. If not zero, handle appropriately.

If the return code is zero, then the exit program has the following value set by DMGTAGL:

Field Name	Set Value
TLCDATAD =	address of the data (the value for this tag)
TLCDATLN =	length of the data
TLCTAGDA =	internal table entry for this tag; of no use to user exit program, but be aware the value has changed.
TLCVRFTN =	tag sequence number in VRF in this Merge Set, or zero if tag is not in VRF
TLCFLAGO =	bits set as needed.
TLCRETC =	return code
TLCREAC =	reason code

To Change the Value of a Tag

1. Set **TLCCOMND =** to V
2. Set **TLCTGGNAM =** to the desired tag name
3. Set **TLCOCNUM =** to zero for next undeleted occurrence, or set to greater than zero for a specific occurrence.
4. Set **TLCDATAD =** to address of new data. You must also build this data in storage if not already in storage.
5. Set **TLCDATLN =** to the length of new data; it must be a range from zero to 65,535.
6. Set **TLCTAGDA =** to zero.
7. Set **TLCFLAGI =** to the desired options; zero is normal.
8. Call **DMGTAGL** passing the TLCB as the only parameter.
9. Test the return code. If not zero, handle appropriately.

If the return code is zero, then the exit program has the following value set by DMGTAGL:

Field Name	Set Value
TLCTAGDA =	internal table entry for this tag; of no use to user exit program, but be aware the value has changed.
TLCFLAGO =	bits set as needed.
TLCRETC =	return code
TLCREAC =	reason code

DMGTAGL Return/Reason Codes

Return	Reason	Description
0	0	NORMAL
4	0	Tag name not found; EOF on sequential read.
12	all	"C" (TLCCHGV) error.
12	1	Tag name mismatch between TLCTGNAM and TDSNAME. This means item TLCTAGDA contained a non-zero value, but TLCTGNAM is not equal to the TDSNAME (tag name) value in the tag dsect pointed to by TLCTAGDA.
12	2	Data length (TLCDATLN) is negative or > 65535.
16	0	Invalid command.
20	all	Internal program logic error. Should be fatal and produce a core dump. VRCREAC contains an internal logic error number. Calling program should display this value.
20	1	Length error reading data from LM/MM.
20	2	Tag area forward chain does not point to valid TAGA.
24	**	DMGLMMM error. This is a fatal Documerge error.
28	all	GETMAIN/GETVIS failure. VRCREAC is set to zero if "out of storage". VRCREAC is set to the return code from GETMAIN/GETVIS if not "out of storage".
32	all	Error in DMGSTATS, a Documerge program to maintain certain reserved tags. TLCREAC = DMGSTATS return code. This is a fatal Documerge error.
36	all	Error processing DMG.C. tag. Processing of this DMG.C. tag stops. DMGTAGL sets the following if DMGRETC=36: TLCDATAD = A(bad spot in data causing the error) TLCDATLN = offset in data of bad spot TLCREAC, described below. NOTE: See also return code 40.
36	200	"T" format tag name is blank (TFMTMAND = N).
36	204	"T" format tag name not found (TFMTMAND = M or N).
36	208	"C" format program name not found.
36	212	Nesting error...Exceeds maximum allowed nested DMG.C. tags.
36	216	Output area overflow (>32,759 bytes).
36	220	Invalid format for data.
36	224	"T" format tag value too large for output area size (pos 32-36 output area size is too small).
40	all	Non-zero return code from DMG.C. tag user exit. Halts processing of this DMG.C. tag. TLCDATAD = A(user exit command in data). TLCOCNUM = offset in data to user exit command. TLCREAC = user exit return code.

Optional Command Tag User-Exit Subprogram

This topic provides a technical reference for the programmer who wants to code a custom command tag user-exit subprogram.

You can use a command tag user-exit subprogram to dynamically manipulate data and/or maintain user statistics during the DMGMERGE run. Command tag user-exit programs are linked as load modules, and follow all standard IBM linkage conventions. These load modules must be available to the DMGMERGE step, usually through a JOBLIB or STEPLIB JCL statement.

The CALL command sets and passes the user-exit subprogram's input parameters. The user-exit subprogram receives two parameters: 1) a control block and 2) a character string that contains the value and length of the current command output area.

Each user-exit subprogram for each DMG.C.*xxx* tag receives input when the subprogram starts processing; each subprogram receives input again after all DMG.C.*xxx* tags for all forms processed, at the end of DMGMERGE processing. The user-exit subprogram function code tells the subprogram which condition applies. This function code is located in position 17 of the control block, and is passed to the subprogram dynamically. The user-exit subprogram input parameters depend on which function code is passed to the subprogram.

The first function code passed to the user-exit subprogram is P (Process). This code tells the subprogram to process data. Two parameters are passed to the user-exit subprogram: the control block and the command output area. All control block fields and the current value of the command output area are available for reference. The subprogram modifies the command output area as needed. A user-exit subprogram can be called with the P function code each time a CALL command occurs for this subprogram within each DMG.C.*xxx* tag.

The second function code passed to the user-exit subprogram is T (Terminate). This code tells the subprogram to stop processing. Only one parameter is passed to the user-exit subprogram: the control block. You must code the user-exit subprogram so that it *does not reference* any control block fields beyond the function code (position 17) when the function code is T.

Because all DMG.C.*xxx* tags have been processed when the subprogram function code is T, the command output area no longer contains a current value; therefore, the subprogram does not receive the command output area as input. The user-exit subprogram completes any remaining tasks, such as closing files or writing statistical reports, and processing terminates. A user-exit subprogram is called with the T function code only once, at the end of DMGMERGE processing, and only if the subprogram was called previously with the P function code.

CALL Command Control Block

The first CALL command parameter is the control block. This parameter contains eight fields. The first four fields of the control block create a standard Documerge control block header; the user-exit subprogram can use these four fields as input. The remaining fields contain data that the user-exit subprogram can either use or ignore, depending on how the program is written. The user-exit subprogram chooses whether and how to use the fields in the control block parameter.

Using the CALLRETC Return Code to Control Document Package Processing

CALLRETC, the CALL command control block return code, can be set by a DMG.C.xxx tag user exit to -1 (minus one or all hex FF or Cobol HIGH-VALUE). The -1 value directs DMGMERGE not to print the the current Document Package (to treat it as if COPIES=0).

There are no DMGMERGE messages generated if CALLRETC= -1 (minus one). If desired, you can program your user exit to write error messages to its own message file to indicate when and why DMGMERGE skipped a particular Document Package.

You can set CALLRETC= -1 (minus one) to dynamically select one Document Package instead of another Document Package in the same processing run.

NOTE

This does not work for DMG.CHECKPOINT.END.PACKAGE since at this time the Document Package has been written to the output file.

Command Output Area

The second CALL command parameter is the entire command output area for this DMG.C.*xxx* tag. This parameter is passed only if the subprogram function code is P (control block position 17). The value for this parameter is the halfword (two bytes) binary length of the current command output character string followed by the output character string. (The halfword length does not include itself). For example, an empty string has a length of 0; the string "ABC" has a length of 3.

This string is both the input and the output for the user-exit subprogram. When DMG.C.*xxx* tag processing begins, the calling program uses the string value as input. After processing is complete, the calling program resets this string to the new value; it changes the length field and the character string as needed. The minimum length-field value is zero; the maximum length-field value is 32,759.

Control Block Parameter Table

Name	Position	Length	Description
CALLID	01-04	4	**CALL**, a constant value identifying this control block. This field is input to the user-exit subprogram.
CALLFLEN	05-08	4	The fullword binary length of the control block: **29** (**x'0000001D'**). This field is input to the user-exit subprogram. **NOTE:** The user-exit subprogram can determine dynamically if new fields exist in this control block. ■ The subprogram can use the fullword control block length in positions 05-08 to check for new fields. ■ Or, the subprogram can use positions 09-12 (VVRR) of the Documerge version and release number and positions 15-16 of the control block modification number to check for new fields. New fields are added only at the control block's end. Existing called programs work even when new fields are added. Compatibility is always maintained.
CALLVRL	09-14	6	The current Documerge version, release, and level numbers: **VVRRLL**, using six digits. For Documerge 3.2, the number is 030200. This field is input to the user-exit subprogram.
CALLMOD	15-16	2	The control block modification number, using two digits. This number is incremented only when this control block is changed; a new Documerge level does not always create a control block modification. Currently, this value is 00. This field is input to the user-exit subprogram. **NOTE:** The user-exit subprogram can determine dynamically if new fields exist in this control block. ■ The subprogram can use positions 09-12 (VVRR) of the Documerge version and release number and positions 15-16 of the control block modification number to check for new fields. ■ Or, the subprogram can use the fullword length of this control block in positions 05-08 to check for new fields. New fields are added only at the control block's end. Existing called programs work even when new fields are added. Compatibility is always maintained.
CALLFUNC	17	1	The user-exit subprogram function code. This code tells the user-exit subprogram to start or stop processing. Valid values are: **P** Process. The user-exit subprogram receives the control block and the command output area as input parameters. The user-exit subprogram modifies the command output area as needed. **T** Terminate. The user-exit subprogram receives only the control block as an input parameter and *should not reference* any control block fields beyond the function code; it does not receive the command output area. The user-exit subprogram completes any remaining tasks and terminates.

Name	Position	Length	Description
The following fields apply only if the subprogram function code is P.			
	18-20	3	Filler for alignment. This value is not used.
CALLRETC	21-24	4	The fullword (four bytes) binary return code set by the user-exit subprogram. This is a value from 0-99,999. This return code is preset to 0. A non-zero value indicates an error and stops processing for this DMG.C.*xxx* tag. Beginning with Documerge 3.1, if the return code is set to minus one (all hex FF), Documerge will ignore the current Document Package with no warning or error messages as if the Document Package was never requested.
CALLBPSL	25-28	4	The fullword (four bytes) binary length of the BPSD. This is the value of LENGTH in the BPSD--the number of BPSD replacement characters. The user-exit subprogram can use this value for right justification or centering.
CALLAST	29-30	2	The halfword (two bytes) binary length of the last command output area returned by a program call in this DMG.C.*xxx* tag. The user-exit subprogram can use this value to process only new data appended to the command output area since the last call to any subprogram for this DMG.C.*xxx* tag. If no program call has been executed previously for this DMG.C.*xxx* tag, this value is zero. If this value equals the length of the input string, no new command values have been added since the last program call. This value is useful when two or more CALL commands exist in a single DMG.C.*xxx* tag, and the programs want to process only the new values rather than the entire command output area.
CALLCONS	31-38	8	The eight-character value that CALL command passes to the user-exit subprogram. This is the value from positions 10-17 in the CALL Command Coding Table. Refer to "Coding the CALL Command" on page 340. The user-exit subprogram chooses whether and how to use this value.

COBOL Skeleton of User-Exit Subprogram for DMG.C.*xxx* Tags

```
DATA DIVISION.

LINKAGE SECTION.

 01  CALL-CONTROL-BLOCK.
*** First parameter ***

     05  CCB-ID                  PIC X(4).
*** above will be constant "CALL" ***

     05  CCB-LENGTH              PIC S9(9) COMP.

     05  CCB-VVRRLL.
         10  CCB-VERSION         PIC 99.
         10  CCB-RELEASE         PIC 99.
         10  CCB-LEVEL           PIC 99.

     05  CCB-MODIFICATION        PIC 99.
*** above initially will be "00" ***

     05  CCB-FUNCTION            PIC X.
*** above is "P" to process or "T" for termination (final CALL).
*** You get "P" for each CALL Command for each DMG.C.xxx Tag.
*** You get "T" at end of DMGMERGE processing, and only once.
*** You must handle the "T" function, even if nothing to do.

     05  FILLER                  PIC X(3).
*** for fullword alignment of next field.

     05  CCB-RETURN-CODE         PIC S9(9) COMP.
*** Above will be zero at entry; you change this to non-zero ***
*** to indicate an error and reject the DMG.C.xxx Tag.       ***

     05  CCB-BPSD-LENGTH         PIC S9(9) COMP.
 01  DYNAMIC-OUTPUT-AREA.
*** Second parameter; use only if CCB-FUNCTION is "T".

     05  DOA-LENGTH              PIC S9(4) COMP.
*** This is the length of the data that follows.
*** Documerge sets this to the current DOA length.
*** You can reset this to any new length, either less or more.

     05  DOA-DATA.
          10  FILLER              PIC X
              OCCURS 0 TO 9999 TIMES DEPENDING ON DOA-LENGTH.
*  Actually can occur 0 to 32,759 times, but 9,999 is more than
*  enough for this program's use.  Of course, there is a way
*  to compile that allows for PIC S9(4) items to extend
*  up to 32,767 in value if you really need this
*  (the NOTRUNC option).
*  This method sets item DOA-DATA to the appropriate length.
*  You might use another method to handle the variable length.

PROCEDURE DIVISION  USING  CALL-CONTROL-BLOCK
                           DYNAMIC-OUTPUT-AREA.

       --- your program logic goes here ---
       GOBACK.
```

Special Considerations for MVS COBOL II

Documerge is written in PL/I and ALC (IBM mainframe assembler). If you use MVS and your user-exit subprogram is written in COBOL II, you must initialize the MVS COBOL II environment by coding the INITENV= control card in a GLOBAL command in the DMGMERGE JCL. (Refer to "INITENV=" on page 392 for more information.)

Several other ways of initializing the MVS COBOL II environment are documented in "Top-Level Programs Invoked as Subprograms" of IBM's *VS COBOL II Application Programming Guide*.

The following figure shows the code for a front-end ALC program to initialize the MVS COBOL II environment. If you use MVS COBOL II for your user-exit subprogram, enter this ALC program and link it with your MVS COBOL II program, with the ALC program as the entry point.

ALC program to initialize the MVS COBOL II environment

```
FRONTEND START 0
         *  LINK THIS WITH YOUR MVS COBOL II PROGRAM SPECIFYING "ENTRY
FRONTEND".
         *  CHANGE "PROGRAMNAME" TO YOUR COBOL II PROGRAM NAME.
         USING FRONTEND,15
MODINSTR B     FIRSTIME          *** CHANGED TO NOP ***
         L     15,=V(PROGNAME)   *** CHANGE THIS TO YOUR COBOL II NAME ***
         DROP  15
         BR    15
FIRSTIME DS    0H
         STM   14,12,12(13)
         LR    12,15
         USING FRONTEND,12
         LA    2,SAVEAREA
         ST    2,8(13)
         ST    13,4(2)
         LR    13,2
         LOAD  EP=ILBOSTP0       *** INITIALIZE COBOL II ***
         LR    15,0
         BASR  14,15
         NI    MODINSTR+1,X'0F'  *** CHANGES B TO NOP ***
         L     13,4(13)
         LM    14,12,12(13)
         DROP  12
         USING FRONTEND,15
         B     MODINSTR+4
         DROP  15
SAVEAREA DC    18F'0'
         END   FRONTEND
```

Initialization Procedure

Change **PROGNAME** to your COBOL II subprogram name (the name in the PROGRAM-ID in the IDENTIFICATION DIVISION).

Assemble this code and place the text module in a library.

Link this code to your load library with this same COBOL II subprogram name, specifying **ENTRY FRONTEND**.

Verify in the Linkage Editor listing that the entry point is FRONTEND.

VRF Structure and Example

This section is a technical description of the VRF, and also provides a sample VRF.

The VRF is a series of records containing tagged data. Tag data is made up of variable data for merging onto forms, or data used by DMGMERGE for selection of forms and for collation.

The VRF does not have any limitations on the number of tags. However, during DMGMERGE, all tags and data for a single Merge Set should be contained in memory for fastest processing. The VRF Allocation (DMGVRFA) file tells DMGMERGE exactly how much memory to allocate for reading in the tags and data for the largest Merge Set in the VRF. Therefore, all tags and data usually fit in memory automatically.

A single tag may be a maximum of 65K bytes long. Data for tags does not require trailing blanks, and if trailing blanks are present, DMGRFMT will remove these. If you call DMGVRFWR directly to write a tag and its data, you should first call ISIFLAST to remove the trailing blanks (see "The ISIFLAST Subprogram" on page 243). At the time of variable data merging by DMGMERGE, trailing blanks are added to data if it is shorter than the length specified in the boilerplate index.

There is a sample VRF in "DMG.C.xxx Reserved Tag Processing" on page 331.

VRF Tag Structure

The following is the structure of a tag in the VRF:

- Byte 1 — unsigned binary length of tag name
- Byte 2 — tag name
- Bytes following tag name, 2 byte unsigned binary length of data
- Bytes following length of data, variable data assigned to tag name.

Every tag in the VRF has the above structure. A series of tags ended with a byte of HIGH VALUES comprises a Merge Set. A series of Merge Sets comprises the VRF.

Sample VRF

The following VRF was produced by the installation demo. See "VRF Structure and Example" on page 353 for more information.

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  NAME  Brandon M. Williams AGE  27 SEX  Male POLIC
ZONE 0DCDC01C9898994D44E899889A0CCC00FF0ECE00D8980DDDCC
NUMR 4514503291546504B069339142317502273257044135D76393
    01...5...10...15...20...25...30...35...40...45...50
CHAR Y.NUMBER  1-966-3452 DMG.MERGESET.ID  1-966-3452 P
ZONE E4DEDCCD00F6FFF6FFFF0CDC4DCDCCECE4CC00F6FFF6FFFF0D
NUMR 8B5442590A1096603452F447B45975253B940A1096603452B7
    01...5...10...15...20...25...30...35...40...45...50
CHAR OLICY.DATE  Jan 18 1984 SUM.INSURED  $ 25,000 PREM
ZONE DDCCE4CCEC00D894FF4FFFF0EED4CDEEDCC0054FF6FFF0DDCD
NUMR 63938B41350B11501801984B244B952495408B025B000D7954
    01...5...10...15...20...25...30...35...40...45...50
CHAR IUM.CLASS  Standard BENEFIT.DESC  Flexible Whole L
ZONE CED4CDCEE00EA8988980CCDCCCE4CCEC01C98A88984E89984D
NUMR 944B331220823154194C2555693B4523036357923506863503
    01...5...10...15...20...25...30...35...40...45...50
CHAR ife TERMINATION.PERIOD  Life ANNUAL.MAX.1  $166.25
ZONE 8881ECDDCDCECDD4DCDCDC00D8880CDDECD4DCE4F005FFF4FF
NUMR 9652359495139 65B759964043965C155413B417B107B166B25
    01...5...10...15...20...25...30...35...40...45...50
CHAR  ANNUAL.MAX.2  $205.25 PREMIUM.PERIOD  3 Years The
ZONE 0CDDECD4DCE4F005FFF4FF0DDCDCED4DCDCDC02F4E889A4E88
NUMR C155413B417B207B205B25E7954944B7599640230851920385
    01...5...10...15...20...25...30...35...40...45...50
CHAR reafter to Jan 17, 2048 ANNUAL.PERIOD  $166.25 DMG
ZONE 9888A894A94D894FF64FFFF0CDDECD4DCDCDC005FFF4FF1CDC
NUMR 95163590360115017B02048D155413B75996407B166B250447

    01...5...10...15...20...25...30...35...40...45...50
CHAR .GCPY.INSURED  01 DMG.GCPY.FILE  01 DDNAME.INSURED
ZONE 4CCDE4CDEEDCC00FF0CDC4CCDE4CCDC00FF0CCDCDC4CDEEDCC
NUMR B7378B95249540201D447B7378B69350201E445145B9524954
    01...5...10...15...20...25...30...35...40...45...50
CHAR   INSURED DDNAME.FILE  FILE DMG.FLST.9700.INSURED
ZONE 00CDEEDCC0CCDCDC4CCDC00CCDC1CDC4CDEE4FFFF4CDEEDCC0
NUMR 079524954B445145B69350469355447B6323B9700B95249540
    01...5...10...15...20...25...30...35...40...45...50
CHAR yLM.LIFE.OVERLAY                  00001     LM.LIFE
ZONE ADD4DCCC4DECDDCE4444444444444444FFFFF44444DD4DCCC
NUMR 834B3965B6559318000000000000000000000010000034B3965
    01...5...10...15...20...25...30...35...40...45...50
CHAR .DEC                       00001     LM.LIFE.OVERLAY
ZONE 4CCC4444444444444444444FFFFF44444DD4DCCC4DECDDCE
NUMR B45300000000000000000000000010000034B3965B6559318
    01...5...10...15...20...25...30...35...40...45...50
CHAR                    00001     LM.LIFE.CONTRACT
ZONE 4444444444444444FFFFF44444DD4DCCC4CDDEDCCE4444444
NUMR 00000000000000000000010000034B3965B365391330000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR           00001      DMG.OPT.9700.INSURED {DUP SEP
ZONE 444444444FFFFF444441CDC4DDE4FFFF4CDEEDCC0CCED4ECD4
NUMR 00000000000001000004447B673B9700B95249540044702570
    01...5...10...15...20...25...30...35...40...45...50
CHAR MAI STA ODD POR OVL US1 US2 US3 US4 US5 DUP SEP MA
ZONE DCC4EEC4DCC4DDD4DED4EEF4EEF4EEF4EEF4CED4ECD4ECD4DC
NUMR 41902310644076906530421042204230424042504470257041
    01...5...10...15...20...25...30...35...40...45...50
CHAR I STA ODD POR ON  US1 US2 US3 US4 US5 DUP SEP AUX
ZONE C4EEC4DCC4DDD4DD44EEF4EEF4EEF4EEF4EEF4CED4ECD4CEE4
NUMR 902310644076906500421042204230424042504470257014 70
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR STA ODD POR OVL                        DUP SEP AUX ST
ZONE EEC4DCC4DDD4DED4444444444444444444444CED4ECD4CEE4EE
NUMR 2310644076906530000000000000000000000044702570147023
    01...5...10...15...20...25...30...35...40...45...50
CHAR A ODD POR ON                          DMG.SRC.9700.IN
ZONE C4DCC4DDD4DD4444444444444444444444441CDC4EDC4FFFF4CD
NUMR 1064407690650000000000000000000000004447B293B9700B95
    01...5...10...15...20...25...30...35...40...45...50
CHAR SURED   DMG.IDEF.9700.INSURED   DMG.SKEY.9700.INSU
ZONE EEDCC001CDC4CCCC4FFFF4CDEEDCC001CDC4EDCE4FFFF4CDEE
NUMR 24954005447B9456B9700B9524954005447B2258B9700B9524
    01...5...10...15...20...25...30...35...40...45...50
CHAR RED  01Brandon M. Williams 27 Male  1-966-3452Jan
ZONE DCC03FFC9898994D44E899889A4FF4D89844F6FFF6FFFFD894
NUMR 9540501291546504B069339142027041350010966034521150
    01...5...10...15...20...25...30...35...40...45...50
CHAR 18 1984  DMG.FLST.9700.FILE  LM.LIFE.DEC
ZONE FF4FFFF41CDC4CDEE4FFFF4CCDC05DD4DCCC4CCC4444444444
NUMR 180198402447B6323B9700B69350434B3965B4530000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR            00001     LM.LIFE.CONTRACT
ZONE 44444444444FFFFF44444DD4DCCC4CDDEDCCE4444444444444
NUMR 00000000000000010000034B3965B3653913300000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR    00001      DMG.OPT.9700.FILE -SIM SEP SRC STA O
ZONE 444FFFFF444441CDC4DDE4FFFF4CCDC06ECD4ECD4EDC4EEC4D
NUMR 00000001000001447B673B9700B6935002940257029302310644
    01...5...10...15...20...25...30...35...40...45...50
CHAR DD POR OFF                          SIM SEP SRC STA ODD
ZONE CC4DDD4DCC4444444444444444444ECD4ECD4EDC4EEC4DCC
NUMR 4407690666000000000000000000002940257029302310644
    01...5...10...15...20...25...30...35...40...45...50
CHAR  POR OFF                          DMG.SRC.9700.FILE
ZONE 4DDD4DCC4444444444444444444441CDC4EDC4FFFF4CCDC010
NUMR 0769066600000000000000000001447B293B9700B6935000
    01...5...10...15...20...25...30...35...40...45...50
CHAR  FEED10  FEED30 DMG.IDEF.9700.FILE   DMG.SKEY.9700
ZONE 0CCCCFF00CCCCFF1CDC4CCCC4FFFF4CCDC001CDC4EDCE4FFFF
NUMR 665541006654302447B9456B9700B6935002447B2258B9700
    01...5...10...15...20...25...30...35...40...45...50
CHAR .FILE  021-966-3452 DMG.FLST.TUMB.INSURED  LM.LIFE
ZONE 4CCDC00FFF6FFF6FFFF1CDC4CDEE4EEDC4CDEEDCC05DD4DCCC
NUMR B69350C0210966034525447B6323B3442B95249540434B3965
    01...5...10...15...20...25...30...35...40...45...50
CHAR .DEC                       00001     LM.LIFE.CONTRAC
ZONE 4CCC4444444444444444444FFFFF44444DD4DCCC4CDDEDCC
NUMR B45300000000000000000000000010000034B3965B3653913
    01...5...10...15...20...25...30...35...40...45...50
CHAR T                 00001      DMG.OPT.TUMB.INSURED -
ZONE E4444444444444444FFFFF444441CDC4DDE4EEDC4CDEEDCC06
NUMR 3000000000000000000001000004447B673B3442B952495400
    01...5...10...15...20...25...30...35...40...45...50
CHAR TUM SEP MAI STA ODD POR OFF                        TU
ZONE EED4ECD4DCC4EEC4DCC4DDD4DCC4444444444444444444444EE
NUMR 3440257041902310644076906660000000000000000000034
    01...5...10...15...20...25...30...35...40...45...50
CHAR M SEP MAI STA ODD POR OFF                        DMG
ZONE D4ECD4DCC4EEC4DCC4DDD4DCC4444444444444444444441CDC
NUMR 4025704190231064407690666000000000000000000004447
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR .SRC.TUMB.INSURED   DMG.IDEF.TUMB.INSURED   DMG.SK
ZONE 4EDC4EEDC4CDEEDCC001CDC4CCCC4EEDC4CDEEDCC001CDC4ED
NUMR B293B3442B9524954005447B9456B3442B9524954005447B22
    01...5...10...15...20...25...30...35...40...45...50
CHAR EY.TUMB.INSURED  03Brandon M. Williams  DMG.FLST.T
ZONE CE4EEDC4CDEEDCC01FFC9898994D44E899889A41CDC4CDEE4E
NUMR 58B3442B952495406032915465O4B06933914202447B6323B3
    01...5...10...15...20...25...30...35...40...45...50
CHAR UMB.FILE  LM.LIFE.DEC                      00001
ZONE EDC4CCDC05DD4DCCC4CCC4444444444444444444444FFFFF444
NUMR 442B69350434B3965B453000000000000000000000001000
    01...5...10...15...20...25...30...35...40...45...50
CHAR   LM.LIFE.CONTRACT                 00001      DMG.O
ZONE 44DD4DCCC4CDDEDCCE4444444444444444FFFFF444441CDC4D
NUMR 0034B3965B365391330000000000000000000001000001447B6
    01...5...10...15...20...25...30...35...40...45...50
CHAR PT.TUMB.FILE -TUM SEP MAI STA ODD POR OFF
ZONE DE4EEDC4CCDC06EED4ECD4DCC4EEC4DCC4DDD4DCC444444444
NUMR 73B3442B69350034402570419023106440769066600000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR              TUM SEP MAI STA ODD POR OFF
ZONE 444444444444EED4ECD4DCC4EEC4DCC4DDD4DCC44444444444
NUMR 000000000000344025704190231064407690666000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR             DMG.SRC.TUMB.FILE   DMG.IDEF.TUMB.FILE
ZONE 44444444441CDC4EDC4EEDC4CCDC001CDC4CCCC4EEDC4CCDC0
NUMR 00000000001447B293B3442B6935002447B9456B3442B69350

    01...5...10...15...20...25...30...35...40...45...50
CHAR   DMG.SKEY.TUMB.FILE  041-966-3452 DMG.MISSING.FOR
ZONE 01CDC4EDCE4EEDC4CCDC00FFF6FFF6FFFF1CDC4DCEECDC4CDD
NUMR 02447B2258B3442B69350C0410966034521447B4922957B669
    01...5...10...15...20...25...30...35...40...45...50
CHAR MS  EAST.BACK.OVERLAY                 00000     EAST
ZONE DE05CCEE4CCCD4DECDDCE444444444444444FFFFF00000CCEE
NUMR 42045123B2132B6559318000000000000000000000005123
    01...5...10...15...20...25...30...35...40...45...50
CHAR .FRONT.OVERLAY               00000       NAME  Rach
ZONE 4CDDDE4DECDDCE44444444444444FFFFF00000F0DCDC00D888
NUMR B69653B6559318000000000000000000000000F451450F9138
    01...5...10...15...20...25...30...35...40...45...50
CHAR el F. Scott AGE  37 SEX  Female POLICY.NUMBER  1-0
ZONE 894C44E89AA0CCC00FF0ECE00C898980DDDCCE4DEDCCD00F6F
NUMR 5306B023633317502373257066541350763938B5442590A100
    01...5...10...15...20...25...30...35...40...45...50
CHAR 85-9833 DMG.MERGESET.ID  1-085-9833 POLICY.DATE  D
ZONE FF6FFFF0CDC4DCDCCECE4CC00F6FFF6FFFF0DDDCCE4CCEC00C
NUMR 8509833F447B45975253B940A1008509833B763938B41350B4
    01...5...10...15...20...25...30...35...40...45...50
CHAR ec 15 1983 SUM.INSURED  $ 20,000 PREMIUM.CLASS  St
ZONE 884FF4FFFF0EED4CDEEDCC0054FF6FFF0DDCDCED4CDCEE00EA
NUMR 5301501983B244B952495408B020B000D7954944B331220823
    01...5...10...15...20...25...30...35...40...45...50
CHAR andard BENEFIT.DESC  Flexible Whole Life TERMINATI
ZONE 8988980CCDCCCE4CCEC01C98A88984E89984D8881ECDDCDCEC
NUMR 154194C2555693B4523036357923506863503965235949513 9
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR ON.PERIOD  Life ANNUAL.MAX.1  $134.25 ANNUAL.MAX.2
ZONE DD4DCDCDC00D8880CDDECD4DCE4F005FFF4FF0CDDECD4DCE4F
NUMR 65B759964043965C155413B417B107B134B25C155413B417B2
    01...5...10...15...20...25...30...35...40...45...50
CHAR   $176.50 PREMIUM.PERIOD  3 Years Thereafter to De
ZONE 005FFF4FF0DDCDCED4DCDCDC02F4E889A4E889888A894A94C8
NUMR 07B176B50E7954944B75996402308519203859516359036045
    01...5...10...15...20...25...30...35...40...45...50
CHAR c 14, 2047 ANNUAL.PERIOD  $145.00 DMG.GCPY.INSURED
ZONE 84FF64FFFF0CDDECD4DCDCDC005FFF4FF1CDC4CCDE4CDEEDCC
NUMR 3014B02047D155413B75996407B145B000447B7378B9524954
    01...5...10...15...20...25...30...35...40...45...50
CHAR   01 DMG.GCPY.FILE  01 DDNAME.INSURED  INSURED DDN
ZONE 00FF0CDC4CCDE4CCDC00FF0CCDCDC4CDEEDCC00CDEEDCC0CCD
NUMR 0201D447B7378B69350201E445145B9524954079524954B445
    01...5...10...15...20...25...30...35...40...45...50
CHAR AME.FILE  FILE DMG.FLST.9700.INSURED yLM.LIFE.OVER
ZONE CDC4CCDC00CCDC1CDC4CDEE4FFFF4CDEEDCC0ADD4DCCC4DECD
NUMR 145B69350469355447B6323B9700B95249540834B3965B6559
    01...5...10...15...20...25...30...35...40...45...50
CHAR LAY                 00001     LM.LIFE.DEC
ZONE DCE44444444444444444FFFFF44444DD4DCCC4CCC444444444
NUMR 318000000000000000000010000034B3965B453000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR             00001     LM.LIFE.OVERLAY
ZONE 444444444444FFFFF44444DD4DCCC4DECDDCE4444444444444
NUMR 00000000000000010000034B3965B6559318000000000000000
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR     00001     LM.LIFE.CONTRACT                 0000
ZONE 4444FFFFF44444DD4DCCC4CDDEDCCE4444444444444444FFFF
NUMR 000000010000034B3965B365391330000000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 1      DMG.OPT.9700.INSURED {DUP SEP MAI STA ODD P
ZONE F444441CDC4DDE4FFFF4CDEEDCC0CCED4ECD4DCC4EEC4DCC4D
NUMR 1000004447B673B9700B952495400447025704190231064407
    01...5...10...15...20...25...30...35...40...45...50
CHAR OR OVL US1 US2 US3 US4 US5 DUP SEP MAI STA ODD POR
ZONE DD4DED4EEF4EEF4EEF4EEF4EEF4CED4ECD4DCC4EEC4DCC4DDD
NUMR 690653042104220423042404250447025704190231064407 69
    01...5...10...15...20...25...30...35...40...45...50
CHAR  ON  US1 US2 US3 US4 US5 DUP SEP AUX STA ODD POR O
ZONE 4DD44EEF4EEF4EEF4EEF4EEF4CED4ECD4CEE4EEC4DCC4DDD4D
NUMR 065004210422042304240425044702570147023106440769 06
    01...5...10...15...20...25...30...35...40...45...50
CHAR VL                       DUP SEP AUX STA ODD POR ON
ZONE ED4444444444444444444CED4ECD4CEE4EEC4DCC4DDD4DD4
NUMR 530000000000000000000044702570147023106440769065 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR                        DMG.SRC.9700.INSURED   DMG.I
ZONE 4444444444444444444441CDC4EDC4FFFF4CDEEDCC001CDC4C
NUMR 00000000000000000000044447B293B9700B9524954005447B9
    01...5...10...15...20...25...30...35...40...45...50
CHAR DEF.9700.INSURED   DMG.SKEY.9700.INSURED  01Rachel
ZONE CCC4FFFF4CDEEDCC001CDC4EDCE4FFFF4CDEEDCC03FFD88889
NUMR 456B9700B9524954005447B2258B9700B95249540501913853
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  F. Scott     37 Female1-085-9833Dec 15 1983  DMG.
ZONE 4C44E89AA44444FF4C89898F6FFF6FFFFC884FF4FFFF41CDC4
NUMR 06B0236330000037065413510085098334530150198302447B
    01...5...10...15...20...25...30...35...40...45...50
CHAR FLST.9700.FILE  LM.LIFE.DEC                      00
ZONE CDEE4FFFF4CCDC05DD4DCCC4CCC4444444444444444444FF
NUMR 6323B9700B69350434B3965B45300000000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 001     LM.LIFE.CONTRACT                 00001
ZONE FFF44444DD4DCCC4CDDEDCCE4444444444444444FFFFF44444
NUMR 0010000034B3965B365391330000000000000000000100000
    01...5...10...15...20...25...30...35...40...45...50
CHAR  DMG.OPT.9700.FILE -SIM SEP SRC STA ODD POR OFF
ZONE 1CDC4DDE4FFFF4CCDC06ECD4ECD4EDC4EEC4DCC4DDD4DCC444
NUMR 1447B673B9700B693500294025702930231064407690666000
    01...5...10...15...20...25...30...35...40...45...50
CHAR                  SIM SEP SRC STA ODD POR OFF
ZONE 444444444444444ECD4ECD4EDC4EEC4DCC4DDD4DCC44444
NUMR 00000000000000000294025702930231064407690666000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR                 DMG.SRC.9700.FILE    FEED10  FEED
ZONE 44444444444444441CDC4EDC4FFFF4CCDC0100CCCCFF00CCCC
NUMR 00000000000000001447B293B9700B693500066554 10066554
    01...5...10...15...20...25...30...35...40...45...50
CHAR 30 DMG.IDEF.9700.FILE   DMG.SKEY.9700.FILE  021-08
ZONE FF1CDC4CCCC4FFFF4CCDC001CDC4EDCE4FFFF4CCDC00FFF6FF
NUMR 302447B9456B9700B6935002447B2258B9700B69350C021008
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR 5-9833 DMG.FLST.TUMB.INSURED  LM.LIFE.DEC
ZONE F6FFFF1CDC4CDEE4EEDC4CDEEDCC05DD4DCCC4CCC444444444
NUMR 5098335447B6323B3442B95249540434B3965B453000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR              00001     LM.LIFE.CONTRACT
ZONE 444444444444FFFFF44444DD4DCCC4CDDEDCCE444444444444
NUMR 00000000000000010000034B3965B365391330000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR     00001      DMG.OPT.TUMB.INSURED -TUM SEP MAI S
ZONE 4444FFFFF444441CDC4DDE4EEDC4CDEEDCC06EED4ECD4DCC4E
NUMR 0000000010000044 47B673B3442B9524954003440257041902
    01...5...10...15...20...25...30...35...40...45...50
CHAR TA ODD POR OFF                        TUM SEP MAI STA
ZONE EC4DCC4DDD4DCC4444444444444444444444EED4ECD4DCC4EEC
NUMR 3106440769066600000000000000000000000344025704190231
    01...5...10...15...20...25...30...35...40...45...50
CHAR  ODD POR OFF                          DMG.SRC.TUMB.INS
ZONE 4DCC4DDD4DCC4444444444444444444441CDC4EDC4EEDC4CDE
NUMR 0644076906660000000000000000000004447B293B3442B952
    01...5...10...15...20...25...30...35...40...45...50
CHAR URED   DMG.IDEF.TUMB.INSURED   DMG.SKEY.TUMB.INSUR
ZONE EDCC001CDC4CCCC4EEDC4CDEEDCC001CDC4EDCE4EEDC4CDEED
NUMR 4954005447B9456B3442B9524954005447B2258B3442B95249
    01...5...10...15...20...25...30...35...40...45...50
CHAR ED  03Rachel F. Scott      DMG.FLST.TUMB.FILE  LM.
ZONE CC01FFD888894C44E89AA444441CDC4CDEE4EEDC4CCDC05DD4
NUMR 54060391385306B023633000002447B6323B3442B69350434B
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR LIFE.DEC                        00001     LM.LIFE.CON
ZONE DCCC4CCC4444444444444444444FFFFF44444DD4DCCC4CDD
NUMR 3965B453000000000000000000000010000034B3965B365
    01...5...10...15...20...25...30...35...40...45...50
CHAR TRACT                  00001      DMG.OPT.TUMB.FILE
ZONE EDCCE4444444444444444FFFFF444441CDC4DDE4EEDC4CCDC0
NUMR 391330000000000000000000001000001447B673B3442B69350
    01...5...10...15...20...25...30...35...40...45...50
CHAR -TUM SEP MAI STA ODD POR OFF                       T
ZONE 6EED4ECD4DCC4EEC4DCC4DDD4DCC4444444444444444444444E
NUMR 0344025704190231064407690666000000000000000000000003
    01...5...10...15...20...25...30...35...40...45...50
CHAR UM SEP MAI STA ODD POR OFF                        DM
ZONE ED4ECD4DCC4EEC4DCC4DDD4DCC4444444444444444444441CD
NUMR 4402570419023106440769066600000000000000000000144
    01...5...10...15...20...25...30...35...40...45...50
CHAR G.SRC.TUMB.FILE   DMG.IDEF.TUMB.FILE   DMG.SKEY.TU
ZONE C4EDC4EEDC4CCDC001CDC4CCCC4EEDC4CCDC001CDC4EDCE4EE
NUMR 7B293B3442B6935002447B9456B3442B6935002447B2258B34
    01...5...10...15...20...25...30...35...40...45...50
CHAR MB.FILE  041-085-9833 DMG.MISSING.FORMS  EAST.BACK
ZONE DC4CCDC00FFF6FFF6FFFF1CDC4DCEECDC4CDDDE05CCEE4CCCD
NUMR 42B69350C041008509833144 7B4922957B66942045123B2132
    01...5...10...15...20...25...30...35...40...45...50
CHAR .OVERLAY                00000     EAST.FRONT.OVERLA
ZONE 4DECDDCE444444444444444FFFFF00000CCEE4CDDDE4DECDDC
NUMR B6559318000000000000000000000005123B69653B655931
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR Y               00000       NAME  Justin Wakefield
ZONE E44444444444444FFFFF00000F0DCDC01DAAA894E898888980
NUMR 800000000000000000000000F45145001423950612569534 3
    01...5...10...15...20...25...30...35...40...45...50
CHAR AGE  32 SEX  Male POLICY.NUMBER  2-478-9047 DMG.ME
ZONE CCC00FF0ECE00D8980DDDCCE4DEDCCD00F6FFF6FFFF0CDC4DC
NUMR 17502323257044135D763938B5442590A2047809047F447B45
    01...5...10...15...20...25...30...35...40...45...50
CHAR RGESET.ID  2-478-9047 POLICY.DATE  Sep 28 1983 SUM
ZONE DCCECE4CC00F6FFF6FFFF0DDDCCE4CCEC00E894FF4FFFF0EED
NUMR 975253B940A2047809047B763938B41350B25702801983B244
    01...5...10...15...20...25...30...35...40...45...50
CHAR .INSURED  $ 25,000 PREMIUM.CLASS  Standard BENEFIT
ZONE 4CDEEDCC0054FF6FFF0DDCDCED4CDCEE00EA8988980CCDCCCE
NUMR B952495408B025B000D7954944B331220823154194C2555693
    01...5...10...15...20...25...30...35...40...45...50
CHAR .DESC  Flexible Whole Life TERMINATION.PERIOD  Lif
ZONE 4CCEC01C98A88984E89984D8881ECDDCDCECDD4DCDCDC00D88
NUMR B45230363579235068635039652359495139 5B75996404396
    01...5...10...15...20...25...30...35...40...45...50
CHAR e ANNUAL.MAX.1  $177.25 ANNUAL.MAX.2  $200.00 PREM
ZONE 80CDDECD4DCE4F005FFF4FF0CDDECD4DCE4F005FFF4FF0DDCD
NUMR 5C155413B417B107B177B25C155413B417B207B200B00E7954
    01...5...10...15...20...25...30...35...40...45...50
CHAR IUM.PERIOD  3 Years Thereafter to Sep 27, 2047 ANN
ZONE CED4DCDCDC02F4E889A4E889888A894A94E894FF64FFFF0CDD
NUMR 944B75996402308519203859516359036025702 7B02047D155
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR UAL.PERIOD  $235.00 DMG.GCPY.INSURED  01 DMG.GCPY.
ZONE ECD4DCDCDC005FFF4FF1CDC4CCDE4CDEEDCC00FF0CDC4CCDE4
NUMR 413B75996407B235B000447B7378B95249540201D447B7378B
    01...5...10...15...20...25...30...35...40...45...50
CHAR FILE  02 DDNAME.INSURED  INSURED DDNAME.FILE  FILE
ZONE CCDC00FF0CCDCDC4CDEEDCC00CDEEDCC0CCDCDC4CCDC00CCDC
NUMR 69350202E445145B95249540795249 54B445145B6935046935
    01...5...10...15...20...25...30...35...40...45...50
CHAR  DMG.FLST.9700.INSURED yLM.LIFE.OVERLAY
ZONE 1CDC4CDEE4FFFF4CDEEDCC0ADD4DCCC4DECDDCE44444444444
NUMR 5447B6323B9700B95249540834B3965B6559318000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR       00001     LM.LIFE.DEC                      00
ZONE 444444FFFFF44444DD4DCCC4CCC4444444444444444444444FF
NUMR 0000000000 10000034B3965B45300000000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 001     LM.LIFE.OVERLAY                  00001
ZONE FFF44444DD4DCCC4DECDDCE4444444444444444FFFFF44444
NUMR 0010000034B3965B6559318000000000000000000000100000
    01...5...10...15...20...25...30...35...40...45...50
CHAR LM.LIFE.CONTRACT                 00001      DMG.OPT
ZONE DD4DCCC4CDDEDCCE4444444444444444FFFFF444441CDC4DDE
NUMR 34B3965B365391330000000000000000000001000004447B673
    01...5...10...15...20...25...30...35...40...45...50
CHAR .9700.INSURED {DUP SEP MAI STA ODD POR OVL US1 US2
ZONE 4FFFF4CDEEDCC0CCED4ECD4DCC4EEC4DCC4DDD4DED4EEF4EEF
NUMR B9700B952495400447025704190231064407690653042104 22
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  US3 US4 US5 DUP SEP MAI STA ODD POR ON  US1 US2 U
ZONE 4EEF4EEF4EEF4CED4ECD4DCC4EEC4DCC4DDD4DD44EEF4EEF4E
NUMR 042304240425044702570419023106440769065004210422 04
    01...5...10...15...20...25...30...35...40...45...50
CHAR S3 US4 US5 DUP SEP AUX STA ODD POR OVL
ZONE EF4EEF4EEF4CED4ECD4CEE4EEC4DCC4DDD4DED444444444444
NUMR 230424042504470257014702310644076906530000000000 00
    01...5...10...15...20...25...30...35...40...45...50
CHAR           DUP SEP AUX STA ODD POR ON
ZONE 444444444CED4ECD4CEE4EEC4DCC4DDD4DD444444444444444
NUMR 000000000447025701470231064407690650000000000000 00
    01...5...10...15...20...25...30...35...40...45...50
CHAR          DMG.SRC.9700.INSURED   DMG.IDEF.9700.INSUR
ZONE 44444441CDC4EDC4FFFF4CDEEDCC001CDC4CCCC4FFFF4CDEED
NUMR 00000004447B293B9700B9524954005447B9456B9700B95249
    01...5...10...15...20...25...30...35...40...45...50
CHAR ED   DMG.SKEY.9700.INSURED  01Justin Wakefield
ZONE CC001CDC4EDCE4FFFF4CDEEDCC03FFDAAA894E898888984444
NUMR 54005447B2258B9700B952495405011423950612569534 0000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 32 Male  2-478-9047Sep 28 1983  DMG.FLST.9700.FILE
ZONE FF4D89844F6FFF6FFFFE894FF4FFFF41CDC4CDEE4FFFF4CCDC
NUMR 32041350020478090472570280198302447B6323B9700B6935
    01...5...10...15...20...25...30...35...40...45...50
CHAR   LM.LIFE.DEC                       00001     LM.LIF
ZONE 05DD4DCCC4CCC4444444444444444444444FFFFF44444DD4DCC
NUMR 0434B3965B45300000000000000000000000010000034B396
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR E.CONTRACT                 00001      DMG.OPT.9700.
ZONE C4CDDEDCCE444444444444444FFFFF444441CDC4DDE4FFFF4
NUMR 5B365391330000000000000000000001000001447B673B9700B
    01...5...10...15...20...25...30...35...40...45...50
CHAR FILE -SIM SEP SRC STA ODD POR OFF
ZONE CCDC06ECD4ECD4EDC4EEC4DCC4DDD4DCC44444444444444444
NUMR 693500294025702930231064407690666000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR     SIM SEP SRC STA ODD POR OFF
ZONE 4444ECD4ECD4EDC4EEC4DCC4DDD4DCC4444444444444444444
NUMR 000029402570293023106440769066600000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR    DMG.SRC.9700.FILE    FEED10  FEED30 DMG.IDEF.97
ZONE 441CDC4EDC4FFFF4CCDC0100CCCCFF00CCCCFF1CDC4CCCC4FF
NUMR 001447B293B9700B693500066554100665543024 47B9456B97
    01...5...10...15...20...25...30...35...40...45...50
CHAR 00.FILE   DMG.SKEY.9700.FILE  022-478-9047 DMG.FLS
ZONE FF4CCDC001CDC4EDCE4FFFF4CCDC00FFF6FFF6FFFF1CDC4CDE
NUMR 00B6935002447B2258B9700B69350C0220478090475447B632
    01...5...10...15...20...25...30...35...40...45...50
CHAR T.TUMB.INSURED  LM.LIFE.DEC                      00
ZONE E4EEDC4CDEEDCC05DD4DCCC4CCC4444444444444444444444FF
NUMR 3B3442B95249540434B3965B453000000000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 001     LM.LIFE.CONTRACT                 00001
ZONE FFF44444DD4DCCC4CDDEDCCE4444444444444444FFFFF44444
NUMR 0010000034B3965B365391330000000000000000000100000
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  DMG.OPT.TUMB.INSURED -TUM SEP MAI STA ODD POR OFF
ZONE 1CDC4DDE4EEDC4CDEEDCC06EED4ECD4DCC4EEC4DCC4DDD4DCC
NUMR 4447B673B3442B952495400344025704190231064407690666
    01...5...10...15...20...25...30...35...40...45...50
CHAR                        TUM SEP MAI STA ODD POR OFF
ZONE 4444444444444444444EED4ECD4DCC4EEC4DCC4DDD4DCC44
NUMR 000000000000000000003440257041902310644076906660 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR                     DMG.SRC.TUMB.INSURED   DMG.IDE
ZONE 4444444444444444441CDC4EDC4EEDC4CDEEDCC001CDC4CCC
NUMR 000000000000000000004447B293B3442B9524954005447B945
    01...5...10...15...20...25...30...35...40...45...50
CHAR F.TUMB.INSURED   DMG.SKEY.TUMB.INSURED  03Justin W
ZONE C4EEDC4CDEEDCC001CDC4EDCE4EEDC4CDEEDCC01FFDAAA894E
NUMR 6B3442B9524954005447B2258B3442B9524954060314239506
    01...5...10...15...20...25...30...35...40...45...50
CHAR akefield     DMG.FLST.TUMB.FILE  LM.LIFE.DEC
ZONE 8988889844441CDC4CDEE4EEDC4CCDC05DD4DCCC4CCC444444
NUMR 1256953400002447B6323B3442B69350434B3965B453000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR                00001     LM.LIFE.CONTRACT
ZONE 444444444444444FFFFF44444DD4DCCC4CDDEDCCE444444444
NUMR 000000000000000000010000034B3965B36539133000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR        00001      DMG.OPT.TUMB.FILE -TUM SEP MAI S
ZONE 4444444FFFFF444441CDC4DDE4EEDC4CCDC06EED4ECD4DCC4E
NUMR 000000000001000001447B673B3442B6935003440257041902
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR TA ODD POR OFF                        TUM SEP MAI STA
ZONE EC4DCC4DDD4DCC4444444444444444444444EED4ECD4DCC4EEC
NUMR 3106440769066600000000000000000000000344025704190231
    01...5...10...15...20...25...30...35...40...45...50
CHAR  ODD POR OFF                         DMG.SRC.TUMB.FIL
ZONE 4DCC4DDD4DCC4444444444444444444441CDC4EDC4EEDC4CCD
NUMR 0644076906660000000000000000000001447B293B3442B693
    01...5...10...15...20...25...30...35...40...45...50
CHAR E   DMG.IDEF.TUMB.FILE   DMG.SKEY.TUMB.FILE  042-4
ZONE C001CDC4CCCC4EEDC4CCDC001CDC4EDCE4EEDC4CCDC00FFF6F
NUMR 5002447B9456B3442B6935002447B2258B3442B69350C04204
    01...5...10...15...20...25...30...35...40...45...50
CHAR 78-9047 DMG.MISSING.FORMS  EAST.BACK.OVERLAY
ZONE FF6FFFF1CDC4DCEECDC4CDDDE05CCEE4CCCD4DECDDCE444444
NUMR 78090471447B4922957B66942045123B2132B6559318000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR          00000     EAST.FRONT.OVERLAY
ZONE 444444444FFFFF00000CCEE4CDDDE4DECDDCE4444444444444
NUMR 000000000000000000005123B69653B6559318000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR  00000       NAME  Lauren D. Huffman AGE  23 SEX
ZONE 4FFFFF00000F0DCDC01D8A9894C44CA889890CCC00FF0ECE00
NUMR 00000000000F451450131495504B0846641531750223325706
    01...5...10...15...20...25...30...35...40...45...50
CHAR 1-766-4361 POLICY.DATE  Jul 13 1983 SUM.INSURED  $
ZONE F6FFF6FFFF0DDDCCE4CCEC00DA94FF4FFFF0EED4CDEEDCC005
NUMR 1076604361B763938B41350B14301301983B244B952495408B
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  15,000 PREMIUM.CLASS  Standard BENEFIT.DESC  Flex
ZONE 4FF6FFF0DDCDCED4CDCEE00EA8988980CCDCCCE4CCEC01C98A
NUMR 015B000D7954944B3312208231541 94C2555693B4523036357
    01...5...10...15...20...25...30...35...40...45...50
CHAR ible Whole Life TERMINATION.PERIOD  Life ANNUAL.MA
ZONE 88984E89984D8881ECDDCDCECDD4DCDCDC00D8880CDDECD4DC
NUMR 923506863503965235949513965B759964043965C155413B41
    01...5...10...15...20...25...30...35...40...45...50
CHAR X.1  $ 98.00 ANNUAL.MAX.2  $115.50 PREMIUM.PERIOD
ZONE E4F0054FF4FF0CDDECD4DCE4F005FFF4FF0DDCDCED4DCDCDC0
NUMR 7B107B098B00C155413B417B207B115B50E7954944B7599640
    01...5...10...15...20...25...30...35...40...45...50
CHAR  3 Years Thereafter to Jul 12, 2047 ANNUAL.PERIOD
ZONE 2F4E889A4E889888A894A94DA94FF64FFFF0CDDECD4DCDCDC0
NUMR 230851920385951635903601430 12B02047D155413B7599640
    01...5...10...15...20...25...30...35...40...45...50
CHAR  $ 98.00 DMG.GCPY.INSURED  02 DMG.GCPY.FILE  02 DD
ZONE 054FF4FF1CDC4CCDE4CDEEDCC00FF0CDC4CCDE4CCDC00FF0CC
NUMR 7B098B000447B7378B95249540202D447B7378B69350202E44
    01...5...10...15...20...25...30...35...40...45...50
CHAR NAME.INSURED  INSURED DDNAME.FILE  FILE DMG.FLST.9
ZONE DCDC4CDEEDCC00CDEEDCC0CCDCDC4CCDC00CCDC1CDC4CDEE4F
NUMR 5145B9524954079524954B445145B69350469355447B6323B9
    01...5...10...15...20...25...30...35...40...45...50
CHAR 700.INSURED yLM.LIFE.OVERLAY                   00001
ZONE FFF4CDEEDCC0ADD4DCCC4DECDDCE44444444444444444FFFFF
NUMR 700B95249540834B3965B6559318000000000000000000001
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR      LM.LIFE.DEC                        00001     LM.
ZONE 44444DD4DCCC4CCC444444444444444444444FFFFF44444DD4
NUMR 0000034B3965B453000000000000000000000000010000034B
    01...5...10...15...20...25...30...35...40...45...50
CHAR LIFE.OVERLAY                   00001     LM.LIFE.CON
ZONE DCCC4DECDDCE4444444444444444FFFFF44444DD4DCCC4CDD
NUMR 3965B6559318000000000000000000000010000034B3965B365
    01...5...10...15...20...25...30...35...40...45...50
CHAR TRACT                 00001      DMG.OPT.9700.INSUR
ZONE EDCCE4444444444444444FFFFF444441CDC4DDE4FFFF4CDEED
NUMR 391330000000000000000000010000044 47B673B9700B95249
    01...5...10...15...20...25...30...35...40...45...50
CHAR ED {DUP SEP MAI STA ODD POR OVL US1 US2 US3 US4 US
ZONE CC0CCED4ECD4DCC4EEC4DCC4DDD4DED4EEF4EEF4EEF4EEF4EE
NUMR 5400447025704190231064407690653042104220423042404 2
    01...5...10...15...20...25...30...35...40...45...50
CHAR 5 DUP SEP MAI STA ODD POR ON  US1 US2 US3 US4 US5
ZONE F4CED4ECD4DCC4EEC4DCC4DDD4DD44EEF4EEF4EEF4EEF4EEF4
NUMR 5044702570419023106440769065004210422042304240425 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR DUP SEP AUX STA ODD POR OVL                      DU
ZONE CED4ECD4CEE4EEC4DCC4DDD4DED444444444444444444444CE
NUMR 44702570147023106440769065300000000000000000000044
    01...5...10...15...20...25...30...35...40...45...50
CHAR P SEP AUX STA ODD POR ON                        DMG
ZONE D4ECD4CEE4EEC4DCC4DDD4DD444444444444444444444441CDC
NUMR 70257014702310644076906500000000000000000000004447
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR .SRC.9700.INSURED   DMG.IDEF.9700.INSURED   DMG.SK
ZONE 4EDC4FFFF4CDEEDCC001CDC4CCCC4FFFF4CDEEDCC001CDC4ED
NUMR B293B9700B9524954005447B9456B9700B9524954005447B22
    01...5...10...15...20...25...30...35...40...45...50
CHAR EY.9700.INSURED  01Lauren D. Huffman   23 Female1-
ZONE CE4FFFF4CDEEDCC03FFD8A9894C44CA88989444FF4C89898F6
NUMR 58B9700B952495405013149550 4B0846641500023065413510
    01...5...10...15...20...25...30...35...40...45...50
CHAR 766-4361Jul 13 1983  DMG.FLST.9700.FILE  LM.LIFE.D
ZONE FFF6FFFFDA94FF4FFFF41CDC4CDEE4FFFF4CCDC05DD4DCCC4C
NUMR 766043611430130198302447B6323B9700B69350434B3965B4
    01...5...10...15...20...25...30...35...40...45...50
CHAR EC                     00001     LM.LIFE.CONTRACT
ZONE CC4444444444444444444FFFFF44444DD4DCCC4CDDEDCCE4
NUMR 530000000000000000000000000010000034B3965B365391330
    01...5...10...15...20...25...30...35...40...45...50
CHAR                 00001      DMG.OPT.9700.FILE -SIM S
ZONE 444444444444444FFFFF444441CDC4DDE4FFFF4CCDC06ECD4E
NUMR 00000000000000000001000001447B673B9700B69350029402
    01...5...10...15...20...25...30...35...40...45...50
CHAR EP SRC STA ODD POR OFF                       SIM SEP
ZONE CD4EDC4EEC4DCC4DDD4DCC4444444444444444444444ECD4ECD
NUMR 57029302310644076906660000000000000000000002940257
    01...5...10...15...20...25...30...35...40...45...50
CHAR  SRC STA ODD POR OFF                        DMG.SRC.
ZONE 4EDC4EEC4DCC4DDD4DCC4444444444444444444441CDC4EDC4
NUMR 0293023106440769066600000000000000000000001447B293B
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR 9700.FILE    FEED10  FEED30 DMG.IDEF.9700.FILE   D
ZONE FFFF4CCDC0100CCCCFF00CCCCFF1CDC4CCCC4FFFF4CCDC001C
NUMR 9700B693500066554100665543024 47B9456B9700B69350024
    01...5...10...15...20...25...30...35...40...45...50
CHAR MG.SKEY.9700.FILE  021-766-4361 DMG.FLST.TUMB.INSU
ZONE DC4EDCE4FFFF4CCDC00FFF6FFF6FFFF1CDC4CDEE4EEDC4CDEE
NUMR 47B2258B9700B69350C0210766043615447B6323B3442B9524
    01...5...10...15...20...25...30...35...40...45...50
CHAR RED  LM.LIFE.DEC                       00001     LM.
ZONE DCC05DD4DCCC4CCC444444444444444444444FFFFF44444DD4
NUMR 9540434B3965B453000000000000000000000010000034B
    01...5...10...15...20...25...30...35...40...45...50
CHAR LIFE.CONTRACT                 00001      DMG.OPT.TU
ZONE DCCC4CDDEDCCE4444444444444444FFFFF444441CDC4DDE4EE
NUMR 3965B365391330000000000000000001000004447B673B34
    01...5...10...15...20...25...30...35...40...45...50
CHAR MB.INSURED -TUM SEP MAI STA ODD POR OFF
ZONE DC4CDEEDCC06EED4ECD4DCC4EEC4DCC4DDD4DCC44444444444
NUMR 42B952495400344025704190231064407690666000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR           TUM SEP MAI STA ODD POR OFF
ZONE 4444444444EED4ECD4DCC4EEC4DCC4DDD4DCC4444444444444
NUMR 0000000000344025704190231064407690666000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR          DMG.SRC.TUMB.INSURED   DMG.IDEF.TUMB.INSU
ZONE 444444441CDC4EDC4EEDC4CDEEDCC001CDC4CCCC4EEDC4CDEE
NUMR 000000004447B293B3442B9524954005447B9456B3442B9524
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR RED   DMG.SKEY.TUMB.INSURED  03Lauren D. Huffman
ZONE DCC001CDC4EDCE4EEDC4CDEEDCC01FFD8A9894C44CA8898944
NUMR 954005447B2258B3442B952495406033149550 4B0846641500
    01...5...10...15...20...25...30...35...40...45...50
CHAR   DMG.FLST.TUMB.FILE  LM.LIFE.DEC
ZONE 41CDC4CDEE4EEDC4CCDC05DD4DCCC4CCC44444444444444444
NUMR 02447B6323B3442B69350434B3965B4530000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR     00001     LM.LIFE.CONTRACT                0000
ZONE 4444FFFFF44444DD4DCCC4CDDEDCCE4444444444444444FFFF
NUMR 0000000010000034B3965B365391330000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 1      DMG.OPT.TUMB.FILE -TUM SEP MAI STA ODD POR
ZONE F444441CDC4DDE4EEDC4CCDC06EED4ECD4DCC4EEC4DCC4DDD4
NUMR 1000001447B673B3442B69350034402570419023106440769 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR OFF                       TUM SEP MAI STA ODD POR OF
ZONE DCC4444444444444444444EED4ECD4DCC4EEC4DCC4DDD4DC
NUMR 6660000000000000000000003440257041902310644076906 6
    01...5...10...15...20...25...30...35...40...45...50
CHAR F                        DMG.SRC.TUMB.FILE   DMG.IDE
ZONE C4444444444444444444441CDC4EDC4EEDC4CCDC001CDC4CCC
NUMR 60000000000000000000001447B293B3442B6935002447B945
    01...5...10...15...20...25...30...35...40...45...50
CHAR F.TUMB.FILE   DMG.SKEY.TUMB.FILE  041-766-4361 DMG
ZONE C4EEDC4CCDC001CDC4EDCE4EEDC4CCDC00FFF6FFF6FFFF1CDC
NUMR 6B3442B6935002447B2258B3442B69350C0410766043611447
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR .MISSING.FORMS  EAST.BACK.OVERLAY                 00
ZONE 4DCEECDC4CDDDE05CCEE4CCCD4DECDDCE444444444444444FF
NUMR B4922957B66942045123B2132B6559318000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 000     EAST.FRONT.OVERLAY              00000
ZONE FFF00000CCEE4CDDDE4DECDDCE44444444444444FFFFF00000
NUMR 000000005123B69653B6559318000000000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR   NAME  Daniel J. Fairbanks AGE  65 SEX  Male POLI
ZONE F0DCDC01C898894D44C8898899A0CCC00FF0ECE00D8980DDDC
NUMR F451450341595301B06199215223175026532570441350D7639
    01...5...10...15...20...25...30...35...40...45...50
CHAR CY.NUMBER  1-344-2605 DMG.MERGESET.ID  1-344-2605
ZONE CE4DEDCCD00F6FFF6FFFF0CDC4DCDCCECE4CC00F6FFF6FFFF0
NUMR 38B5442590A1034402605F447B45975253B940A1034402605B
    01...5...10...15...20...25...30...35...40...45...50
CHAR POLICY.DATE  Jun 28 1983 SUM.INSURED  $ 85,000 PRE
ZONE DDDCCE4CCEC00DA94FF4FFFF0EED4CDEEDCC0054FF6FFF0DDC
NUMR 763938B41350B14502801983B244B952495408B085B000D795
    01...5...10...15...20...25...30...35...40...45...50
CHAR MIUM.CLASS  Standard BENEFIT.DESC  Flexible Whole
ZONE DCED4CDCEE00EA8988980CCDCCCE4CCEC01C98A88984E89984
NUMR 4944B3312208231541 94C2555693B4523036357923506863 50
    01...5...10...15...20...25...30...35...40...45...50
CHAR Life TERMINATION.PERIOD  Life ANNUAL.MAX.1  $235.2
ZONE D8881ECDDCDCECDD4DCDCDC00D8880CDDECD4DCE4F005FFF4F
NUMR 3965235949513965B759964043965C155413B417B107B235B2
```

```
   01...5...10...15...20...25...30...35...40...45...50
CHAR 5 ANNUAL.MAX.2  $280.00 PREMIUM.PERIOD  3 Years Th
ZONE F0CDDECD4DCE4F005FFF4FF0DDCDCED4DCDCDC02F4E889A4E8
NUMR 5C155413B417B207B280B00E7954944B759964023085192038
    01...5...10...15...20...25...30...35...40...45...50
CHAR ereafter to Jun 27, 2047 ANNUAL.PERIOD  $240.00 DM
ZONE 89888A894A94DA94FF64FFFF0CDDECD4DCDCDC005FFF4FF1CD
NUMR 595163590360145027B02047D155413B75996407B240B00044
    01...5...10...15...20...25...30...35...40...45...50
CHAR G.GCPY.INSURED  01 DMG.GCPY.FILE  00 DDNAME.INSURE
ZONE C4CCDE4CDEEDCC00FF0CDC4CCDE4CCDC00FF0CCDCDC4CDEEDC
NUMR 7B7378B95249540201D447B7378B69350200E445145B952495
    01...5...10...15...20...25...30...35...40...45...50
CHAR D  INSURED DDNAME.FILE  FILE DMG.FLST.9700.INSURED
ZONE C00CDEEDCC0CCDCDC4CCDC00CCDC1CDC4CDEE4FFFF4CDEEDCC
NUMR 4079524954B445145B69350469355447B6323B9700B9524954
    01...5...10...15...20...25...30...35...40...45...50
CHAR  yLM.LIFE.OVERLAY                  00001     LM.LIF
ZONE 0ADD4DCCC4DECDDCE44444444444444444FFFFF44444DD4DCC
NUMR 0834B3965B6559318000000000000000000000010000034B396
    01...5...10...15...20...25...30...35...40...45...50
CHAR E.DEC                        00001     LM.LIFE.OVERLA
ZONE C4CCC4444444444444444444FFFFF44444DD4DCCC4DECDDC
NUMR 5B45300000000000000000000000010000034B3965B655931
    01...5...10...15...20...25...30...35...40...45...50
CHAR Y                  00001     LM.LIFE.CONTRACT
ZONE E44444444444444444FFFFF44444DD4DCCC4CDDEDCCE444444
NUMR 8000000000000000000000010000034B3965B36539133000000
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR           00001      DMG.OPT.9700.INSURED {DUP SEP
ZONE 4444444444FFFFF444441CDC4DDE4FFFF4CDEEDCC0CCED4ECD
NUMR 000000000000001000004447B673B9700B9524954004470257
    01...5...10...15...20...25...30...35...40...45...50
CHAR  MAI STA ODD POR OVL US1 US2 US3 US4 US5 DUP SEP M
ZONE 4DCC4EEC4DCC4DDD4DED4EEF4EEF4EEF4EEF4EEF4CED4ECD4D
NUMR 041902310644076906530421042204230424042504470257 04
    01...5...10...15...20...25...30...35...40...45...50
CHAR AI STA ODD POR ON  US1 US2 US3 US4 US5 DUP SEP AUX
ZONE CC4EEC4DCC4DDD4DD44EEF4EEF4EEF4EEF4EEF4CED4ECD4CEE
NUMR 190231064407690650042104220423042404250447025701 47
    01...5...10...15...20...25...30...35...40...45...50
CHAR  STA ODD POR OVL                       DUP SEP AUX S
ZONE 4EEC4DCC4DDD4DED444444444444444444444CED4ECD4CEE4E
NUMR 023106440769065300000000000000000000004470257014702
    01...5...10...15...20...25...30...35...40...45...50
CHAR TA ODD POR ON                        DMG.SRC.9700.I
ZONE EC4DCC4DDD4DD444444444444444444444441CDC4EDC4FFFF4C
NUMR 3106440769065000000000000000000000004447B293B9700B9
    01...5...10...15...20...25...30...35...40...45...50
CHAR NSURED   DMG.IDEF.9700.INSURED   DMG.SKEY.9700.INS
ZONE DEEDCC001CDC4CCCC4FFFF4CDEEDCC001CDC4EDCE4FFFF4CDE
NUMR 524954005447B9456B9700B9524954005447B2258B9700B952
    01...5...10...15...20...25...30...35...40...45...50
CHAR URED  01Daniel J. Fairbanks 65 Male  1-344-2605Jun
ZONE EDCC03FFC898894D44C8898899A4FF4D89844F6FFF6FFFFDA9
NUMR 4954050141595301B06199215220650413500103440260514 5
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  28 1983  DMG.FLST.9700.FILE  LM.LIFE.DEC
ZONE 4FF4FFFF41CDC4CDEE4FFFF4CCDC05DD4DCCC4CCC444444444
NUMR 0280198302447B6323B9700B69350434B3965B453000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR             00001     LM.LIFE.CONTRACT
ZONE 444444444444FFFFF44444DD4DCCC4CDDEDCCE444444444444
NUMR 00000000000000001000034B3965B36539133000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR     00001      DMG.OPT.9700.FILE -SIM SEP SRC STA
ZONE 4444FFFFF444441CDC4DDE4FFFF4CCDC06ECD4ECD4EDC4EEC4
NUMR 0000000010000014 47B673B9700B6935002940257029302310
    01...5...10...15...20...25...30...35...40...45...50
CHAR ODD POR OFF                        SIM SEP SRC STA OD
ZONE DCC4DDD4DCC444444444444444444444ECD4ECD4EDC4EEC4DC
NUMR 64407690666000000000000000000000294025702930231064
    01...5...10...15...20...25...30...35...40...45...50
CHAR D POR OFF                          DMG.SRC.9700.FILE
ZONE C4DDD4DCC444444444444444444441CDC4EDC4FFFF4CCDC01
NUMR 407690666000000000000000000001447B293B9700B693500
    01...5...10...15...20...25...30...35...40...45...50
CHAR   FEED10  FEED30 DMG.IDEF.9700.FILE   DMG.SKEY.970
ZONE 00CCCCFF00CCCCFF1CDC4CCCC4FFFF4CCDC001CDC4EDCE4FFF
NUMR 0665541006655430 2447B9456B9700B6935002447B2258B970
    01...5...10...15...20...25...30...35...40...45...50
CHAR 0.FILE  021-344-2605 DMG.FLST.TUMB.INSURED  LM.LIF
ZONE F4CCDC00FFF6FFF6FFFF1CDC4CDEE4EEDC4CDEEDCC05DD4DCC
NUMR 0B69350C021034402605 5447B6323B3442B95249540434B396
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR E.DEC                        00001     LM.LIFE.CONTRA
ZONE C4CCC4444444444444444444FFFFF44444DD4DCCC4CDDEDC
NUMR 5B45300000000000000000000000010000034B3965B365391
    01...5...10...15...20...25...30...35...40...45...50
CHAR CT                  00001      DMG.OPT.TUMB.INSURED
ZONE CE4444444444444444FFFFF444441CDC4DDE4EEDC4CDEEDCC0
NUMR 330000000000000000000001000004447B673B3442B95249540
    01...5...10...15...20...25...30...35...40...45...50
CHAR -TUM SEP MAI STA ODD POR OFF                       T
ZONE 6EED4ECD4DCC4EEC4DCC4DDD4DCC44444444444444444444E
NUMR 03440257041902310644076906660000000000000000000003
    01...5...10...15...20...25...30...35...40...45...50
CHAR UM SEP MAI STA ODD POR OFF                        DM
ZONE ED4ECD4DCC4EEC4DCC4DDD4DCC444444444444444444441CD
NUMR 44025704190231064407690666000000000000000000000444
    01...5...10...15...20...25...30...35...40...45...50
CHAR G.SRC.TUMB.INSURED   DMG.IDEF.TUMB.INSURED   DMG.S
ZONE C4EDC4EEDC4CDEEDCC001CDC4CCCC4EEDC4CDEEDCC001CDC4E
NUMR 7B293B3442B9524954005447B9456B3442B9524954005447B2
    01...5...10...15...20...25...30...35...40...45...50
CHAR KEY.TUMB.INSURED  03Daniel J. Fairbanks  DMG.FLST.
ZONE DCE4EEDC4CDEEDCC01FFC898894D44C8898899A41CDC4CDEE4
NUMR 258B3442B952495406034159530 1B061992152202447B6323B
    01...5...10...15...20...25...30...35...40...45...50
CHAR TUMB.FILE  LM.LIFE.DEC                      00001
ZONE EEDC4CCDC05DD4DCCC4CCC4444444444444444444FFFFF44
NUMR 3442B69350434B3965B45300000000000000000000000100
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR    LM.LIFE.CONTRACT                  00001      DMG.
ZONE 444DD4DCCC4CDDEDCCE4444444444444444FFFFF444441CDC4
NUMR 00034B3965B365391330000000000000000000001000001447B
    01...5...10...15...20...25...30...35...40...45...50
CHAR OPT.TUMB.FILE -TUM SEP MAI STA ODD POR OFF
ZONE DDE4EEDC4CCDC06EED4ECD4DCC4EEC4DCC4DDD4DCC44444444
NUMR 673B3442B693500344025704190231064407690666000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR              TUM SEP MAI STA ODD POR OFF
ZONE 4444444444444EED4ECD4DCC4EEC4DCC4DDD4DCC4444444444
NUMR 00000000000034402570419023106440769066600000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR             DMG.SRC.TUMB.FILE   DMG.IDEF.TUMB.FILE
ZONE 444444444441CDC4EDC4EEDC4CCDC001CDC4CCCC4EEDC4CCDC
NUMR 000000000001447B293B3442B6935002447B9456B3442B6935
    01...5...10...15...20...25...30...35...40...45...50
CHAR    DMG.SKEY.TUMB.FILE  041-344-2605 DMG.MISSING.FO
ZONE 001CDC4EDCE4EEDC4CCDC00FFF6FFF6FFFF1CDC4DCEECDC4CD
NUMR 002447B2258B3442B69350C0410344026051447B4922957B66
    01...5...10...15...20...25...30...35...40...45...50
CHAR RMS  EAST.BACK.OVERLAY                00000     EAS
ZONE DDE05CCEE4CCCD4DECDDCE444444444444444FFFFF00000CCE
NUMR 942045123B2132B6559318000000000000000000000000512
    01...5...10...15...20...25...30...35...40...45...50
CHAR T.FRONT.OVERLAY              00000       NAME  Bry
ZONE E4CDDDE4DECDDCE44444444444444FFFFF00000F0DCDC01C9A
NUMR 3B69653B6559318000000000000000000000000F4514500298
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR an B. Masters AGE  55 SEX  Male POLICY.NUMBER  1-5
ZONE 894C44D8AA89A0CCC00FF0ECE00D8980DDDCCE4DEDCCD00F6F
NUMR 1502B0412359231750255325704413 5D763938B5442590A105
    01...5...10...15...20...25...30...35...40...45...50
CHAR 66-3423 DMG.MERGESET.ID  1-566-3423 POLICY.DATE  O
ZONE FF6FFFF0CDC4DCDCCECE4CC00F6FFF6FFFF0DDDCCE4CCEC00D
NUMR 6603423F447B45975253B940A1056603423B763938B41350B6
    01...5...10...15...20...25...30...35...40...45...50
CHAR ct  3 1983 SUM.INSURED  $ 55,000 PREMIUM.CLASS  St
ZONE 8A44F4FFFF0EED4CDEEDCC0054FF6FFF0DDCDCED4CDCEE00EA
NUMR 3300301983B244B952495408B055B000D7954944B331220823
    01...5...10...15...20...25...30...35...40...45...50
CHAR andard BENEFIT.DESC  Flexible Whole Life TERMINATI
ZONE 8988980CCDCCCE4CCEC01C98A88984E89984D8881ECDDCDCEC
NUMR 154194C2555693B4523036357923506863503965235949513 9
    01...5...10...15...20...25...30...35...40...45...50
CHAR ON.PERIOD  Life ANNUAL.MAX.1  $235.00 ANNUAL.MAX.2
ZONE DD4DCDCDC00D8880CDDECD4DCE4F005FFF4FF0CDDECD4DCE4F
NUMR 65B759964043965C155413B417B107B235B00C155413B417B2
    01...5...10...15...20...25...30...35...40...45...50
CHAR   $280.25 PREMIUM.PERIOD  3 Years Thereafter to Oc
ZONE 005FFF4FF0DDCDCED4DCDCDC02F4E889A4E889888A894A94D8
NUMR 07B280B25E7954944B75996402308519203859516359036063
    01...5...10...15...20...25...30...35...40...45...50
CHAR —  2, 2047 ANNUAL.PERIOD  $235.00 DMG.GCPY.INSURED
ZONE A44F64FFFF0CDDECD4DCDCDC005FFF4FF1CDC4CCDE4CDEEDCC
NUMR 3002B02047D155413B75996407B235B000447B7378B9524954
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR   02 DMG.GCPY.FILE  02 DDNAME.INSURED  INSURED DDN
ZONE 00FF0CDC4CCDE4CCDC00FF0CCDCDC4CDEEDCC00CDEEDCC0CCD
NUMR 0202D447B7378B69350202E445145B9524954079524954B445
    01...5...10...15...20...25...30...35...40...45...50
CHAR AME.FILE  FILE DMG.FLST.9700.INSURED yLM.LIFE.OVER
ZONE CDC4CCDC00CCDC1CDC4CDEE4FFFF4CDEEDCC0ADD4DCCC4DECD
NUMR 145B69350469355447B6323B9700B95249540834B3965B6559
    01...5...10...15...20...25...30...35...40...45...50
CHAR LAY                  00001     LM.LIFE.DEC
ZONE DCE44444444444444FFFFF44444DD4DCCC4CCC444444444
NUMR 3180000000000000000000010000034B3965B453000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR              00001     LM.LIFE.OVERLAY
ZONE 444444444444FFFFF44444DD4DCCC4DECDDCE4444444444444
NUMR 0000000000000000010000034B3965B6559318000000000000 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR     00001     LM.LIFE.CONTRACT                  0000
ZONE 4444FFFFF44444DD4DCCC4CDDEDCCE4444444444444444FFFF
NUMR 0000000010000034B3965B365391330000000000000000000 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR 1      DMG.OPT.9700.INSURED {DUP SEP MAI STA ODD P
ZONE F444441CDC4DDE4FFFF4CDEEDCC0CCED4ECD4DCC4EEC4DCC4D
NUMR 1000004447B673B9700B952495400447025704190231064407
    01...5...10...15...20...25...30...35...40...45...50
CHAR OR OVL US1 US2 US3 US4 US5 DUP SEP MAI STA ODD POR
ZONE DD4DED4EEF4EEF4EEF4EEF4EEF4CED4ECD4DCC4EEC4DCC4DDD
NUMR 690653042104220423042404250447025704190231064407 69
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  ON  US1 US2 US3 US4 US5 DUP SEP AUX STA ODD POR O
ZONE 4DD44EEF4EEF4EEF4EEF4EEF4CED4ECD4CEE4EEC4DCC4DDD4D
NUMR 0650042104220423042404250447025701470231064407690 6
    01...5...10...15...20...25...30...35...40...45...50
CHAR VL                      DUP SEP AUX STA ODD POR ON
ZONE ED4444444444444444444CED4ECD4CEE4EEC4DCC4DDD4DD4
NUMR 530000000000000000000004470257014702310644076906 50
    01...5...10...15...20...25...30...35...40...45...50
CHAR                       DMG.SRC.9700.INSURED   DMG.I
ZONE 4444444444444444444441CDC4EDC4FFFF4CDEEDCC001CDC4C
NUMR 0000000000000000000004447B293B9700B9524954005447B9
    01...5...10...15...20...25...30...35...40...45...50
CHAR DEF.9700.INSURED   DMG.SKEY.9700.INSURED  01Bryan
ZONE CCC4FFFF4CDEEDCC001CDC4EDCE4FFFF4CDEEDCC03FFC9A894
NUMR 456B9700B9524954005447B2258B9700B95249540501298150
    01...5...10...15...20...25...30...35...40...45...50
CHAR B. Masters    55 Male  1-566-3423Oct  3 1983  DMG.
ZONE C44D8AA89A4444FF4D89844F6FFF6FFFFD8A44F4FFFF41CDC4
NUMR 2B0412359200005504135001056603423633003019830 2447B
    01...5...10...15...20...25...30...35...40...45...50
CHAR FLST.9700.FILE  LM.LIFE.DEC                      00
ZONE CDEE4FFFF4CCDC05DD4DCCC4CCC4444444444444444444444FF
NUMR 6323B9700B69350434B3965B45300000000000000000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR 001     LM.LIFE.CONTRACT                 00001
ZONE FFF44444DD4DCCC4CDDEDCCE4444444444444444FFFFF44444
NUMR 0010000034B3965B365391330000000000000000000100000
```

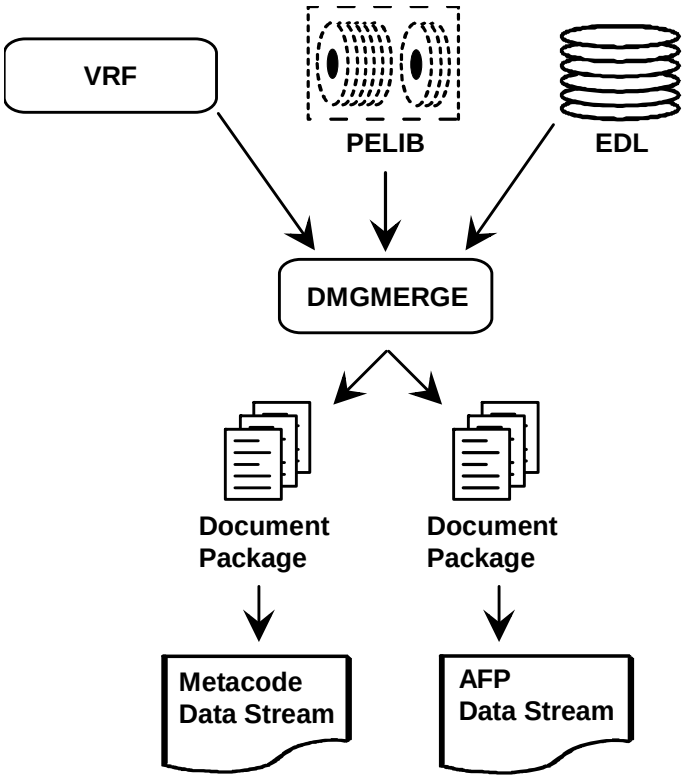
```
    01...5...10...15...20...25...30...35...40...45...50
CHAR  DMG.OPT.9700.FILE -SIM SEP SRC STA ODD POR OFF
ZONE 1CDC4DDE4FFFF4CCDC06ECD4ECD4EDC4EEC4DCC4DDD4DCC444
NUMR 1447B673B9700B693500294025702930231064407690666000
    01...5...10...15...20...25...30...35...40...45...50
CHAR                    SIM SEP SRC STA ODD POR OFF
ZONE 44444444444444444ECD4ECD4EDC4EEC4DCC4DDD4DCC44444
NUMR 0000000000000000002940257029302310644076906660000 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR                   DMG.SRC.9700.FILE    FEED10  FEED
ZONE 4444444444444441CDC4EDC4FFFF4CCDC0100CCCCFF00CCCC
NUMR 0000000000000001447B293B9700B69350006655410066554
    01...5...10...15...20...25...30...35...40...45...50
CHAR 30 DMG.IDEF.9700.FILE   DMG.SKEY.9700.FILE  021-56
ZONE FF1CDC4CCCC4FFFF4CCDC001CDC4EDCE4FFFF4CCDC00FFF6FF
NUMR 302447B9456B9700B6935002447B2258B9700B69350C021056
    01...5...10...15...20...25...30...35...40...45...50
CHAR 6-3423 DMG.FLST.TUMB.INSURED  LM.LIFE.DEC
ZONE F6FFFF1CDC4CDEE4EEDC4CDEEDCC05DD4DCCC4CCC444444444
NUMR 6034235447B6323B3442B95249540434B3965B453000000000
    01...5...10...15...20...25...30...35...40...45...50
CHAR             00001     LM.LIFE.CONTRACT
ZONE 444444444444FFFFF44444DD4DCCC4CDDEDCCE444444444444
NUMR 0000000000000000010000034B3965B365391330000000000 0
    01...5...10...15...20...25...30...35...40...45...50
CHAR     00001      DMG.OPT.TUMB.INSURED -TUM SEP MAI S
ZONE 4444FFFFF444441CDC4DDE4EEDC4CDEEDCC06EED4ECD4DCC4E
NUMR 0000000010000044 47B673B3442B952495400344025704190 2
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR TA ODD POR OFF                        TUM SEP MAI STA
ZONE EC4DCC4DDD4DCC4444444444444444444444EED4ECD4DCC4EEC
NUMR 3106440769066600000000000000000000000344025704190231
    01...5...10...15...20...25...30...35...40...45...50
CHAR  ODD POR OFF                         DMG.SRC.TUMB.INS
ZONE 4DCC4DDD4DCC4444444444444444444441CDC4EDC4EEDC4CDE
NUMR 0644076906660000000000000000000004447B293B3442B952
    01...5...10...15...20...25...30...35...40...45...50
CHAR URED   DMG.IDEF.TUMB.INSURED   DMG.SKEY.TUMB.INSUR
ZONE EDCC001CDC4CCCC4EEDC4CDEEDCC001CDC4EDCE4EEDC4CDEED
NUMR 4954005447B9456B3442B9524954005447B2258B3442B95249
    01...5...10...15...20...25...30...35...40...45...50
CHAR ED  03Bryan B. Masters     DMG.FLST.TUMB.FILE  LM.
ZONE CC01FFC9A894C44D8AA89A44441CDC4CDEE4EEDC4CCDC05DD4
NUMR 5406032981502B0412359200002447B6323B3442B69350434B
    01...5...10...15...20...25...30...35...40...45...50
CHAR LIFE.DEC                      00001     LM.LIFE.CON
ZONE DCCC4CCC4444444444444444444FFFFF44444DD4DCCC4CDD
NUMR 3965B453000000000000000000000010000034B3965B365
    01...5...10...15...20...25...30...35...40...45...50
CHAR TRACT                 00001      DMG.OPT.TUMB.FILE
ZONE EDCCE4444444444444444FFFFF444441CDC4DDE4EEDC4CCDC0
NUMR 391330000000000000000000010000014 7B673B3442B69350
    01...5...10...15...20...25...30...35...40...45...50
CHAR -TUM SEP MAI STA ODD POR OFF                       T
ZONE 6EED4ECD4DCC4EEC4DCC4DDD4DCC4444444444444444444444E
NUMR 0344025704190231064407690666000000000000000000003
```

```
    01...5...10...15...20...25...30...35...40...45...50
CHAR UM SEP MAI STA ODD POR OFF                        DM
ZONE ED4ECD4DCC4EEC4DCC4DDD4DCC4444444444444444444441CD
NUMR 4402570419023106440769066600000000000000000000144
    01...5...10...15...20...25...30...35...40...45...50
CHAR G.SRC.TUMB.FILE   DMG.IDEF.TUMB.FILE   DMG.SKEY.TU
ZONE C4EDC4EEDC4CCDC001CDC4CCCC4EEDC4CCDC001CDC4EDCE4EE
NUMR 7B293B3442B6935002447B9456B3442B6935002447B2258B34
    01...5...10...15...20...25...30...35...40...45...50
CHAR MB.FILE  041-566-3423
ZONE DC4CCDC00FFF6FFF6FFFFF4444444444444444444444444444
NUMR 42B69350C041056603423F0000000000000000000000000000
```

The DMGMERGE Program

The DMGMERGE program performs the final steps necessary to produce finished output. DMGMERGE performs these functions based upon information from the Variable Replacement File (VRF), the Publishing Environment Library (PELIB), the Electronic Document Library (EDL), and DMGMERGE parameters.



The DMGMERGE Processing Steps

For each Merge Set, DMGMERGE

(1) Reads the Merge Set data from the VRF.

(2) Reads the forms from the EDL (or EDLs) listed in the DMG.FLST.*Groupname* tag in the VRF.

(3) Creates output Document Packages for each recipient based on the order and presence of the MERGE GROUP= parameter, and the number of Document Package copies based upon the value of the DMG.GCPY.*Groupname* Reserved Tag value or DMGMERGE parameter COPIES=.

(4) Merges variable data into the appropriate Boilerplate Spaces on each form.

(5) Accumulates audit statistics.

(6) Directs the completed Document Package to the appropriate output destination, determined either by the DMGMERGE parameter DDNAME=, or by the reserved tag DMG.DD.*Groupname*.

(7) If the Merge Set is in error Documerge will determine output destination based on the DMGMERGE parameter ERRDDN=, or the DMG.ERRDD.*Groupname*.

Documerge specifies only one VRF per execution. In MVS, multiple VRF's may be concatenated in the JCL, and would be processed as if they were one large VRF.

DMGMERGE sample JCL

```
//DMGMERGE  ** put your job card here **
//*
//* ********************************************************************
//* **                                                                **
//* **  This jcl file runs the installation demo of Documerge         **
//* **  for an IBM AFP printer (3820/3827/3825/3838/3835/3900/4000)   **
//* **                                                                **
//* ********************************************************************
//*
//JOBLID   DD DISP=SHR,DSN=documerg.v03r02.loadlib
//*
//DMGSTEP EXEC PGM=DMGMERGE,
//            REGION=4096K,
//            PARM='PARMFILE=PARMFILE'
//PARMFILE DD *
FORMSBUFF=1000K
VLMCONTROL=KEEP
WORKBUFF=2000K
/*
//SYSUDUMP DD SYSOUT=*
//ISIWTL   DD SYSOUT=*
//ISIWTO   DD SYSOUT=*
//MESSAGE  DD SYSOUT=*,OUTLIM=1000,
//            DCB=(RECFM=FBM,LRECL=150,BLKSIZE=1500)
//WRKFIL   DD DSN=&&WRKFIL,
//            UNIT=sysda,
//            DISP=(NEW,DELETE),
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                    HALF TRACK
//PEDEF    DD DSN=documerg.v03r02.pelib,DISP=SHR
//VLM2LIB  DD DSN=documerg.v03r02.edl,
//            DISP=SHR
//DMGVRF1  DD DSN=documerg.v03r02.vrf,
//            DISP=SHR
//DMGVRFa  DD DSN=documerg.v03r02.vrfa,
//            DISP=SHR
//*
//*  DOCUMERGE OUTPUT GROUPS
//INSURED  DD SYSOUT=*,
//            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//FILE     DD SYSOUT=*,
//            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//*
//*  DOCUMERGE GROUPS-IN-ERROR
//INSERR   DD SYSOUT=*,
//            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//FILERR   DD SYSOUT=*,
//            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//SYSIN    DD  *
MERGE GROUP=INSURED -
      DDNAME=INSURED -
      MERGEDEF=3820  -
      PORTSEP=.0I    -
      CHAIN=AFP      -
      ERRDDN=INSERR  -
      BANNER=LM.LIFE.BANNER(1) -
      TRAILER=LM.LIFE.TRAILER(1)
MERGE GROUP=FILE    -
      DDNAME=FILE    -
      MERGEDEF=3820  -
      PORTSEP=.0I    -
      CHAIN=AFP      -
      ERRDDN=FILERR  -
      BANNER=LM.LIFE.BANNER(1) -
      TRAILER=LM.LIFE.TRAILER(1)
/*
//
```

DMGMERGE EXEC Parameters

The following parameters offer options for DMGMERGE processing. You can code one or more of these parameters in the PARM parameter of the VDR EXEC statement.

Coding PARM Parameters in a PARMFILE

Documerge offers the option of defining EXEC parameters in a separate, external file called by the keyword PARMFILE. The syntax is

```
PARMFILE=ddname
```

Where `ddname` is the name you give to your external file. The external file must have the following attributes:

BLKSIZEuser selectable but must be a multiple of 80

LRECL80

RECFMFB

You can use any input medium (e.g. cards or disk). The parameter file is limited to 20 records maximum.

You can use the IBM utility IEBGENER to create this file, copying cards to disk.

You can include parameters in your JCL as well as the parameter file. If a parameter is coded in both the JCL and the parmfile, the JCL takes precedence.

All 80 columns of each input record are usable. A single parameter (e.g. FORMSBUFF=10K) cannot span multiple records. However, two or more parameters may occupy the same record, separated by one or more blanks.

If sequence numbers are present (usually in columns 73-80), then the column before the sequence numbers (usually 72), must be a blank or a comma. Otherwise, Documerge might interpret the sequence numbers as data belonging to a parameter.

Since parameters can begin in column one, and since Documerge currently uses less than 20 parameters, probably the easiest method of coding a parameter file is to include each parameter on a separate record, beginning in column one.

In case of duplicate parameters, the parameter file uses the first value given. If the same parameter appears in both the parameter file and in the EXEC PARAMETER JCL, DMGMERGE uses the value from the EXEC PARAMETER.

Sample parameter file

```
//MYPARMS   DD  *
FORMSBUFF=300K
WORKBUFF=200K
VLMCONTROL=KEEP
VLMACCESS=RO
EDLNAMES=(TESTEDL,PRODEDL)
/*
```

EXEC Parameters

The following parameters can be coded either in an optional parameter file or in the DMGMERGE JCL.

Parameter	Value
DATABUFF=	Defines the amount of storage for the data in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the DATABUFF= value: ▪ If the DMGVRFA= value is N, DATABUFF= defines the exact amount of storage. The default value is 500 K. ▪ If the DMGVRFA= value is not N, DATABUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tag data combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is not N, omit the DATABUFF= parameter. **NOTE:** This parameter is optional. Use it to limit the storage defined by the DMGVRFA file. Do not use this parameter if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.
DMGVRFA=	Indicates whether DMGMERGE uses a DMGVRFA file. (See "" on page 252 for more information.) When you code a sorted VRF as input to DMGMERGE, use the same DMGVRFA file for DMGMERGE that you used for DMGSORT. **N** or **NO** DMGMERGE does not use a DMGVRFA file. **Y** or **YES** DMGMERGE uses a DMGVRFA file. The default value is YES. **NOTE:** If you specify any value other than N, DMGMERGE uses a DMGVRFA file. (DMGMERGE reads only the first character of this value.) Therefore, omit the DMGVRFA= parameter if you want the DMGVRFA file.
EDLNAMES=	Supplies the name of an appended list file containing from one to nine EDL DDnames. Documerge treats the EDL names in the list as a logical EDL (that is, as one big EDL). **NOTE:** If an EDL name in the list is preceded by a slash (/), DMGMERGE will not place any form from that EDL into the FORMSBUFF area, even if the FORMSBUFF parameter value is more than enough for all the forms that are buffered. The default is to assume one EDL name. See "Multiple EDLs" on page 117 and "DMGDELET" on page 458 for more information about some of the uses of multiple EDLs. Format: `EDLNAMES=(name1,/name2,...,name9)` **CAUTION:** Due to space limitations, the SYSOUT listing for a Documerge run will typically only list the first three multiple (concatenated) EDLs. Normally, this parameter should be the same as the one in the corresponding VDR EXEC parameter EDLNAMES (see "EDLNAMES=" on page 194 for more information).

Parameter	Value
EDTBDT=	This parameter affects AFP output. The format for this parameter is: **EDTBDT=*value*** Valid values are: **Y** *or* **YES** Normally DMGMERGE will issue an IMM to force a new sheet of paper. A value of **Y** or **YES** causes DMGMERGE to issue an EDT/BDT sequence instead. Use this for post processors that do not recognize an IMM as forcing a new sheet of paper. **N** *or* **NO** This is the default. It causes DMGMERGE to issue an IMM to force a new sheet of paper.
FORMSBUFF=	Defines the amount of storage available to DMGMERGE for buffering the EDL members. The FORMSBUFF should be large enough to hold all EDL members used. The default is 100k. **NOTE:** If an EDL name defined in an EDLNAMES parameter (see "EDLNAMES=" on page 375) starts with a slash (/), DMGMERGE will not place any form from that EDL into the FORMSBUFF area, even if the FORMSBUFF parameter value is more than enough for all the forms that are buffered. DMGMERGE will issue a DMGMRG180I message reporting the number of forms that did not fit in FORMSBUFF, including those forms in an EDL which is named with a starting slash and specified in an EDLNAMES parameter.
MSGCASE=	Indicates whether Documerge programs generate messages in mixed-case or upper-case letters. This parameter applies to messages generated by all Documerge programs whose names begin with DMG. These messages are written to the MESSAGE file or to the error DD. Valid values are: **M** *or* **MIXED** Mixed-case (the default value) **U** *or* **UPPER** Upper-case
NUMAREAS=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. The memory allocated for LM/MM is the number you specify multiplied by the block size of the WRKFIL. (See "WRKFIL" on page 378.) Specify a number from **2** to **2048**. You can use this parameter instead of the WORKBUFF= parameter. If you use both parameters, DMGMERGE uses WORKBUFF= and ignores NUMAREAS=. If you do not use either parameter, DMGMERGE uses a NUMAREAS= default value of **8**. (Therefore, the LM/MM allocation is 8 multiplied by the WRKFIL block size.) **NOTE:** As of version 3.1, DMGMERGE no longer uses the STACKBUFF and STCK2BUFF parameters. However, VLAMMAINT still uses the STACKBUFF parameter.

Parameter	Value
TAGBUFF=	Optional. Defines the amount of storage for the tags in a Merge Set. The value of the DMGVRFA= parameter determines how DMGMERGE uses the TAGBUFF= value: If the DMGVRFA= value is N, TAGBUFF= defines the exact amount of storage. The default value is 500 K. If the DMGVRFA= value is not N, TAGBUFF= defines the limit for storage; DMGMERGE can use less than this amount, but it cannot use more. If this amount is less than the size of all tags combined, DMGMERGE runs slower. If you want to use the amount DMGMERGE allocates when the DMGVRFA= parameter is not N, omit the TAGBUFF= parameter. **NOTE:** You can use TAGBUFF= to limit the storage defined by the DMGVRFA file. Do not use TAGBUFF= if you want DMGMERGE to use the exact storage defined by the DMGVRFA file.
VLMACCESS=	Specifies how VLAM accesses the VLAM Libraries for a job step. **RO** *or* **READONLY** Read-only access. Access dates are NOT written to the VLAM Libraries. If a security system is installed, a user with READ only authority may submit this step for execution. **RW** *or* **READWRITE** Read-write access. Access dates are written to the VLAM Library. If a security system is installed, a user must have UPDATE authority to submit this step for execution. READWRITE is the default and is more efficient than READONLY.
VLMCONTROL=	Specifies how many times DMGMERGE will access VLAM before VLAM releases exclusive control (lock) of the EDL allowing other users to access it. While maintaining exclusive control, VLAM holds the alphabetic index in memory, allowing for more efficient processing of members. **KEEP** Prevents any updates to the EDL during DMGMERGE execution (places other jobs attempting to update the EDL in the wait state). If another job is already updating the EDL, the job containing KEEP will also be placed in the wait state. **200** Allows that many accesses to the EDL before giving up control. When all requests from other users have been satisfied, VLAM will lock the EDL for another 200 accesses by DMGMERGE. The default is one (1).
WORKBUFF=	Allocates memory for the List Manager/Memory Manager (LM/MM), an internal program which manages storage for many Documerge programs. For a discussion of performance considerations related to WORKBUFF, see "WORKBUFF" on page 446. If the memory allocation is smaller than the amount of data, LM/MM uses the WRKFIL for secondary space. (See "WRKFIL" on page 378.) The WORKBUFF= value must be at least twice the value of the WRKFIL block size. The WORKBUFF= value must not exceed the WRKFIL block size multiplied by 2048, or 1,073,741,823. Valid values are: **nnnK** nnn units of 1024 bytes **nnnM** nnn units of 1,048,576 bytes (1024 — 1024) You can use this parameter instead of the NUMAREAS= parameter. If you use both parameters, DMGMERGE uses WORKBUFF and ignores NUMAREAS. If you do not use either parameter, DMGMERGE uses a NUMAREAS value of eight as the default.

DMGMERGE Files

- **MESSAGE**

  The file to which all SYSIN echos and DMGMERGE messages are written.

- ISIWTL

  Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTL is intended for programmer and application user error messages. Using Oracle, any program that has written to ISIWTL has encountered a fatal error.

- **ISIWTO**

  Documerge uses this message DD when a subprogram needs to issue an error message. ISIWTO is intended for computer operator messages. Using Oracle standards, a program that has written to ISIWTO has encountered a fatal error.

- **WRKFIL**

  Indicates a sequential work file for DMGMERGE. WRKFIL is used when the parameter WORKBUFF (or NUMAREAS) space is full. This is a temporary file, created for only one DMGMERGE execution. It is not passed to any other job step.

NOTE

As of version 3.1, DMGMERGE no longer uses the ISIWORK and ISIWRK2 files. However, VLAMMAINT still uses the ISIWORK file.

- FONTMETA

  Indicates the name of the Metacode font library. *This must be the same font library as used during DMGMETP Normalization*. This font library is only accessed during Tumble or Imposition processing in a Metacode environment.

- VLM2LIB

  Indicates the VLAM 2 EDL library name. This is the default if there is no EDLNAMES parameter present.

- DMGVRF1

  Indicates the VRF file name.

- DMGVRFA

  Indicates the VRF Allocation file name.

- **PEDEF**

  Indicates the PELIB file name.

- FORMDEF

  Mandatory for AFP printing if the GLOBAL parameter VERIFYCG= is YES or Y. This is the same name as the FORMDEF in your PSF JCL.

- OUTPUT

Indicates the file name or SYSOUT used for the printable Document Package. Requirements of the physical file characteristics are based upon the type of printer. This file's recommended physical characteristics (for AFP) are:

- Record format = VBM
- Record length = 8205; for Metacode, 155 (the Metacode LRECL cannot exceed 256)
- Blocksize = 8209; for Metacode, 3000

IMPORTANT!

The LRECL for a DMGMERGE production output (clean) file needs to be

- No larger than (i.e., both need to be the same) the LRECL of the corresponding error file.
  DMGMERGE will switch to the error file when the first error is found for a package, but it is unpredictable at what point, if any, DMGMERGE finds the first error. It could be found after DMGMERGE manipulates a record using the clean file. If it subsequently switches to the error file, and the error file has an LRECL less than the clean file, then the error file won't write successfully (because the record is too large).
- At least as large as the largest record in any EDL form that will be written to the file. For example, consider an EDL member that contains Metacode records, and the largest record is 145 bytes long. If the output file is defined for variable length records, then the DMGMERGE output file required to print the EDL member would have to have an LRECL of 149 — 145 bytes plus an additional 4 bytes for a variable length RDW (record descriptor word).

Documerge usually installs automatically with the maximum LRECL for the type of printer, which is typically is 155 for Metacode and 8905 for AFP. However, to ensure the correct capacity for your output file records you can do the following:

1. Determine which of your forms has the longest record in bytes.
2. Print that form only, using VLMMAINT DUMP, IEBGENER, etc., before loading the form (member and chain) to the EDL).
3. Ensure that your DMGMERGE output file LRECL is at least as large as the number of bytes that were required when you printed the form stand-alone. For Metacode processing, the LRECL cannot exceed 256 bytes.

- OUTPUTE

  Indicates the file name or SYSOUT to be used for the printable Document Package with errors. Requirements or the physical file characteristics are based upon the type of printer. This file's recommended physical characteristics (for AFP) are:

  - Record format = VBM
  - Record length = 8205; for Metacode, 155 (the Metacode LRECL cannot exceed 256)
  - Blocksize = 8209; for Metacode, 3000

- **SYSIN**

  Parameters available to DMGMERGE. An unlimited number of Merge commands can be used with DMGMERGE.

IMPORTANT!

DCB information is required for all DMGMERGE output files. This requirement applies even to output files for which you specify DD DUMMY.

Omission of this DCB information can result in error messages or ABENDs.

DMGMERGE Commands

The COMMONFONTS Command

DMGMERGE can dynamically convert Metacode forms to match a new Common Font List. The **COMMONFONTS** command tells DMGMERGE which Common Font List to use.

The COMMONFONTS command is optional. It can be used only once in a DMGMERGE execution. If used, it must precede any MERGE command in the DMGMERGE SYSIN.

When you use COMMONFONTS, DMGMERGE makes the following changes to each Metacode form:

- Replaces the font names in the DJDE records with the font names in the new Common Font List
- Changes the ISIFONTDEF comment record
- Updates any Metacode font references

Sample COMMONFONTS Command

The following figure shows a COMMONFONTS coding sample:

```
COMMONFONTS FONTS=(FORMSX H4106E H4108E H4110E H4112E HB110E -
HB112E HB114E LG110E P0812C H6212E 4635AL P08OAA CB124E -
HB111E H4512E H4114E LG112E H4108D -
H6216E LG212E HB108E H4109E P0612C LG210E H6214E)
```

NOTE

The COMMONFONTS command does not change the forms in the EDL. Each time DMGMERGE reads a form from the EDL, it converts that form dynamically in memory. Therefore, the Common Font List generated by the COMMONFONTS command applies only to one DMGMERGE execution. (See "FORMSBUFF=" on page 376 and "FORMSBUFF" on page 448 for more information about dynamic processing of EDL members by DMGMERGE.)

The Common Font Update (DMGCMFN) utility changes the forms in the EDL to a new Common Font List by updating specified chains. These chains remain stored in the EDL; they are not changed dynamically. Therefore, you can use the DMGCMFN Common Font List for other DMGMERGE executions. Refer to "The Common Font Update (DMGCMFN) Utility" on page 119.

Neither the DMGCMFN utility nor the COMMONFONTS command support Dynacomp forms. You must reprocess the Dynacomp source files with DCOPCOMP and specify the new Common Font List in the ENVDEF.

COMMONFONTS Command Requirements

The COMMONFONTS command requires three valid PEDEFs:

- Output Environment Definition (ENVDEF)
- Printer Definition (PRINTDEF)
- Font Group Definition (FGRPDEF)

The ENVDEF contains the Common Font List, in the FONTS= parameter. This Common Font List applies to all Groups in the Merge Set when the COMMONFONTS command is used. The ENVDEF also indicates a PRINTDEF that contains a FGRPDEF. DMGMERGE needs to know the FGRPDEF, because it names a Group of fonts that are valid for the printer.

Boilerplate Index Packet Requirements

For conversion to a new Common Font List, a form's Boilerplate Index Packet must meet the following requirements:

- The ISIFONTDEF record cannot contain fonts that are not in the new Common Font List.
- The DJDE record cannot contain a font list that uses a PDE on the printer (a global PDE).
- If the DJDE has a FORMS= parameter (which calls an FRM), and the form's font list does not already match the new Common Font List, the following changes must be made:

  1. Renormalize the form.
  2. Reload the form to the EDL.
  3. Recreate the FRM.

     For example, you could use TEMMETP to renormalize and create a template.
  4. Reload the FRM to the printer.

NOTE

If the form's font list already matches the new Common Font List, these changes are not necessary.

If any of these requirements is not met, the form's font list is not changed, and the form cannot be concatenated. Also, depending on the ERRMSG= parameter, DMGMERGE generates an error message and sends the Merge Set to the error DD.

NOTE

Oracle recommends that you print a form with the new Common Font List on each printer that receives Documerge output. This test ensures that the Common Font List is valid for all your printers.

COMMONFONTS Parameters

Parameter	Value
ENVDEF=	The name of the ENVDEF that contains the new Common Font List. Code ENVDEF= if you want DMGMERGE to use a Common Font List that exists in an ENVDEF. You must code either ENVDEF= or FONTS=. You cannot code them both. Specify 1 to 6 characters, not including the DE prefix assigned by the PEDEF utility DPLDUTL.
ERRMSG=	Indicates whether DMGMERGE processes errors it finds while converting forms. Optional. Valid with ENVDEF= or FONTS=. Valid values are: **Y** *or* **YES** — DMGMERGE generates error messages and sends the Merge Set to the error DD. YES is the default value. **N** *or* **NO** — DMGMERGE does not generate error messages or send the Merge Set to the error DD.
FGRPDEF=	The 1- to 6-character name of the FGRPDEF that defines a Group of valid fonts for the printer. Optional. The default value is the FGRPDEF in the PRINTDEF that is contained in the ENVDEF. If the ENVDEF does not contain a PRINTDEF that specifies a FGRPDEF, you must code the FGRPDEF= parameter. If you code the FONTS= parameter, the FGRPDEF= parameter is not necessary and is ignored if coded.

FONTS=	The new Common Font List.
	Code FONTS= if you want to create the Common Font List independent of an ENVDEF. You must code either FONTS= or ENVDEF=. You cannot code them both.
	Code the names of the Metacode fonts you want to include in the Common Font List. The font names must be:
	▪ enclosed in parentheses
	▪ separated by blanks (do not separate font names with commas)
	▪ no longer than 6 characters
	The FORMSX font name should be included, and can be coded anywhere within the parentheses. You can split the font names between lines by using the hyphen as a continuation character.
	For example:
	`COMMONFONTS FONTS=(PR111E PR211E -`
	`UN104E FORMSX -`
	`P07TDC)`
	Documerge does not limit the number of fonts you can code in FONTS=. However, the Xerox maximum is 127. And, depending on font size and available storage, printers often limit the number of fonts to less than 100.

The DDRENAME Command

Renames a DDname to one or more other DDnames. Documerge replaces the *oldname* with the *newname(s)*, as if they were the one(s) originally specified.

You can use DDRENAME to rename

- The single DDname used by a VDR to quickly implement *routing-by-sheets* without changing the VDR program code. The VDR still requests a single DD, but you rename this DD to one or more DDnames defined to change when the output exceeds the specified number of sheets.

  For details about routing-by-sheets, see "The FILEDEF Command and Routing-by-Sheets (MAXSHEETS)" on page 420.

- The single DDname used by a VDR to quickly implement *output segmentation* without changing the VDR program code. The VDR still requests a single DD, but you rename this DD to one or more DDnames defined to change when the number and the type of output units (SHEETS, PAGES, PACKAGES, RECORDS, or BYTES) exceed the limit specified with the SEGMENT parameter.

  For details about output segmentation, see "" on page 424.

- The DDnames used by VDRs that create the DMG.DD.*Groupname* or DMG.ERDD.*Groupname* tags. The VDR itself can continue to request the old DDname, but the user can tell Documerge to rename the DD to a different name.

- Several old DDnames to the same new DDname, so that several unique DDnames can be combined into one big DD file for testing or production.

DDRENAME Command Format

```
DDRENAME OLDDD=oldname NEWDD=newname
```

oldname	The current DDname specified in a DDNAME or ERRDDN parameter in a MERGE command, or specified in a DMG.DD.*Groupname* or DMG.ERDD.*Groupname* tag in the VRF. `oldname` must be a single DDname (not a list) of up to 8 characters.
newname	A list of DDnames enclosed in parentheses, or `newname` can be a single DDname (parentheses optional). Any DDnames listed in the NEWDD must have a DD JCL statement (MVS) or a FILEDEF command (VSE). For every DD except the last in a list, there needs to be a FILEDEF. Without a FILEDEF, the user cannot specify any selection criteria, and the first DD will be selected, and the subsequent DDs in the list will never be used.

DDRENAME Coding Rules

Here are some rules for coding DDRENAME:

- You cannot rename a DDname coded for the NEWDD parameter (DDRENAME cannot be nested or used recursively).

- The new DDname coded must point to a file defined with a DD JCL statement. However, the NEWDD can be a list of DDnames, one of which happens to be the OLDDDname.

- You can code more than one DDRENAME command. You must have one DDRENAME for each OLDDD you want to rename. There is no limit to the number of DDRENAME commands you can code.

- If two or more DDRENAME commands specify the same OLDDDname, Documerge uses the first one, and ignores the remaining commands without warning or error.

- Documerge globally implements the DDRENAME command. You cannot selectively rename by Group or Merge Set.
- The OLDDD specifies a name in a DDNAME or ERRDDN parameter of a MERGE command, or in a DMG.DD.*Groupname* or DMG.ERDD.*Groupname* reserved tag. If any of these contain a list of DDnames, you can code individual DDRENAME commands for each name in the list.
- To easily implement segmentation for VDRs that generate independent routing tags (DMG.DD.*Groupname*), an OLDDD or NEWDD can be a variable DDname.
- All DDRENAME commands must precede any MERGE commands. The correct sequence of DMGMERGE commands is

  1. GLOBAL
  2. COMMONFONTS
  3. DDRENAME
  4. FILEDEF
  5. MERGE

- The DDRENAME command does not affect any DMGMERGE system files such as MESSAGE, DMGVRF1, DMGVRFA, SYSIN, VLM2LIB, or WRKFIL, but you cannot specify a NEWDD that is the same as a Documerge reserved name or system file.

  If a VRF contains a DMG.DD.*groupname* tag or a DMG.ERDD.*groupname* tag that specifies a DMGMERGE system output file (i.e., a *reserved* DD name), the DDRENAME command can be used to change the name to a non-reserved (non-system) user name.

  For example, assume a Documerge 2.1 application had a DMG.DD.GROUP1 tag with a value of **WRKFIL**. WRKFIL was not a reserved name for Documerge 2.1, but is a reserved name for 3.x versions. Instead of changing the VDR application to generate a different value for DMG.DD.GROUP1, you could use Documerge 3.2 to code a DDRENAME command as follows:

  ```
  DDRENAME OLDDD=WRKFIL NEWDD=G1OUT
  ```

  You could use any NEWDD name you want, as long as it is not a DMGMERGE system file or reserved name. The DDRENAME example does not change DMGMERGE's use of its WRKFIL system file.

DDRENAME Command Examples

The following two DMGMERGE SYSINs will cause the same processing:

```
SYSIN 1:
FILEDEF NAME=INSFILE1 MAXSHEETS=100
FILEDEF NAME=INSFILE2
MERGE GROUP=INSURED -
                DDNAME=(INSFILE1 INSFILE2) -
                rest of MERGE parameters here
SYSIN 2:
FILEDEF NAME=INSFILE1 MAXSHEETS=100
FILEDEF NAME=INSFILE2
DDRENAME OLDDD=INSFILE NEWDD=(INSFILE1 INSFILE2)
MERGE GROUP=INSURED -
                DDNAME=INSFILE -
                rest of MERGE parameters here
```

The GLOBAL Command

The GLOBAL command allows the user to specify several Documerge run-time options. You must specify all GLOBAL commands before you specify any MERGE commands.

You can code GLOBAL more than once in a DMGMERGE run, but each parameter can appear only once. In other words, the following two examples produce equivalent results:

(1)
```
GLOBAL   ERRVRF=ERRORVRF
GLOBAL    STATSFILE=AUDIT
```

(2)
```
GLOBAL   ERRVRF=ERRORVRF -
STATSFILE=AUDIT
```

GLOBAL is an optional command.

GLOBAL Parameters

Parameter	Value
ALLERROR=	Selects a processing option that routes ALL Groups to the error DD if ANY Group has an error. ALLERROR= is optional. ALLERROR accepts the parameters **YES**, **Y**, **NO**, and **N**. The DEFAULT is NO (N). When ALLERROR=NO (or N), or when the option is not coded, then for each Merge Set, each Group is processed independently. When ALLERROR=YES (or Y), then if ANY Group contains an error, each Group in every Merge Set is routed to its appropriate error stack. Message DMGMRG399W is generated for any Group that had no errors. **NOTE:** Specifying ALLERROR=YES may require considerable additional WORKBUFF space (all Groups will be held in the WORKBUFF until the entire Merge Set is processed). In addition, specifying ALLERROR=YES may cause Documerge to run slower, especially if insufficient WORKBUFF space is specified.
BVL=( )	(Optional) specifies a **Back Overlay** for all the even pages of all Document Packages processed in a DMGMERGE run. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.

Parameter	Value
BVLDEFAULT=	Resets the default value for the WHEN parameter of the Rulebase Library Structure Rule BOVERLAY=. The *when* parameter can have one of three values: **ALWAYS** The Overlay prints on every page in the specified DTN. **NONBLANK** The Overlay prints only if the page is non-blank; that is, if the page contains a normal (non-Overlay) form. **ONLYBLANK** The Overlay prints only on blank pages; that is, pages that have no other form printed on them. For example, this option could be used, to automatically generate pages that say "This page intentionally left blank". The BOVERLAY= default value is ALWAYS.
CDFROM=	For Metacode only. The Font Translation Table (CODEDEF) used by DMGMERGE to translate Replacement Characters and variable data from EBCDIC to ASCII. Valid values are: **FORM** DMGMERGE uses the CODEDEF from the form's Boilerplate Index Packet. FORM is the default value Refer to "Boilerplate Index Packet Column Definitions: DMGMETP" on page 61 for more information. This CODEDEF is listed in columns 72-79 of the Index Begin Record as shown below:

```
     ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00001  %%%DMG2NDXBEG%%% XXXX #### #### #### #### #### XXXX CR ##/##/## ##.## XXXXXXXX
```

	FGRPDEF DMGMERGE uses multiple CODEDEFs from the Font Group Definition (FGRPDEF). The FGRPDEF contains one or more subcommands called Font Definitions (FONTDEFs). Each FONTDEF contains parameters that describe one member of a non-IBM font family. The FGRPDEF must contain a FONTDEF for each font-family member that is specified in a form. Each FONTDEF in the FGRPDEF can have a different value for its CODEDEF= parameter. When you code CDFROM=**FGRPDEF**, DMGMERGE uses the CODEDEF from each FONTDEF. Therefore, you can specify a different CODEDEF for each font-family member in a form. This option could be useful for special fonts, such as languages other than English. You can select the FGRPDEF with either of two DMGMERGE commands: • **FILEDEF**, in one of two parameters: FGRPDEF= selects the FGRPDEF directly PRINTDEF= selects the FGRPDEF named in the PRINTDEF • **MERGEDEF**, in one of two parameters: FGRPDEF= selects the FGRPDEF directly PRINTDEF= selects the FGRPDEF named in the PRINTDEF **NOTE:** Refer to "The FILEDEF Command" on page 406 and "The MERGE Command" on page 435, and "FGRPDEF (Font Group Definition)" on page 83 for more information.

Parameter	Value
CKPERROR=	(Optional) use to generate a W-level error message if a form specified for CKP (concatenate and keep with previous form if possible) does not have space to print on the page with the previous form. For details about CKP, see "EOO-SEP-CON" on page 231. Valid values are: `Y` or `YES` — (Default) generate a W-level error message if a form has a print option of CKP but cannot be concatenated with the previous form. `N` or `NO` — Do not generate an error message.
COMPATV1=	Indicates whether DMGMERGE processes Documerge 1.7 Reserved Tags as version 3.x Reserved Tags or ordinary tags. Some Documerge 1.7 Reserved Tag names did not have the prefix **DMG.**, which identifies all Documerge 3.x Reserved Tags. Some version 1.7 Reserved Tag names began with **DMG.**, but their names were changed in subsequent Documerge versions. With the COMPATV1=YES parameter, you can continue to use these tags as Reserved Tags in VRFs and BPSDs without recoding. Valid values are: **Y** *or* **YES** — DMGMERGE processes Documerge 1.7 Reserved Tags as if they had the DMG. prefix. The default value is YES. **N** *or* **NO** — DMGMERGE processes Documerge 1.7 Reserved Tags as ordinary tags. COMPATV1=NO also helps DMGMERGE run slightly faster. Refer to "Documerge Version 1.7 Reserved Tags" on page 329 for a list of the version 1.7 Reserved Tags and their equivalents in Documerge 3.x.
CTAGTRIGGER=	Specifies a 3-character prefix for reserved tags that evoke command tag processing. For an explanation of command tags, and instructions for preparing them, see "GLOBAL Parameters" on page 386. Format: `CTAGTRIGGER=xxx` -or- `CTAGTRIGGER=NONE` (the default, no trigger) `xxx` can be any three character string. Oracle recommends using **$$$** for this string, but the string characters do not have to be the same. To use special characters such as commas, blanks, or parentheses, place those characters in single quotes. For example: `CTAGTRIGGER='(+)'.`
DEFAULTCD=	Specifies a CODEDEF (Metacode EBCDIC to ASCII translation table) to use when Documerge has been using the default internal CODEDEF, and not properly printing the **~** (tildy: EBCDIC hex A1) and the **{** (left brace:EBCDIC hex C0). Use if the GLOBAL CDFROM parameter isn't set to get CODEDEF(s) from the form-index begin record or FGRPDEF. Format: `DEFAULTCD=`*`codetable`* Replace the *`codetable`* placeholder with a CODEDEF from your PELIB. CODEDEFs must begin with the letters **DC, and are** created using the DPLDUTL utility program. For details, see "The DPLDUTL Utility" on page 74.

Parameter	Value
DELINTTAGS=	Indicates whether DMGMERGE honors the following parameter values for BPSDs that specify internal Reserved Tags: ■ DELETE=**Y** ■ MULTI=**1** *through* **9** (all valid numeric values other than 0) If you code any of these BPSD parameter values for an internal Reserved Tag you must also code **Y** or **YES** in DELINTTAGS= to get the results you intend. Valid values are **Y** *or* **YES** — DMGMERGE deletes the internal Reserved Tag's variable-data string after inserting it once. **N** *or* **NO** — DMGMERGE does not delete the internal Reserved Tag's variable-data string. The data string remains in memory and is inserted into the next Boilerplate Space that specifies the internal Reserved Tag. NO is the default. The DELINTTAGS= parameter applies only to internal Reserved Tags. Refer to "Internal Reserved Tags" on page 284 for more information. Refer to *Composing Forms for Documerge* for information about the BPSD parameters DELETE= and MULTI=.
DJDECR=	(Optional) writes EBCDIC-coded comment records containing the current DJDE parameters at the beginning of each Merge Set. Use this parameter when you have a reprint utility program that requires the full set EBCDIC DJDE records before it can start reprinting a Merge Set. Because DJDECR= saves the DJDE data occurring at the start of the output of the Merge Set, it eliminates need for the reprint program have complex logic to parse the DJDEs. DMGMERGE writes the records inside the current begin-end DJDE comment records, in the following format: 1 *Constant — %%%D97DJDEBEG%%% — with hex 03 carriage control.* 2 *New EBCDIC DJDE comment records, with hex 03 carriage control.* 3 *Current ASCII DJDE comment records, with carriage control other than hex 03 (usually a hex 01).* 4 *Constant — %%%D97DJDEEND%%% — with hex 03 carriage control.* Valid values are: **N** or **NO** — (Default) do not write EBCDIC DJDE comment records. **Y** or **YES** — Write EBCDIC comment records containing the current DJDE parameters at the start of each Merge Set.
DYNAMICCG=	(For AFP only.) Indicates whether DMGMERGE uses dynamic COPYGROUP names. Valid values are: **Y** *or* **YES** — DMGMERGE uses the dynamic COPYGROUP names and ignores: • The default COPYGROUP • A COPYGROUP specified by a USERx option • A COPYGROUP that exists in a form **N** *or* **NO** — DMGMERGE does not use the dynamic COPYGROUP names. The default value is NO. Refer to "Using Dynamic COPYGROUPs in an AFP Environment" on page 180 for more information.

Parameter	Value
ERRVRF=	The name of the error VRF output file. This error VRF duplicates the regular VRF for the Merge Set, with the addition of Reserved Tags that identify the errors. The syntax is: `ERRVRF=ddname` You can name the file whatever you want. Oracle suggests that you use **ERRVRF** for the DDname. This parameter is optional, and it has no default. It overrides any DDname in the DMG.ERROR.VRF Reserved Tag. Here are its processing rules: **(1)** If you use the ERRVRF parameter, DMGMERGE writes *every* Merge Set that has an error to this DDname, regardless of any DDname in the DMG.ERROR.VRF tag. **(2)** If a DDname is not specified in the ERRVRF parameter but is specified in characters 1-8 of the DMG.ERROR.VRF tag's value, DMGMERGE writes only the Merge Sets that have errors and contain the tag to the DDname. **(3)** If no DDname is specified in either the ERRVRF parameter or the DMG.ERROR.VRF tag, DMGMERGE does not write Merge Set errors to any output file and does not use an allocation file. Refer to "DMG.ERROR.VRF" on page 307 for more information. **NOTE:** A JCL DD statement is required for the specified DDNAME. **IMPORTANT:** Do not use the ERRVRF file when your DMGMERGE return code is higher than 12 (i.e. RC=16 or RC=20, an F or P level error). With a return code of 16 or 20 DMGMERGE terminates abruptly, without completing all of its normal functions. In this case the error VRF would not be a complete dataset; for example, it would not contain any Merge Sets subsequent to the one it was processing at the time of the system abend.
ERRVRFA=	The name of the allocation (DMGVRFA) file for the error VRF. This DMGVRFA file defines the amount of storage that DMGMERGE uses for the file named in the ERRVRF parameter. It is used as input to the Generic VDR (DMGVDRG) program, with the DDname DMGVRFA. A DMGVRFA file contains only one record. (Refer to "" on page 252 for more information.) This parameter is optional. The default DDname is ERRVRFA. The syntax is: `ERRVRFA=ddname` You can name the file whatever you want. Oracle suggests that you use **ERRVRFA** for the DDname. This DDname overrides any error VRF allocation file name specified in characters 9-16 of the DMG.ERROR.VRF Reserved Tag value. Refer to "DMG.ERROR.VRF" on page 307. **NOTE:** A JCL DD statement is required for the specified DDNAME. For more information refer to "DMGMERGE Step" in *Migrating Documerge*.

Parameter	Value
FLIPINVF=	Optionally use only in Metacode printing environments to direct the DMGPRNTX program to scan the FGRPDEF (Font Group Definition) to determine if the current font name for the first character of the current BPSD is an inverse font. If this is the case, DMGPRNTX reverses (flips) the print data (represented by the number of replacement characters in the BPSD) to be printed with that font. When using FLIPINVF, all fonts specified within a single BPSD must be either normal or inverse orientation, and the FGRPDEF must be set up for all fonts used in the DMGMERGE run, with the fonts in their proper slots. For details about coding a FGRPDEF with inverse fonts, see "FGRPDEF (Font Group Definition)" on page 83. If an inverse font is not found in the FGRPDEF, the DMGPRNTX program assumes it is **not** an inverse font, and does not reverse the data, and does not issue an error message. Valid values are: `N` or `NO` — (Default) do not scan the FGRPDEF to determine whether or not to reverse the BPSD data. `Y` or `YES` — Scan the FGRPDEF to determine whether or not to reverse the BPSD data.
FORMNAMECR=	(Optional) writes a comment record when reading the first record of an EDL form, and another comment record when reaching the EOF (end-of-file) for the form. Use this parameter when you have a reprint utility program that needs information about a form it is about to reprint. The written comment records contain the following items: ▪ EDL member name ▪ Revision ▪ Chain ▪ EDL DDname (filename for VSE) **NOTE:** due to Overlay processing with a main (non-Overlay) form of multiple pages, FORMNAMECR=YES might generate two start-of-form comments before generating an end-of-form comment. Valid values are: **N** or **NO** — (Default) do not write any comment records for the EDL forms. **Y** or **YES** — Write a comment when starting to process each new EDL form, and write another comment when finished processing the form.
FVL=( )	(Optional) specifies a **Front Overlay** for all the odd pages — except the banner and trailer pages — of all Document Packages processed in a DMGMERGE run. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.
FVLDEFAULT=	Resets the default value for the WHEN parameter of the Rulebase Library Structure Rule FOVERLAY=. The WHEN parameter can have one of three values: **ALWAYS** — The Overlay prints on odd every page in the DTN. **NONBLANK** — The Overlay prints only if the page is odd and non-blank; that is, if the page contains a non-Overlay form. **ONLYBLANK** — The Overlay prints only on blank pages; that is, pages that have no other form printed on them. This option could be used, for example, to automatically generate pages that say "This page intentionally left blank."

Parameter	Value
INDEXCR=	(Optional) writes **index packet** comment records, at the start of the form after the comment record generated by the FORMNAMECR= optional parameter. For details about the FORMNAMECR= parameter, see "FORMNAMECR=" on page 391. **NOTE:** this action of this parameter does not affect existing comment records written because of the MERGEDEF OUTCR=Y parameter. Valid values are: **N** or **NO** (Default) do not write any index packet comment records. **Y** or **YES** Write index packet comment records, at the start of the form after the optional "FORMNAMECR" comment record.

INITENV=	The name of the load module that initializes the environment of a language in which an optional user-exit subprogram is written. (The load module is usually supplied by the vendor of the compiler.)
	You must code the INITENV= parameter if the environment of the subprogram's language must be initialized before the subprogram can be called.
	Also, to avoid problems due to the re-initialization of the language environment every time the user-exit subprogram is called, Oracle strongly recommends that you code the INITENV= control statement. For example, COBOL-2 will always reinitialize working storage, if you don't initialize the COBOL-2 environment.
	You can call a user-exit subprogram during DMGMERGE processing to manipulate the value of a DMG.C.*xxx* Reserved Tag. (Refer to "DMG.C.xxx Reserved Tag Processing" on page 331 for more information.)
	Following are some examples that use the initialization load module for COBOL II:
	COBOL II `ILBOSTP0`
	(The fourth character is an alphanumeric **O**; the last character is a numeric zero.)
	The following example shows the valid syntax for INITENV=:
	`GLOBAL INITENV=progname`
	Therefore, to call a COBOL II subprogram in MVS you must code
	`GLOBAL INITENV=ILBOSTP0`
	You can code more than one INITENV= parameter in the same GLOBAL command. You can also code multiple INITENV= parameters in separate GLOBAL commands in the same DMGMERGE run. You can code the other GLOBAL parameters with INITENV= in the same GLOBAL commands.
	NOTE: No check is made for duplicate INITENV= parameters. The language of your user-exit subprogram determines whether the environment can be initialized more than once.

IOCLAN=	(Optional) Controls whether the X-coordinate or the Y-coordinate of an AFP IOC image is shifted for landscape forms (DMG.OPT orientation value = "LAN"). Valid values are: **Y** or **YES** Shift the X-coordinate (horizontal). This is the recommended setting. **N** or **NO** Shift the Y-coordinate (vertical). This is the default, to maintain compatibility with prior Documerge releases.

Parameter	Value
MANDATORY=	Allows the user to select value criteria for mandatory BPSD tags; specifically, allows the user to select whether Documerge accepts blank (x'40') characters as defining a mandatory tag. Mandatory BPSDs must be defined using the TYPE=M parameter (see *Composing Forms for Documerge*, Chapter 3, for further discussion of the TYPE= parameter). TYPE=M merely determines whether a given BPSD is present in the form; by itself it does not check to ensure that the BPSD has a non-blank value associated with it. The MANDATORY= parameter allows the user to select whether Documerge should test for non-blank values. MANDATORY= has two parameters: `MANDATORY=`**`PRESENT`** *and* `MANDATORY=`**`NONBLANK`** The default is PRESENT. PRESENT merely checks for the presence of the BPSD tag in the form. NONBLANK not only checks for the presence of the BPSD tag in the form, but also tests to ensure that the tag has a non-blank value. **NOTE:** Mandatory=Nonblank checks for tags with zero data length. Since it is possible for a VDR to generate a tag directly by DMGVRFWR containing blanks, but with a non-zero length, it is possible that occasionally Documerge accepts a technically "blank" tag as non-blank. Tags written by DMGRFMT, by design, have trailing blanks truncated; all blank data is truncated and the tag has a zero length. MANDATORY examines every BPSD in the DMGMERGE run that has TYPE=M. MANDATORY has no effect on Dynacomp VSDs.
MSG540=	(**Metacode users only**) Suppresses message “DMGMRG540W Tagged data contained one or more Metacodes...”. The format of this parameter is `MSG540=`*`value`* where ***value*** is the placeholder for one of the following values: **Y** *or* **YES** — Issue message 540 if the variable data contains characters with EBCDIC values less than Hex 10 (decimal 16). *This is the default.* **N** *or* **NO** — Don’t issue message 540.
OPENOUT=	Indicates whether DMGMERGE opens an output file immediately. **Y** *or* **YES** — DMGMERGE opens an output file immediately if there are no SYSIN errors. **CAUTION:** The GLOBAL OPENOUT=Y parameter opens any DMGMERGE output files as soon as they are known. When this occurs for segmented files (for an explanation of segmentation, see "" on page 424), DMGMERGE opens all possible real segment names. If there are many file segments, DMGMERGE might run out of REGION memory because opening a file requires storage below the 16M line. Also, if the segmented files are tape files, DMGMERGE might run out of tape drives. For segmented output, Oracle recommends using the FILEDEF OPENOUT parameter instead. For details, see "OPENOUT=" on page 413. **NOTE:** Assign the Y or YES value if you post-process the DMGMERGE output. This value ensures that an empty file has an end-of-file record. **A** *or* **ALL** — DMGMERGE opens all output files immediately if there are no SYSIN errors. **N** *or* **NO** — (Default) DMGMERGE opens an output file only when data for writing to the output file exists.

Parameter	Value
OUTCR=	(Optional) overrides the OUTCR values in all MERGEDEFs. Indicates whether DMGMERGE places comments in the output files. Valid values are: **Y** *or* **YES** DMGMERGE places comments in the output. **N** *or* **NO** DMGMERGE does not place comments in the output. **M** *or* **MERGEDEF**(Default) use the OUTCR value from the MERGEDEF. For example, all your MERGEDEFs specify `OUTCR=NO`, but you want to generate form-name comment records (`FORMNAMECR=Y`) to aid in debugging. To accomplish this, you would code GLOBAL OUTCR=Y FORMNAMECR=Y
OVL=( )	(Optional) specifies an **Overlay** for all of the even and odd pages — except the banner and trailer pages — of all Document Packages processed in a DMGMERGE run. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.
OVLDEFAULT=	Resets the default value for the WHEN parameter of the Rulebase Library Structure Rule. The WHEN parameter can have one of three values: **ALWAYS** The Overlay prints on every page in the DTN. **NONBLANK** The Overlay prints only if the page is non-blank; that is, if the page contains a non-Overlay form. **ONLYBLANK** The Overlay prints only on blank pages; that is, pages that have no other form printed on them. This option could be used, for example, to automatically generate pages that say "This page intentionally left blank."
PASS2MSG360=	Suppresses message “DMGMRG360W Data length is greater than number of subchars...” for certain tags. The tags are: ▪ DMG.TOTAL.SHEETS ▪ DMG.TOTAL.PAGES ▪ DMG.ITEM.COUNT.VERIFY The format of this parameter is `PASS2MSG360=`*`value`* where ***value*** is the placeholder for one of the following values: **Y** *or* **YES** Issue message 360 if the data is truncated to fit into the BPSD for any of the above tags. *This is the default.* **N** *or* **NO** Don’t issue message 360 if data is truncated to fit into the BPSD for any of the above tags. This value makes DMGMERGE behave as it did in release 3.0, truncating with no warning for the above tags.

Parameter	Value
PROCESS=	(Optional) A debugging and optimization tool that suspends generation of printed output for a Merge Set while still reading the SYSIN and the VRF. When you specify PROCESS=NO, DMGMERGE will ■ Validate the SYSIN syntax. ■ Validate the GLOBAL STARTID= and STOPID= values. DMGMERGE will report if any STARTID or STOPID values were not found. **Y** *or* **YES** (Default) DMGMERGE calls all the merge routines to process the Merge Set and produce output. **N** *or* **NO** DMGMERGE does not produce any output other than a message file, but still reads the SYSIN and the VRF. **Note:** DMGMERGE also generates the following informational messages: ■ DMGMRG184I reports WORKBUFF was not fully used because DMGMERGE did not collate or merge any forms, or produce any output files. ■ `DMGMRG191I Note..GLOBAL PROCESS=N or NO was specified...No merge sets were processed` DMGMERGE generates this message to remind you that you specified `PROCESS=N` or `PROCESS=NO`. ■ `DMGMRG192I Number of merge sets that would have been processed: xxx` `xxx` is the number of Merge Sets that would have processed and matched at least one of the `STARTID/STOPID` or `SELECT` criteria (or all the Merge Sets that would have processed if no `STARTID/STOPID` or `SELECT` parameter were used). ■ DMGMRG303I reports FORMSBUFF as an unused parameter because DMGMERGE did not read any forms from the EDL. For details about these error messages, see their entries in *Documerge Error Messages*.
RPAGE=	Optionally use in a Metacode printing environment to specify how DMGMERGE forces a new sheet of paper. Produces faster print speed by reducing the number of DJDE records. The syntax is: RPAGE=***value*** Where ***value*** can be one of the following: **NONE** This is the default, which is not to generate an RPAGE statement, but to generate the standard DJDE SIDE=NUFRONT statement. **NOGRAPHIC** If not at the beginning of a new Merge Set, and there is no graphic on the page, generate an RPAGE statement. Otherwise, generate the standard DJDE SIDE=NUFRONT statement. **IMPORTANT:** To avoid errors in printing graphics, use the ALWAYS value only when none of the forms to be printed contain inline graphics (DJDE GRAPHIC commands). **ALWAYS** In all cases, generate an RPAGE statement to force a new sheet. Oracle recommends the ALWAYS option when there are a large number of small Merge Sets with no graphics to be printed, or the printer displays frequent "OUTPUT HAS CAUGHT UP WITH INPUT" messages. If you choose the NOGRAPHIC or ALWAYS option, RPAGE statements must be defined for your printer JDEs, and for the PRINTDEFs used by Documerge.

Parameter	Value
SELECTID=	(Optional) specifies the DMG.MERGESET.ID reserved tag value that identifies the Merge Set to be processed. Use as a shortcut replacement for STARTID=value / STOPID=value. SELECTID sets the STARTID and STOPID to the same values so that a particular Merge Set is selected and processed. You can code Multiple SELECTID statements for a single GLOBAL command, and/or multiple GLOBAL commands with SELECTID statements. Format: **`SELECTID=`***`idvalue`* *`idvalue`* is the exact value of the DMG.MERGESET.ID reserved tag to select from the VRF. If this value contains non-alphanumeric characters (characters other than letters or numbers), it must be enclosed in single quotation marks. The value can be 1 to 35 characters. If the value is less than 35 characters, DMGMERGE assumes trailing blanks. For example, `SELECTID='one single MergeSetid'` requests that DMGMERGE select for processing, the Merge Set identified by the value of its DMG.MERGESET.ID tag (one single MergeSetid).
SORTTAGS=	(Optional) specifies whether or not the tags written to the VRF are to be sorted for faster look up. Usually, this parameter should be omitted, or have a value of Y or YES. **Y** *or* **YES** — DMGMERGE sorts the tags. **N** *or* **NO** — DMGMERGE does not sort the tags. This value yields faster processing when there are many Merge Sets with few tags in each Merge Set.
STARTID=	(Optional) locates a Merge Set ID in the VRF to mark the beginning of a partial Merge Set run. Format: `STARTID=startvalue` Where `startvalue` is the exact unique value of the DMG.MERGESET.ID Reserved Tag in the VRF. If this value contains other than alphanumeric characters (characters other than letters or numbers), it must be enclosed in single quotation marks. Here are the rules for coding the STARTID= parameter: ▪ You can code multiple STARTID /STOPID parameters in a single GLOBAL or multiple GLOBAL commands. ▪ All Merge Set IDs must be unique, and a Merge Set ID included in more than one STARTID/STOPID pair will be processed only once. ▪ You must code all STARTIDs and STOPIDs as pairs with no intervening other parameters. DMGMERGE searches sequentially for each STARTID value, and then processes up to and including the paired STOPID value. If Documerge does not find a STARTID value, then it does not process any of the Merge Sets in the range specified by a STARTID/STOPID. ▪ To start with the first Merge Set in the VRF, you can use the reserved word `FIRST` to code `STARTID=FIRST` or `STARTID='FIRST'`. For the STARTID value, the reserved word `LAST` is not allowed. To code a Merge Set ID tag with the value of *FIRST* instead of the reserved word `FIRST`, place at least one trailing blank after the value and delimit the value and blank with single quotes. For example, `STARTID='FIRST'` will search for the Merge Set ID with the value of `FIRST`. **NOTE:** If you run DMGSORT to sort the VRF, the resulting Merge Set sequence is not necessarily the same as the sequence generated by the VDR.

Parameter	Value
STATSFILE=	Generates a separate file containing audit statistics about the forms used in the Documerge run, including: ▪ Form names used ▪ Libraries accessed for forms (if multiple EDLs) ▪ Occurrences of each form (how many times used and locations) ▪ TAG names referenced ▪ Errors generated Format: `STATSFILE=ddname` Where `ddname` is the DDname for the output file. STATSFILE is optional. The Documerge default is to NOT generate a report.
STATSTYPE=	(Optional) specifies the type of STATSFILE. Format: `STATSTYPE=(type type type ...)` For the following `type` values, a value with the **FULL...** prefix overrides a value with the **SUMM...** prefix if both are specified. Each `type` can be one of the following values: **SUMMARY** (Default) Produces a summary list of all forms used in this run. For use with Oracle I.R.I.S. with the DMGDELET program, SUMMARY is sufficient. **SUMMFORM** Same as SUMMARY. **SUMMTAG** Produces a summary of all tag names looked up during the DMGMERGE run, including internal tags such as "DMG.FLST.*Groupname*" and "DMG.DD.*Groupname*" and other "DMG." tags used for DMGMERGE control. **SUMMBPSD** Produces a summary on BPSD tags but not internal tags. **SUMMERROR** Produces a summary of all errors found during the DMGMERGE run. Lists the 3-digit DMGMERGE message identifier and the number of times each error occurred. **FULL** Produces a list of all forms used by Merge Set. **FULLFORM** Same as FULL. **FULLBPSD** Lists all BPSD tag names used in processing each Document Package. **FULLTAG** Lists all tag names looked up, by Merge Set. Reports on all tags, including internal tags such as "DMG.FLST.*Groupname*" and "DMG.DD.*Groupname*" and other "DMG." tags used for DMGMERGE control. **FULLERROR** Produces a list of all errors found during the DMGMERGE run, by Merge Set. Lists the 3-digit DMGMERGE message identifier and the number of times each error occurred in the DMGMERGE run.

Parameter	Value
STOPID=	(Optional) locates a Merge Set ID in the VRF to mark the end of a partial Merge Set run (begun with a STARTID= card). If the next Merge Set has the same value, Documerge processed that Merge Set, and continues processing until it does not find the value. Format: `STOPID=endvalue` Where `endvalue` is the exact value of the selected DMG.MERGESET.ID Tag in the VRF. If this value contains other than alphanumeric characters (characters other than letters or numbers) it must be enclosed in single quotation marks. Here are the rules for coding the STOPID= parameter: ■ You can code multiple STARTID /STOPID parameters in a single GLOBAL or multiple GLOBAL commands. ■ To work correctly, the STOPID Merge Set ID value must be unique (if two Merge Sets have the same STOPID value, Documerge only processes the first occurrence. ■ To process through the last Merge Set in the VRF, you can use the reserved word LAST to code `STOPID=LAST` or `STOPID='LAST'`. For the STOPID value, the reserved word `FIRST` is not allowed. To code a Merge Set ID tag with the value of *LAST* instead of the reserved word LAST, place at least one trailing blank after the value and delimit the value and blank with single quotes. For example, `STOPID='LAST'` will search for the Merge Set ID with the value of *LAST*. **NOTE:** If you run DMGSORT to sort the VRF, the resulting Merge Set sequence is not necessarily the same as the VDR-generated sequence.
STRIPCOLOR=	Indicates whether DMGMERGE deletes Metacode color-printing commands from forms. This deletion depends on the value of the COLOR= parameter in the PRINTDEF (for details, see "COLOR=" on page 100.) Some Xerox printers do not recognize Metacode color-printing commands. You can use the optional STRIPCOLOR= parameter for Merge Sets routed to these printers. **Y** *or* **YES** — If the PRINTDEF does not specify COLOR=Y, DMGMERGE deletes color-printing commands. **N** *or* **NO** — DMGMERGE does not delete color-printing commands. The default value is NO.
STRIPFEED=	(Optional) indicates whether DMGMERGE deletes any FEED= DJDE parameters in Metacode forms. **Y** *or* **YES** — Delete any FEED= DJDE parameters in Metacode forms. Use if there are FEED= DJDE parameters in the Metacode forms, but you want Documerge to control the input tray selection instead (based on MAIN, AUX, or FEED/CLUSTER print options in the Rulebase). **CAUTION:** Oracle recommends that you control input tray selection using Rulebase print options instead of coding FEED= DJDEs in your forms. If the forms have FEED= DJDEs, and you don't use STRIPFEED=Y, the FEED= parameters will be written to the output file, and may conflict with the Rulebase input tray selection values. **N** *or* **NO** — (Default) do not delete any FEED= DJDE parameter in Metacode forms. Any forms that have a FEED= DJDE will print from the input tray specified by their FEED values. Note that DMGMERGE cannot detect this, and therefore its Sheet-Totals-By-Tray report might be incorrect. Any forms that do not have FEED= will use the FEED value from the Rulebase.

Parameter	Value	
STRIPOTEXT=	Specifies if DMGMERGE 3.0.8, 3.1.x or 3.2 should strip (remove) any OTEXT DJDE parameters from Metacode forms. The OTEXT DJDE parameter forces a new sheet of paper, and may cause erroneous DMGMERGE page parity.)	
	Y *or* **YES**	Remove any OTEXT DJDE parameters from Metacode forms. This is the recommended value.
	N *or* **NO**	Do not remove any OTEXT DJDE parameters. This is the default to maintain compatibility with prior Documerge releases. Note that any OTEXT forces a new sheet of paper, and could cause erroneous page parity in the DMGMERGE output.
STRIPINVERT=	(**Metacode users only**) Keeps any INVERT DJDE parameters that exists in EDL forms. The format of this parameter is **STRIPINVERT=***value* where ***value*** is the placeholder for one of the following values:	
	Y *or* **YES**	Strip (remove / ignore) any INVERT parameters in DJDE records in EDL forms. *This is the default.*
	N *or* **NO**	Keep any INVERT parameters in DJDE records in EDL forms.

Parameter	Value

TAG=	A BPSD tag name and its user-defined variable data. This allows for printing of unique variable data on forms specified for all Document Packages in a DMGMERGE processing run.
	For example:
	`TAG=(BPSD.TAG.NAME 'VARIABLE DATA')`
	The tag name cannot contain blanks. Variable data containing blanks must be enclosed in single quotes, variable data containing single quotes must be coded as two single quotes.The total characters inside the parentheses must be 70 characters or less, including any quotes.
	DMGMERGE performs any ***Delete*** processing previously defined for the tag name.
	For the TAG= parameter, the tag's value may span multiple SYSIN records as follows:
	▪ After the last character of the first line of the value, code an ending single quote. (The tag value must be in single quotes.)
	▪ Code a blank/hyphen (standard SYSIN continuation characters).
	▪ On the next record, code a quote, and continue the value. The data following this quote butts up to the data ended by the quote in the preceding record.
	▪ To end the value, code a single quote followed by a right parenthesis (standard for the TAG= parameter).
	▪ If more parameters follow for this GLOBAL command, remember to follow the right parenthesis with a blank/hyphen as usual.
	Example:
	`TAG=(TAGNAME 'This is the value for this tag which'-`
	`'is so large that we must use three SYSIN records' -`
	`'to code all of it.')`
	The priority for searching and processing TAG= is as follows:
	1 TAG= parameter from MERGE command
	2 TAG= parameter from FILEDEF command
	3 TAG= parameter from GLOBAL command
	4 VRF tags
	5 Internally generated tags (such as DMG.PAGE.NUMBER and DMG.CURRENT.DATE)
	NOTE: Documerge places no limit on the number of TAG= parameters per GLOBAL command.

USERTAG=	Turns special processing for User Index Tags on or off. User Index Tags are used for the Oracle product ImageCreate. They are available also for any other program that post-processes DMGMERGE output and needs special information. The names of these tags have the prefix *USER*. See "User Index Tags" on page 188 for more information. **Y** *or* **YES** — Turns on special processing for User Index Tags. The default value is YES. **NOTE:** If this value is Y or YES and the OUTCR= value in the MERGEDEF PEDEF is YES, the values of all User Index Tags in the VDR or Rulebase Library Tag Table are placed in DMGMERGE output as comments. **N** *or* **NO** — Turns off special processing for User Index Tags. The NO value helps DMGMERGE run slightly faster.

Parameter	Value
***x*VL=( )**	Document Package-level **Overlay** which you can specify one or more times in the DMGMERGE SYSIN using the GLOBAL, FILEDEF, or MERGE commands. For further explanation, see "Specifying Overlays with the DMGMERGE Commands" on page 474. Format: ***COMMAND*** *x*VL=(***'member name'***`(revision) when appliesto where`) Replace the ***COMMAND*** placeholder with a value that indicates the DMGMERGE SYSIN command you want: ▪ `GLOBAL` — Applies the Overlay to all Document Packages processed in a DMGMERGE run. ▪ `FILEDEF` — Applies the Overlay to the Document Package specifed by the FILEDEF command. ▪ `MERGE` — Applies the Overlay to the Document Package processed and written for the Group specified in the MERGE command. Replace the *x* placeholder with a value that indicates the type of Overlay you want: ▪ `BVL` — back side only ▪ `FVL` — front side only ▪ `OVL` — both front and back side The outside parentheses are required; if omitting the revision, omit the inner parentheses. Valid values for the placeholders inside the parentheses are: **member name** (Required) the 1- to 32-character name of the form used as an Overlay. If there are blanks, parentheses, or commas in the member name, the delimiting single quotes are required, otherwise they are optional but recommended. **(revision)** (Optional) a 1- to 5-character revision number or identifier delimited with parentheses. Valid values are: **0** The highest revision level at the time DMGMERGE is executed. The default is **0**. **n** A specific revision level. The value for **n** can range from 1 to 32,767. **VDR** The highest revision level at the time the VDR is executed. **MERGE**The highest revision level at the time DMGMERGE is executed. **when** (Optional) specifies the printing an Overlay based on whether or not the page contains other forms. The **when** values are mutually exclusive; code at most one value per Overlay. **when** does not apply for logical page Overlays. For example: `GLOBAL BVL=('FIRSTSHEET' ONLYBLANK)` Specifies that an Overlay named `FIRSTSHEET` is to print *when* an even page is blank (`ONLYBLANK`) for all of the Document Packages processed in a run.

Parameter	Value	
		Valid values for **when** are: **DEFAULT** Use the default specified in the `GLOBAL FVLDEFAULT`, `BVLDEFAULT`, or `OVLDEFAULT` parameters. **ALWAYS**The Overlay prints on every page. `ALWAYS` is the default. **NONBLANK** The Overlay prints only if a non-Overlay form is printed on the page. **ONLYBLANK** The Overlay prints only if no other form is printed on the page.
	appliesto	(Optional) specifies the type of page on which to print an Overlay. The **appliesto** values are mutually exclusive; code only one value per statement.
		TIP: When using the Routing-By-Sheets (MAXSHEETS=) feature, and you want different Overlays for different files, use the SHEET option. Documerge applies SHEET overlays after it has counted the number of sheets and the final output data set has been determined.
		For example: `FILEDEF FVL=('FIRSTSHEET' ONLYBLANK PHYSICAL)` Specifies that an Overlay named `FIRSTSHEET` is to print *when* there is an odd side of a blank (`ONLYBLANK`) page that *appliesto* the physical page. Valid values for **appliesto** are: **LOGICAL**Prints the Overlay on each logical page (or physical page if no logical pages are defined).
		NOTE: The Overlay will print for all logical pages defined for the DTN because any front (FVL) or back (BVL) Overlays refer to the physical page.
		PHYSICALPrints the Overlay on the physical page. `PHYSICAL` is the default currently used by Documerge. **SHEET** Prints the Overlay on the first available side of the sheet of paper.

Parameter	Value	
	where	(Optional) specifies the printing location of an Overlay in an Overlay set. The **where** values are mutually exclusive; code only one value per DTN. **where** does not apply to inline forms or logical overlays. For example: `MERGE OVL=('FIRSTSHEET' ONLYBLANK FIRST)` Specifies that an Overlay named FIRSTSHEET is to print. if the first page (one side of a sheet of paper) of the Document Package is blank. Valid values for **where** are: **FIRST** Prints the Overlay only on the first page of the Overlay set. Complies with other options like **when** and page parity. **NOTFIRST** Prints the Overlay on all pages except the first. **LAST** Prints the Overlay only on the last page of the Document Package. Complies with other options like **when** and page parity. **NOTLAST** Prints the Overlay on all pages except the last page of the Document Package, complying with other Overlay options such as **when** and page parity. **MIDDLE** Prints the Overlay on all pages except the first and last pages. Complies with other options like **when** and page parity. **ALL** (Default) prints the Overlay on all pages, except the banner and trailer pages, complying with other Overlay options such as **when** and page parity.
		TIP: The DMGMETP and DMGAFPP BOTTOM= parameter controls where the Overlay ends. (The BOTTOM= value is sometimes called the "artificial bottom.") If you do not specify a BOTTOM= value, DMGMERGE assumes that the Overlay is a running header; DMGMERGE prints the other Boilerplate after the last line of the Overlay. If you want the other Boilerplate printed actually on top of the Overlay, code BOTTOM=0 when you normalize the Overlay.

The GLOBAL Command and CTAGTRIGGER-Defined Command-Tag Processing

By definition, Documerge automatically processes those tags with names beginning with *DMG.C....* as command tags. Documerge lets you modify other reserved tags with a 3-character prefix and use these tags for command tag processing.

Using the GLOBAL CTAGTRIGGER parameter to specify this new type of command tag, you can implement command processing with existing forms and existing tag names — and avoid having to recompose and reload the forms as was required for DMG.C.xxx tag processing.

For details about DMG.C.xxx tag processing, see "DMG.C.xxx Reserved Tag Processing" on page 331.

How to Set Up Command-Tag Processing

1. Choose the 3-character string that you want to serve as the trigger prefix for the command tags. Here are some guidelines for choosing this string:
   - Oracle recommends using the non-special characters *$$$* for this string, but all three string characters do not have to be the same character.
   - If you use special characters such as commas, blanks, or parentheses, you must place those characters in single quotes or apostrophes when you code the GLOBAL CTAGTRIGGER parameter.
   - Do not use the single quote — or apostrophe) or hexadecimal 00 (low values) as a string character.
2. In the VDR, generate the following data for the reserved tag that you want to be a command tag:
   a. The 3-character command tag prefix string you chose in step 1.
   b. One or more of the DMG.C.xxx commands (**Tag, Call, Literal**) and any associated data — immediately following the three special characters.

   For details about the command codes, see "DMG.C.xxx Value" on page 332.

NOTE

You can also code a command tag as the value of TAG parameter for the MERGE command, but normally command tags are generated during the VDR step and written to the VRF.

3. In the DMGMERGE SYSIN, code the GLOBAL command with the CTAGTRIGGER= parameter with a value equal to the string you chose in step 1. For example, you could code

   ```
   GLOBAL CTAGTRIGGER=$$$
   ```

   -or-

   ```
   GLOBAL CTAGTRIGGER=',,,'
   ```

   For details about the CTAGTRIGGER parameter, see "CTAGTRIGGER=" on page 388.

If Documerge encounters tag data that begins with the 3-character trigger, it automatically invokes command processing, just like it would for a DMG.C.xxx tag.

NOTE

Documerge does not process *nested* command triggers. That is, the second time it encounters a value evoking the DMG.C.xxx Tag command (get the tag value), Documerge does not execute the command; instead it passes the value as-is.

The FILEDEF Command

The **FILEDEF** command defines an output file. FILEDEF is optional. You can use the FILEDEF command to specify these Documerge options:

NAME=	BANNER=	BANNERFEED=	BTEXTINIT=
BTEXTNSEADD=	BVL=	CHAIN=	CLOSEEXIT=
DEALLOCATE=	ERRNUM=	FGRPDEF=	FVL=
MAXSEGMENTS=	MAXSHEETS=	MERGEDEF=	OPENOUT=
OPTIMIZE=	OTEXT=	OVL=	PACKAGE=
PRINTDEF=	SCANBACKUP=	SEGMENT=	TAG=
TRAILER=	TRAILERFEED=		

Except for the MAXSEGMENTS and MAXSHEETS options, you can specify all of these options using the MERGE command. Except for the CHAIN= and PACKAGE= options, you can also specify these options with Rulebase Library Tags in the VRF. For maximum flexibility, Oracle recommends that you specify all of these options with the FILEDEF command.

NOTE

The MERGE command overrides the FILEDEF command and the VRF.

Regardless of how you specify the MERGEDEF= value, it determines the PRINTDEF= and FGRPDEF= values. PRINTDEF= and FGRPDEF= are valid only if specified in the same command (or VRF Tag) in which the MERGEDEF= is specified.

For example, if you specify the MERGEDEF= value in the FILEDEF command and the PRINTDEF= value in the MERGE command, DMGMERGE ignores the PRINTDEF=; DMGMERGE uses the MERGEDEF= and its default PRINTDEF= specified in the FILEDEF command.

FILEDEF Parameters

Parameter	Value
NAME=	Required. The 1- to 8-character name of the file. If you use the FILEDEF command for any type of file **except a master segment file**, you must code a DD statement in the JCL. The syntax is: `//ddname DD`
BANNER=	(Optional) the name of the form used as a banner sheet for the file. **NOTE:** The BANNER= parameter overrides any value in the VRF for the DMG.BNR.*Groupname* Reserved Tag. You can use this parameter to print a banner sheet that's specialized for the output file you define with the FILEDEF command. (The MERGE command's BANNER= parameter prints a banner sheet for the Group.) You can use this parameter to print a banner sheet for an error file and for a clean file. (These error files are defined in the MERGE command's ERRDDN= parameter or the VRF's DMG.ERDD.*Groupname* Reserved Tag.) Valid values are: **DEFAULT** The Documerge banner sheet, with asterisks. This is the BANNER= default value for clean DDs. **NONE** No banner sheet. This is the BANNER= default value for error DDs. **NOTE:** The MERGEDEF parameter MSGCTLx contains DJDEs that are placed at the beginning of the output file. This initial DJDE packet is separate from the DJDE packet in a form. Xerox processing requires printable characters between DJDE packets; otherwise, the packet in the first form of the output file is ignored. This condition can cause the output file to be unprintable. Therefore, if BANNER=NONE is specified for a clean Metacode file, DMGMERGE places a blank sheet before the first form in the output file. For an error DD, a blank sheet is not needed because the messages follow the initial DJDE packet and precede the first form. **formname** The name of the EDL member you want to use as the banner sheet. This value automatically specifies revision 0. If you want a banner sheet named DEFAULT or NONE, you must specify the revision level.
	NOTE: The MERGEDEF parameter MSGCTLx contains DJDEs that are placed at the beginning of the output file. This initial DJDE packet is separate from the DJDE packet in a form. Xerox processing requires printable characters between DJDE packets; otherwise, the packet in the first form of the output file is ignored. This condition can cause the output file to be unprintable. Therefore, if BANNER=formname is specified for a clean Metacode file, DMGMERGE places a blank sheet before the banner sheet. For an error DD, a blank sheet is not needed because the messages follow the initial DJDE packet and precede the banner sheet. **formname(revision)** The name and revision level of the EDL member you want to use as the banner sheet.

Parameter	Value
BANNERFEED=	(Optional) specifies the input tray name for user banner forms. Format: `BANNERFEED=`*`trayname`* Replace the *`trayname`* placeholder with one of the following: ▪ Xerox Metacode cluster name ▪ IBM AFP COPYGROUP (IMM) name, including the "C" prefix The default value is MAIN. To avoid printing errors, you must ensure that the selected cluster or COPYGROUP name exists. This parameter is ignored for the DMGMERGE default banner page and for line printer output. If you specify BANNERFEED in both the FILEDEF and MERGEDEF, the MERGEDEF takes precedence.
BTEXTINIT=' '	(Optional) specifies that Documerge is to write a BTEXT DJDE record that has the value enclosed in the apostrophes (' '). Documerge writes the BTEXT DJDE before any other DJDEs. You can use the BTEXT DJDE record to produce a bar code for a Document Package. Use this parameter only if your printer is a Xerox 4635 or similar printer. For an explanation of BTEXT, see "The FILEDEF Command and BTEXT-Generated Bar Codes for Xerox 4635 Printers" on page 429. Format: `BTEXTINIT='value'` The subparameter keywords that you code as the BETEXTINIT value are the same as the BTEXT DJDE keywords documented by Xerox. See the Xerox 4635 manual for details. To ensure that you code the BETEXTINIT value correctly, use the following guidelines: ▪ Inside the required apostrophes, code the subparameters that describe exactly what data goes in the BTEXT DJDE record, including any commas between parameters. So that you can use future Xerox enhancements to BTEXT without the requirement of a Documerge upgrade, Documerge does not validate the BTEXTINIT syntax or value. If the BTEXTINIT parameter contains errors, the Xerox printer will probably generate a DJDE syntax error, or BTEXT processing might not work as intended. ▪ If all desired parameters cannot fit into one SYSIN record, you can code multiple BTEXTINIT= parameters, which also generates multiple BTEXT= parameters in the output file. ▪ Because Xerox requires it for proper BTEXT processing, **you must code one RRA= parameter**. ▪ **Do not code any SEQ, TXT, or PRA parameters in the BTEXTINIT value.** DMGMERGE automatically inserts these parameters on pages with a BTEXT barcode. ▪ If you code the %%DATE%% string, DMGMERGE will replace it with today's date in *yy-mm-dd* format (the format required by Xerox). Use this feature to code `PRD=%%DATE%%` to set the process date for the Xerox BTEXT audit report.

Parameter	Value
BTEXTNSEADD=	(Optional) specifies that Documerge is to adjust its computed BTEXT NSE (number of sheets expected) value by adding the number of sheets specified by BTEXTNSEADD. Use BTEXTNSEADD only if your Xerox printer counts JES banner and trailer pages for its internal NSE value, and then generates an error message if that internal NSE value does not equal the NSE value that Documerge reports in the final BTEXT DJDE for the Document Package. For an explanation of BTEXT, see "The FILEDEF Command and BTEXT-Generated Bar Codes for Xerox 4635 Printers" on page 429. Format: `BTEXTNSEADD=nnnnnnnn` The default value is 0; `nnnnnnnn` can be 1 to 8 digits with no commas or other delimiters. The minimum value is 0 and the maximum value is 99999999.
BVL=( )	(Optional) specifies a **Back Overlay** for all the even pages processed and written to the file specifed by the FILEDEF. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.
CHAIN=	(Optional) specifies the chain name to use when selecting EDL members. The default chain name is based on the value of the PRINTDEF PDEV parameter. Use this option to ▪ Mix Template Technology forms and other forms ▪ Combine normalized AFP forms and other AFP forms ▪ Set up independent routing (for details, see "The FILEDEF Command with Independent Routing" on page 418) Valid values are: **name** The chain name. **(name1 name2 ...name5)** Up to 5 chain names. DMGMERGE uses the sequence you specify to search the chains in the EDL member. DMGMERGE uses the first specified chain it finds.
CLOSEEXIT=	(Optional) specifies a custom subprogram for DMGMERGE to call when the associated FILEDEF has finished processing. For each FILEDEF, DMGMERGE sets one or more Function codes (see following close-exit control block format for Function code actions). This option lets you ▪ Start a post-processing job — use with the SEGMENTATION feature to execute a post-processing job while DMGMERGE is still running. ▪ Capture and process information about a file — use to get total sheets, total pages, or total records for a Merge Set. Format: `CLOSEEXIT=`*`pgmname`* Replace the *pgmname* placeholder with the name of a custom load module that you want DMGMERGE to call when the FILEDEF file is closed. When DMGMERGE calls the specified user program, it passes one parameter, a ***close-exit control block***, which is in the following format:

Parameter	Value
	LengthContents 4 bytesConstant "CLSX" indicating a close-exit control block 4 bytesFullword total length of this control block 6 bytesCurrent Documerge version/release/level 1 byteControl block modification number (currently set to "00" indicating the first modification) 1 byteFunction code: **1** = call just before closing the file **2** = call just after closing and optionally de-allocating the file **T** = terminate the DMGMERGE program — final call 3 bytesFiller for alignment purposes 4 bytesFullword user exit return code (set by the user exit, normally should be set to zero) (The following items apply only for Function codes 1 and 2) 8 bytesDDNAME of the real file being closed. With segmentation, this is the real file name, not the master segment name 4 bytesFullword Merge Set count — the number of unique Merge Sets written to the file 4 bytesFullword package count — the number of packages written to the file 4 bytesFullword sheet count — the total number of sheets in the file 4 bytesFullword page count — the total number of pages in the file 4 bytesFullword record count — the record total for the file 4 bytesFullword byte count — the total number of bytes in the file **(The following item applies only for Function code 1)** 4 bytesFullword DCB address of the file
	Here are some rules for CLOSEEXIT processing: ■ CLOSEEXIT applies to both clean and error files. ■ The subprogram will be called twice for each FILEDEF that requests it, plus one additional time at the end of DMGMERGE execution. ■ The first FILEDEF processed is not necessarily the first file to be passed to the CLOSEEXIT subprogram; FILEDEF 2 could be passed first, especially if using segmentation. ■ The subprogram should test the file name field in the Close Exit Control Block to determine the file being closed. The subprogram needs to check the Function code to determine when DMGMERGE called it, and take appropriate action.

Parameter	Value
DEALLOCATE=	Optionally use when defining output segmentation (for an explanation, see "" on page 424). **Because deallocating the file frees it so that it can be printed or post-processed, you will usually need to specify DEALLOCATE=Y for output file master segments.** You can specify DEALLOCATE for any real file, not just a segmented file. Tells MVS to remove a file from Documerge processing resources after the file closes. If the file is a print file, it can now start printing. Or, if the file is a disk or tape file, it can now be read by other programs — all while DMGMERGE is still running. If you use DEALLOCATE in the FILDEF for a master segment, but not in the FILEDEF of a real file segment that belongs to that master segment, then the master segment value applies to the real file. Valid values are: **N** or **NO** When the file closes, do not immediately deallocate it. **Y** or **YES** When the file closes, deallocate it.

ERRNUM=	(Optional) use to specify the routing of a Document Package to an error DD (erred output file) based on the error message(s) generated by Documerge for that Document Package.
	For an explanation of routing-by-errors, see "The FILEDEF Command and Routing-by-Errors" on page 421.
	Format: `ERRNUM=(nnn nnn nnn ...)`
	(or, using a space and the hyphen continuation character)
	`ERRNUM=(nnn -`
	`nnn -`
	`:`
	`nnn)`
	Each `nnn` value is the 3-digit numeric part of a Documerge error message identifier. Each value must be three digits, including any leading zeros.
	The minimum value for each `nnn` is 100 and the maximum value is 999. You can code the error message identifiers in any sequence.
	The parentheses are required if you code more than one value. (As with other SYSIN statements, you can use the hypen continuation character to code values on more than one record.)

	For an alphabetical listing of the error message indentifiers and descriptions of the error messages, see the contents pages in *Documerge Error Messages*.
	In addition to specifying the error message number(s), you must also specify the file to which Documerge will route erred Document Packages by coding the specification for the file in any of the following:
	▪ ERRDDN parameter of the MERGE statement. For details, see "ERRDDN=" on page 437.
	▪ DMG.ERRDD.*Groupname* Reserved BPSD tag. For details, see "DMG.ERDD.Groupname" on page 306.
	▪ NEWDD parameter of the DDRENAME command. For details, see "The DDRENAME Command" on page 384.
	For example, the following Documerge SYSIN:
	`FILEDEF NAME=INSMISSF ERRNUM=(250 410)`
	`MERGE GROUP=INSURED -`
	`DDNAME=INSURED -`
	`ERRDDN=(INSMISSF)`
	specifies that if a Document Package evokes the 250 or 410 error messages, Documerge will write the output Document Package to the INSMISSF error dd specified by the ERRDDN parameter.

Parameter	Value
FGRPDEF=	For Metacode data streams, the name of the Font Group Definition PEDEF to be used during Tumble or Imposition processing. **NOTE:** The MERGE command's FGRPDEF= parameter overrides the FILEDEF command's FGRPDEF= parameter.
FVL=( )	(Optional) specifies a **Front Overlay** for all the odd pages — except the banner and trailer pages — processed and written to the file specifed by the FILEDEF. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.
MAXSEGMENTS=	Use in defining output segmentation (for an explanation, see "" on page 424). Use only in the FILEDEF for the master segment to specify the maximum number of real segments (files) that Documerge will generate. This parameter is not mandatory, but is required to specify more than one segment. Format: `MAXSEGMENTS=nnnnnnn` `nnnnnnn` can be 1 to 7 digits with leading zeros (not required) but no commas. The minimum value is 1 and the maximum value is 9999999. The value for MAXSEGMENTS needs to be compatible with the number of percent signs coded for the master segment filename. For example, if the master segment file name is `FRED%%`, the MAXSEGMENTS value should not exceed 99, because `FRED%%` contains only 2 percent signs for substituting values that range from 01 to 99.
MAXSHEETS=	Defines the maximum number of sheets in a Document Package which can go to the output file specified by the FILEDEF. If Documerge determines that the selection criterion (number of sheets) specified by MAXSHEETS for a file is not met, it checks other user-specifed file names until it finds one that does. Format: `MAXSHEETS=nnnnnnnn` `nnnnnnnn` can be 1 to 8 digits with leading zeros (not required) but no commas. The minimum value is 1 and the maximum value is 99999999. If you omit MAXSHEETS, the default is *no maximum*. Use to ensure finishing equipment requirements for folding, stapling, etc., a maximum number of sheets of paper. For an explanation of routing-by-sheets, see "The FILEDEF Command and Routing-by-Sheets (MAXSHEETS)" on page 420. **NOTE:** MAXSHEETS= does not specify the maximum number of total sheets to write to a file, but the maximum number to write to a single Document Package. Use the SEGMENT= command to specify the total number of sheets. For details, see "SEGMENT=" on page 415.
MERGEDEF=	The name of the MERGEDEF (from the PELIB) that DMGMERGE uses. Code this parameter for the FILEDEF when setting up independent routing. For more information, see "The FILEDEF Command with Independent Routing" on page 418. **NOTE:** The MERGE command's MERGEDEF= parameter overrides the FILEDEF command's MERGEDEF= parameter.

Parameter	Value
OPENOUT=	Indicates whether DMGMERGE opens an output file immediately if there are no SYSIN errors, or waits until there is output to write. The OPENOUT parameter has been implemented for the FILEDEF especially to aid in processing segmented files (for an explanation of segmentation, see "" on page 424). Here are some recommendations for using OPENOUT with segmented files: ■ On a master segment FILEDEF, use the OPENOUT=Y default so the file will open immediately for output. The first real segment will open, and other real segments will not open until needed (unless they have their own FILEDEF without OPENOUT=N, or unless the GLOBAL command OPENOUT=Y). ■ To open all segments for a master segment, either use individual FILEDEFs for each real segment name, or specify the GLOBAL OPENOUT=Y. ■ Alternatively, as in past releases, you can use the DMGOPNCL program to open and close output files. Valid values are: **Y** *or* **YES** DMGMERGE opens an output file immediately if there are no SYSIN errors, and after it knows that the file exists. **A** *or* **ALL** For master segment files, DMGMERGE opens all file segments for the master segement. If there is no master segment, ALL is the same as YES. **NOTE:** Assign the Y or YES value if you post-process the DMGMERGE output. This value ensures that an empty file has an end-of-file record. **N** *or* **NO** (Default) DMGMERGE opens an output file only when data for writing to the output file exists.
OPTIMIZE=	(Optional) indicates whether DMGMERGE should optimize Metacode output. This parameter is ignored for AFP and line printer output. Optimized Metacode combines multiple records into one larger record when possible. This improves the data transfer speed from the mainframe to the printer, and possibly eliminates the Xerox printer message "`Output has caught up with input.`" However, optimization will increase DMGMERGE run time, and requires valid CODEDEF, FONTGRPDEF, and FONT libraries. **TIP:** Because of Xerox processsing rules, it is not always possible to combine form records. Forms containing inline graphics (GRAPHIC= DJDE records) diminish overall printing speed. To improve overall printing speed, you can manually change the forms to use IMG (IMAGE= DJDE records). Valid values are: **Y** or **YES** DMGMERGE attempts to optimize the Metacode output file. **N** or **NO** DMGMERGE doesn't optimize the Metacode output file. The default value is NO.

Parameter	Value

OTEXT=( )	(Optional, Xerox printers only) just before starting to print the file defined for the FILEDEF, specifies a message to display on the Xerox operator's console, and/or suspends printing until the operator restarts the printer.
	
	CAUTION: By XEROX design, an OTEXT DJDE parameter forces a new sheet of paper. Therefore, any OTEXT in a form can cause Documerge page parity to be compromised, and forms that should print on the back side could actually print on the next front side instead.
	
	Oracle strongly recommends composing forms without the OTEXT DJDE parameter. If forms in your EDL already have the OTEXT DJDE parameter, you need to add the GLOBAL STRIPOTEXT=YES parameter to the DMGMERGE SYSIN. For details about STRIPOTEXT=, see "STRIPOTEXT=" on page 399.
	
	The FILEDEF OTEXT parameter displays an OTEXT message before the file starts printing. Setting the GLOBAL STRIPOTEXT parameter does affect the FILEDEF OTEXT parameter processing. If you use the FILEDEF OTEXT parameter, you will get an OTEXT message before the file starts printing, even if you also have specified STRIPOTEXT=YES. STRIPOTEXT=YES only removes OTEXT from forms and MSGCTL.
	
	Use when special preparation is required, such as bar code reader adjustment before BTEXT processing or a change in color ink.
	
	For an explanation of BTEXT, see "Guidelines for Specifying Output Files when Routing-by-Sheets and/or Segmenting Output" on page 418.
	
	Formats: `OTEXT=('message',WAIT)`
	`OTEXT=('message')`
	`OTEXT='message'`
	
	Because of Documerge SYSIN limitations, the total value of the OTEXT parameter (including the parentheses and apostrophes) cannot exceed 70 characters. When used with the `WAIT` constant, the `Message` cannot be more than 65 characters.
	
	For example, you could code the following **OTEXT** value:
	
	`FILEDEF NAME=INSCLEAN -`
	`OTEXT=('-> SETUP FOR DOCUMERGE BTEXT <-',WAIT)`
	
	which causes the following message to display on the printer console:
	
	`-> SETUP FOR DOCUMERGE BTEXT <-`
	
	and also causes printing to be suspended until the operator presses the CONTINUE button on the printer console.

OVL=( )	(Optional) specifies an **Overlay** for all the pages — except the banner and trailer pages — processed and written to the file specifed by the FILEDEF. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.
PACKAGE=	For Metacode data streams, indicates whether DMGMERGE generates the RSTACK value at the end of each Merge Set. Allows stapling the Merge Set as a single Document Package. **N** DMGMERGE does not generate the RSTACK value. **Y** DMGMERGE generates the RSTACK value. **NOTE:** The Y value can cause significantly slower printing. After generating the RSTACK value, DMGMERGE must reissue all DJDEs.
PRINTDEF=	The PRINTDEF name used from the PELIB. Code this parameter for the FILEDEF when setting up independent routing. For more information, see "The FILEDEF Command with Independent Routing" on page 418. **NOTE:** The MERGE command's PRINTDEF= parameter overrides the FILEDEF command's PRINTDEF= parameter.

Parameter	Value
SCANBACKUP=	(Optional) indicates whether the Metacode printer (e.g., Xerox 4635) allows the scan address to move backward to reference a scan pel whose address is lower than a previously printed scan pel in the same Metacode record. If you specify OPTIMIZE=Y and SCANBACKUP=Y, you may get greater Metacode optimization, especially if the output file LRECL is also increased. (Depending how your Metacode printer is defined to the mainframe, the Metacode maximum LRECL can be up to 254 (fixed-length) or 250 (variable-length) for each output record. **CAUTION:** Do not specify **Y** if this output file might be printed on any metacode printer that does not allow the scan address to back up within a record. Those responsible at each Documerge site must determine their Metacode printer capabilities. Valid values are: **N** or **NO** (Default) the Metacode printer does not allow the scan address to back up within a record. **Y** or **YES** The Metacode printer allows the scan address to back up within a record.
SEGMENT=	Use in defining output segmentation (for an explanation, see "" on page 424). Specifies the number and the type of output units that Documerge must meet or exceed before creating an additional real file segment. **Documerge segments only on Merge Set boundaries.** If all file segments will have the same segment criteria, code `SEGMENT=` in the master segment FILEDEF for convenience. If a real segment will have different segment criteria, code the `SEGMENT=` parameter in the real segment FILEDEF. If you code SEGMENT for both the master and one or more real segments, Documerge segments when it detects the first valid value. Format: `SEGMENT=(nnnnnnnnn type)` `nnnnnnnnn` is a number of 1 to 9 digits (leading zeros are valid but no commas are allowed). `TYPE` can be one of the following: ▪ **SHEETS** — segment on or after this many sheets of paper. ▪ **PAGES** — segment on or after this many pages (in duplex, there are always 2 pages per sheet). ▪ **PACKAGES** — segment at one copy for one Group for the Merge Set. ▪ **RECORDS** — segment on or after this many records. ▪ **BYTES** — segment on or after this many bytes of data. ▪ **MERGESETS** – segment when the specified number of mergesets has been written to the output file. This is different from PACKAGES, in that PACKAGES counts each copy and each group, where MERGESETS only counts the actual mergesets regardless of the number of copies or if more than one group writes to the same file. Documerge segments a DD at the next Merge Set boundary, on or after the `nnnnnnnnn` value for the specified real segment **TYPE**. If you code more than one SEGMENT parameter, segmentation will occur for the first segmenting type that meets its criterion.

Parameter	Value

TAG=	A BPSD tag name and its user-defined variable data. This allows for printing of unique variable data on forms specified for a clean (non-erred) output file. Even if there are DMGMERGE errors, the clean file's TAG= value is used. TAG= values for error files are never used.
	
	For example:
	
	`TAG=(BPSD.TAG.NAME 'VARIABLE DATA')`
	
	The tag name cannot contain blanks. Variable data containing blanks must be enclosed in single quotes, variable data containing single quotes must be coded as two single quotes.The total characters inside the parentheses must be 70 characters or less, including any quotes.
	
	DMGMERGE performs any ***Delete*** processing previously defined for the tag name.
	
	For the TAG= parameter, the tag's value may span multiple SYSIN records as follows:
	
	▪ After the last character of the first line of the value, code an ending single quote. (The tag value must be in single quotes.)
	▪ Code a blank/hyphen (standard SYSIN continuation characters).
	▪ On the next record, code a quote, and continue the value. The data following this quote butts up to the data ended by the quote in the preceding record.
	▪ To end the value, code a single quote followed by a right parenthesis (standard for the TAG= parameter).
	▪ If more parameters follow for this FILEDEF command, remember to follow the right parenthesis with a blank/hyphen as usual.
	
	Example:
	`TAG=(TAGNAME 'This is the value for this tag which'-`
	`'is so large that we must use three SYSIN records' -`
	`'to code all of it.')`
	
	The priority for searching and processing TAG= is as follows:
	
	1 TAG= parameter from MERGE command
	2 TAG= parameter from FILEDEF command
	3 TAG= parameter from GLOBAL command
	4 VRF tags
	5 Internally generated tags (such as DMG.PAGE.NUMBER and DMG.CURRENT.DATE)
	
	NOTE: Documerge places no limit on the number of TAG= parameters per FILEDEF command.

Parameter	Value
TRAILER=	The EDL member name used as a trailer sheet for the file. Optional. Use this parameter to print a trailer sheet that's specialized for the output file you define with the FILEDEF command. (The MERGE command's TRAILER= parameter prints a trailer sheet for the Group.) You can use this parameter to print a trailer sheet for an error file and for a clean file. (These error files are defined in the MERGE command's ERRDDN= parameter or the VRF's DMG.ERDD*.Groupname* Reserved Tag.) Valid values are: **DEFAULT** No trailer sheet. **NONE** No trailer sheet. This is the TRAILER= default value for both clean and error DDs. **formname** The name of the EDL member you want to use as the trailer sheet. This value automatically specifies revision 0. If you want a trailer sheet named DEFAULT or NONE, you must specify the revision level. **formname(revision)** The name of the EDL member you want to use as the trailer sheet. This value includes a revision level that you select. **NOTE:** The FILEDEF command's TRAILER= parameter overrides any value in the VRF for DMG.TLR.*Groupname*.
TRAILERFEED=	(Optional) specifies the input tray name for user trailer forms. Format: `TRAILERFEED=`*`trayname`* Replace the *`trayname`* placeholder with one of the following: ▪ Xerox Metacode cluster name ▪ IBM AFP COPYGROUP (IMM) name, including the "C" prefix The default value is MAIN. To avoid printing errors, you must ensure that the selected cluster or COPYGROUP name exists. This parameter is ignored for the DMGMERGE default trailer page and for line printer output. If you specify TRAILERFEED in both the FILEDEF and MERGEDEF, the MERGEDEF takes precedence.

The FILEDEF Command with Independent Routing

By using the FILEDEF command with Independent Routing, you can send different Merge Sets in the same Group to different types of printers in different locations. Instead of using the MERGE command, use the FILEDEF command and the DMG.DD.*Groupname* Reserved Tag to specify these parameters:

- CHAIN
- FGRPDEF (optional)
- MERGEDEF
- PRINTDEF (optional)

Guidelines for Specifying Output Files when Routing-by-Sheets and/or Segmenting Output

DMGMERGE bases output file eligibility for Routing-by-Sheets on any or all of the following:

- The absence or presence of processing errors
- The value of the MAXSHEETS parameter (for routing-by-sheets)
- The values coded for output segmentation

When you specify segmentation, DMGMERGE segments any output files at the start of the Merge Set, according to the SEGMENT and MAXSEGEMENTS parameters defined in one or more FILEDEFs. Segmentation is a separate process from Routing-by-Sheets, and occurs only on Merge Set boundaries.

You can define filenames for clean (error-free) output, output for the routing-by-sheets option, and output for the segmentation option in any of the following:

- The initial MERGE command
- FILEDEF command(s)
- DMG.DD.*Groupname* tag

When determining the output file for routing-by-sheets, DMGMERGE uses the values it detects for the *first* eligible file. When determining segmentation, DMGMERGE also uses the values it detects for the *first* eligible file. DMGMERGE uses these values *before* it selects the final output file. This means that you need to

- Specify the same LRECL for all *eligible* files used in processing each Package.
- Specify the same CHAIN, MERGEDEF, PRINTDEF, and FGRPDEF for all *eligible* files used in processing each Package.

The Processing Sequence for Routing-by-Sheets with Segmentation

If you specify Routing-by-Sheets and Segmentation, DMGMERGE processes the Merge Sets in the following sequence:

(1) Selects the Routing-by-Sheets file from the eligible list of files, based on the actual number of sheets and the MAXSHEETS parameter.

(2) If this selected name is a master segment (ends in one or more percent signs) then use the current segment defined for the file.

For example, if the DDNAME parameter has this value:

```
DDNAME=(FILEA% FILEB FILEC%)
```

FILEA% and FILEC% are segmented files; FILEB is not.

Routing-by-Sheets Must Use the Same Printer Type

Thus, for a Document Package, you can combine Routing-by-Sheets and Segmentation as desired. However, because all output files eligible for routing-by-sheets require the same specifications, you cannot

- Use different printer types for the clean output file and the erred output file
- Mix printer types when routing-by-sheets
- Mix printer types when routing by errors

However, you can mix printer types for segmented file output. For example, you could use different PRINTDEFs to specify that an output file called SEGMENT-1 goes to a Metacode printer, and that an output file called SEGMENT-2 goes to an AFP printer.

Routing-by-Sheets, Overlay Forms, and Dynamic Forms Processing

DMGMERGE uses the values it detects for the *first* eligible file to complete the initial collate step. Uses those values *before* it selects the final output file, and then uses the values from the final clean output file to finish building the Document Package. Due to this sequence, DMGMERGE handles Overlays and dynamic forms as follows:

- For PHYSICAL- or LOGICAL-level Overlays defined in a FILEDEF, DMGMERGE uses the values from the first eligible file because PHYSICAL- and LOGICAL-level Overlays are build during collation, and these Overlays affect the total number of sheets produced.
- DMGMERGE processes SHEET-level Overlays during a sheet Overlay step, after the collate step. Thus, any SHEET Overlays defined in FILEDEFs come from the final clean file that DMGMERGE selects.

  Therefore, if an Overlay needs to vary based on the actual file selected with routing-by-sheets, use SHEET Overlays.
- DMGMERGE takes TAG parameters defined in FILEDEFs from the final clean file it selects for BPSD processing. However, for processing dynamic forms such as Docuword or Dynacomp forms with Variable Space Definitions (VSDs), DMGMERGE uses the first eligible clean file because dynamic forms are built during the collate step, when the final clean file is not yet known.

CAUTION!

Because Documerge processes a dynamic form using the first eligible clean file (the final clean file is not yet known), those Documerge reserved tags for Merge Set page counts and form counts might not contain accurate data.

For more information about the reserved tags that won't work with with dynamic forms, see "Limitations on the Use of Reserved Tags when Processing Dynamic Forms" on page 291.

If a dynamic form uses a TAG parameter value from a FILEDEF, it will use the one from the first clean file, not the final clean file. The exceptions are *dynamic sheet Overlays and dynamic forms used as sheet overlays*, for which DMGMERGE will use TAG parameters from the final clean file.

The FILEDEF Command and Routing-by-Sheets (MAXSHEETS)

For use with its Routing-by-Sheets feature, Documerge 3.1 and later releases let you specify output files with a list of DDnames separated by blanks. You can specify a file list in any of the following:

- DDNAME= parameter in the MERGE command — when listing more than one DDname, put the entire list of names in parentheses. For only one DDname, parentheses are optional.

  For details about the DMGMERGE SYSIN, see "SYSIN" on page 380.

- DMG.DD.*Groupname* tag in the VRF — do not use parentheses, but code the DDnames in sequence with at least one blank separating each DDname.

  For example, the DMG.DD.*Groupname* tag might contain the following DD names:

  ```
  INSC0050 INSC0100 INSCREST
  ```

  For details about the DMG.DD.*Groupname* tag, see "DMG.DD.Groupname" on page 305.

- **NEWDD=** parameter in the Documerge DDRENAME command — for details, see "The DDRENAME Command" on page 384.

DMGMERGE will use the first DD in the file list that matches the selection criterion specified by the MAXSHEETS parameter in the FILEDEF. Therefore, you must list the DDnames in ascending (smallest to largest) order, based on the number of sheets specified by the associated MAXSHEETS parameters.

For example, suppose you need three files for a Group's output Document Packages:

1. for 50 or less sheets
2. for 51 to 100 sheets
3. for over 100 sheets

You could code the following DD JCL statements, FILEDEFs, and MERGE commands:

```
//INSC0050  DD  ...
//INSC0100  DD  ...
//INSCREST  DD  ...
//SYSIN DD *
FILEDEF NAME=INSC0050  MAXSHEETS=50
FILEDEF NAME=INSC0100  MAXSHEETS=100
MERGE GROUP=INSURED -
DDNAME=(INSC0050 INSC0100 INSCREST) -
-rest of MERGE parameters here-
```

Remember that if there is no MAXSHEETS parameter coded for a FILEDEF or if there is no FILEDEF coded for a file, then there is no selection criterion for that file. Therefore, its DDname would always get selected unless another MAXSHEETS value met the test. In the above example, any Document Package with more than 100 sheets of paper is sent to `INSCREST`.

When specifying the output files to use with the MAXSHEETS, follow these rules:

- All files listed in a DDname or a DMG.DD.*Groupname* tag or a DDRENAME command must use the same MERGDEF, PRINTDEF, FGRPDEF (Font Group Definition for Metacode only) and CHAIN name.

  Documerge will use the MERGEDEF, PRINTDEF, and CHAIN from the first DD listed in the DDNAME or DMG.DD.*Groupname* tag or DDRENAME command for all subsequent DDs.

- Documerge selects a DD based on the criterion provided by MAXSHEETS, but you cannot use this to route output to different device types.

  For example, it is NOT possible to write to an IBM 3820 if there are 10 sheets or less in a Document Package, and also to a Xerox 4090 if there are more than 10 sheets in the same Document Package. All DDs in the list must be for the same device type, and must have the same specifications because Documerge uses the same MERGDEF, PRINTDEF, CHAIN, etc., specified for the first physical file in the clean (not error) DD list.

  If Documerge selects none of the DDs because the selection criteria is not met on any of the DDs in the list, then it routes the Document Package to the error stack with a error message indicating that no clean DD could be selected. **You can use this feature to force an error when a Document Package contains too many sheets of paper.**

- You can code a list of DDnames using the DDRENAME command, rather than coding the names with the DDNAME= parameter.

  Your existing VDRs can continue to generate a single DDname which is mapped to the list of DDnames specified by the DDRENAME command. For details, see "The DDRENAME Command" on page 384.

The FILEDEF Command and Routing-by-Errors

Documerge lets you specify an erred output file (error DD) for a Document Package, based on the error message(s) generated when that Document Package processed. Each erred output file can then be network-routed to the workstation or satellite printer for the appropriate person or department with the responsibility for correcting the error.

Using a combination of the ERRNUM parameter, and the ERRDDN parameter or DMG.ERRDD.*Groupname* reserved tag, you can specify

- **The errors that are significant,** by coding those error numbers in the ERRNUM list.
- **The correction priority for the errors,** by coding a sequence of error numbers and/or error files that represent the correction priorities for the errors.
- **A catch-all error DD for all less important errors,** by coding an error file for those error numbers not specified in an ERRNUM parameter.

Note that regardless of how you implement routing-by-errors, **Documerge will write an erred Document Package to only one file** — the first FILEDEF for which the ERRNUM selection criteria are met. Therefore, Oracle recommends that you associate at least one error DD file with the most important correctable error(s).

Routing-by-Errors Coding Examples

The following coding examples illustrate how to combine ERRNUM and ERRDDN specifications to implement routing-by-errors.

Example-1. DMGMERGE SYSIN coding to implement routing-by-errors for two error message numbers and one catch-all for other errors:

```
FILEDEF  NAME=INSMISSF  ERRNUM=(410 250)
FILEDEF  NAME=INSBDATA  ERRNUM=(355 360 540 352)
MERGE       GROUP=INSURED -
            DDNAME=INSURED -
            ERRDDN=(INSMISSF INSBDATA INSERROR)
   .
   .
```

The code shown in Example-1 specifies that

- If DMGMERGE generates error message 250 or 410, then write the Document Package to the INSMISSF error DD defined by the ERRDDN parameter.
- If DMGMERGE did not generate messages 250 or 410, but did generate message 352, 355, 360, or 540, then write the Document Package to the INSBDATA error DD defined by the ERRDDN parameter.
- If DMGMERGE did not generate any of the ERRNUM-specified messages, then write the Document Package to the INSERROR error DD defined by the ERRDDN parameter.

Example-2. DMGMERGE SYSIN coding to implement routing-by-errors for two levels of error message significance without a catch-all for other errors:

```
FILEDEF  NAME=INSMISSF  ERRNUM=(250 410)
FILEDEF  NAME=INSBDATA  ERRNUM=(352 355 360 540)
MERGE       GROUP=INSURED -
            DDNAME=INSURED -
            ERRDDN=(INSMISSF INSBDATA)
   :
   :
```

The code shown in Example-2 specifies the same routing-by-errors as Example-1, except that because no catch-all error DD is specified by the ERRDDN parameter, DMGMERGE will

- Write a C-level error message indicating that no error DD could be found.
- Write the erred Document Package to the last DD in the ERRDDN specified list (INSBDATA in this case).

NOTE

Do not use the error message for missing error DDs in any ERRNUM tests. If specified in code, Documerge will ignore this message because it generates only when Documerge does not find an error DD that matches with ERRNUM values.

How to Code the Number Values for the ERRNUM Parameter

Each Documerge error message has an identifier with the format:

`DMGMRGnnns`

Where **nnn** is a placeholder for the 3-digit error message number (**s** is a placeholder for the severity level).

For an alphabetical listing of the error message identifiers and descriptions of the error messages, see the contents pages in *Documerge Error Messages*.

Using a space and the hyphen continuation character, you can code the ERRNUM parameter in several different formats. Examples:

```
ERRNUM=(nnn nnn nnn ...)
```

-or-

```
ERRNUM=(nnn -
nnn -
:
:
nnn)
```

When coding the ERRNUM parameter, use the following rules:

- Each nnn value must be the 3-digit numeric part of a Documerge error message identifier.
- Each value must be 3 digits, including any leading zeros.
- The minimum value for each `nnn` is 100 and the maximum value is 999. The sequence values of the error numbers do not have to be coded in order.
- To continue a list on more than one record, code a space on the current record that is not in quotes, and then code a hyphen. Start the new record, and if desired, insert leading spaces before continuing the list.
- The parentheses are required if you code more than one value.

For more details about the ERRNUM parameter, see "ERRNUM=" on page 411.

Coding the Error DD File Name Values

In addition to specifying the error message number(s), you must also specify the file(s) to which Documerge will route erred Document Packages by coding the specification for the file(s) in any of the following:

- ERRDDN parameter of the MERGE statement. For details, see "ERRDDN=" on page 437.
- DMG.ERRDD.*Groupname* Reserved BPSD tag. For details, see "DMG.ERDD.Groupname" on page 306.
- NEWDD parameter of the DDRENAME command. For details, see "The DDRENAME Command" on page 384.

NOTE

You cannot use variable DDnames for error DDs. Variable DDnames are used with output segmentation, and are only for clean output.

The FILEDEF Command and Output Segmentation

Documerge gives you the ability to break up one large output file into multiple files. We call this *segmentation*. Each of the multiple files is a file segment.

While DMGMERGE is still running, you can use segmentation to

- Start printing a completed file segment; or, start transmitting a completed segment to a remote printer.
- Free up a tape drive that has processed segment of the output, or keep a tape from going to more than one volume (sometimes XEROX offline printing does not support multiple tapes).
- Release a completed segment so a post-processing program can read it.

How to Specify Segmented Output

NOTE

The following procedure describes the parameters that define segmentation. Some of these parameters work only when coded for the master segment, others work only when coded for one or more of the real segments, and some work when coded for the master and/or real segments.

If a parameter does not apply for given type of FILEDEF, Documerge still checks it for coding validity, but otherwise ignores the parameter.

For example, Documerge would ignore a MAXSEGMENTS=5 parameter you coded on a real segment FILEDEF, and would issue no warning. However, if you code MAXSEGMENTS=X for a real or master segment FILEDEF, Documerge issues an error message saying the value is not numeric.

1 Code FILEDEF for the master file.

a **Code the required NAME parameter** — this standard eight-character filename must end with one or more percent signs. The percent signs are placeholders that DMGMERGE replaces with sequential digits to form the real segment filenames.

For example, you could define the following master segment:

```
NAME=FRED%%
```

Then for the FRED%% master segment, DMGMERGE would generate the following real segment filenames:

```
FRED01, FRED02, FRED03, ... FRED09, FRED10, FRED11, ... up to FRED99
```

For details about the NAME parameter, see "NAME=" on page 407.

b **Code the required MAXSEGMENTS parameter** — the value of this parameter specifies the maximum number of segmented real files to create. **Code MAXSEGMENTS only in the master segment FILEDEF.**

For example, if **FRED%%** is the master segment name, you might not want the 99 possible segments, but only a maximum of 12 segments. Your coded MAXSEGMENTS parameter would look like this:

```
MAXSEGMENTS=12
```

The value for MAXSEGMENTS needs to be compatible with the number of percent signs coded for the master segment filename.

For example, if FRED%% is the master segment name, the MAXSEGMENTS value should not be greater than 99, because FRED%% contains only 2 percent signs, which can be replaced with the values 01 to 99.

For details about the MAXSEGMENTS parameter, see "MAXSEGMENTS=" on page 412.

For an example of MAXSEGMENTS coding, see the next topic.

c (**Optional) code one or more SEGMENT parameters** — the value of this parameter specifies the *number* and the *type of output units* (SHEETS, PAGES, PACKAGES, RECORDS, or BYTES) that Documerge must attain or exceed before creating an additional real file segment.

For example, if you specify the following SEGMENT parameter:

```
SEGMENT=(1000 SHEETS)
```

DMGMERGE will close the FRED01 real segment file, and open and start writing to the next sequential FRED02 real segment when the output to FRED01 exceeds 1000 sheets of paper and has reached the Merge Set boundary.

IMPORTANT!

DMGMERGE segments only on Merge Set boundaries. This ensures that multiple copies of the same Document Package will go to the same real file, and that multiple Groups in the same Merge Set defined for the same output file will go to the same real file.

Referring to the preceding example, if you specified segmentation after 1000 sheets of paper, DMGMERGE will detect when output has reached 1000 sheets, and it will also detect if it is still processing a Merge Set that exceeds 1000 sheets. In this case, DMGMERGE waits until it finishes processing the current Merge Set.

The result is that `FRED01` might actually contain more than 1000 sheets.

SEGMENT parameter coding guidelines — you can code values for this parameter in the master segment FILEDEF and/or in the FILEDEF for each real segment. Here are some guidelines for coding SEGMENT parameters:

- If all real segments have the same segmenting criteria, for convenience, code SEGMENT parameters in the FILEDEF for the master segment.
- If one or more real segments has different segmenting criteria, code the SEGMENT parameter in the FILEDEFs for those real segments.
- If you code SEGMENT parameters with different segmenting criteria in the FILEDEFs of both the master segment and a real segment, DMGMERGE combines those segmenting criteria.

  For example, if the SEGMENT parameter for the master segment specifies 1000 SHEETS, and the real segment specifies 50 PACKAGES, both criteria apply.
- If you code SEGMENT parameters with different numbers but with the same type for both the master segment and a real segment, DMGMERGE uses the segmenting criteria coded for the real segment.

  For example, if you specify 500 SHEETS for a real segment and 1000 SHEETS for a master segment, DMGMERGE will open the next real segment after 500 sheets, because you coded the same type (SHEETS in this example) for both the master and a real segment.

For details about the SEGMENT parameter, see "SEGMENT=" on page 415. For examples of output segmentation coding, see the next topic.

d (**Optional) code the DEALLOCATE resource management parameter** — you can specify this parameter to remove a real segment file from Documerge processing resources after the file closes. Documerge issues a **SVC99** system message to free print, disk, or tape files to be printed or processed by other programs. **Because deallocating the file frees it so that it can be printed or post-processed, you will usually need to perform this step.**

For details, see "DEALLOCATE=" on page 411.

Resource management parameter coding guidelines — you can code values for DEALLOCATE in the master segment FILEDEF and/or in the FILEDEFs for one or more of the real segments. If you code DEALLOCATE for a master segment, but not for a real segment that belongs to the master segment, then Documerge applies the master segment value to the real segment.

For an example of DEALLOCATE coding, see the next topic.

e (**Optional) code other FILEDEF parameters** — a master segment FILEDEF can specify any of the following parameters:

- BANNER
- TRAILER
- PACKAGE
- MAXSHEETS
- ERRNUM

FILEDEF optional parameter coding guidelines — you can code values for the above-listed parameters in the master segment FILEDEF and/or in the FILEDEFs for one or more of the real segments.

For example, you can specify BANNER and TRAILER pages in the master file that DMGMERGE will duplicate for each associated real segment, or you can specify different BANNER and TRAILER pages for each real segment.

If DMGMERGE finds more than one value (or no value) for a parameter, it uses the

following heirachy to determine which value to execute:

(1) The value coded in the FILEDEF of a real segment

(2) The value coded in the FILEDEF of the master segment

(3) The DMGMERGE default value

2 (Optional) **code real segment FILEDEF(s).**

a **Code the required NAME parameter** — for details, see step 1a of this procedure.

b (**Optional) code one or more SEGMENT parameters** — for details, see step 1c of this procedure.

c (**Optional) code the DEALLOCATE resource management parameter** — for details, see step 1d of this procedure.

d (**Optional) code other FILEDEF parameters** — a real segment FILEDEF can specify any of the following parameters:

- BANNER
- TRAILER
- PACKAGE
- MAXSHEETS
- ERRNUM
- OPENOUT

For details about FILEDEF optional parameters, see "" on page 407.

For more information about coding optional FILEDEF parameters, see "FILEDEF optional parameter coding guidelines — you can code values for the above-listed parameters in the master segment FILEDEF and/or in the FILEDEFs for one or more of the real segments." on page 426.

3 Specify the master segment filename in the same place that you normally request the output file in a DMGMERGE run, which is in one of the following:

- The **DDNAME** or **ERRDDN** in the MERGE statement. For details, see "DDNAME=" on page 437.
- The **DMG.DD**.*Groupname* or **DMG.ERDD**.*Groupname* Reserved Tag. For details, see "DMG.DD.Groupname" on page 305.
- The **NEWDD** value of a **DDRENAME** command For details, see "The DDRENAME Command" on page 384.

TIP

DDRENAME is a convenient way to convert to using segments if the DD name is in the DMG.DD.*Groupname* (or DMG.ERDD.*Groupname*) tag — you will not have to change the VDR coding for the original files.

4 Specify the real segment names in DD JCL statements. Each real segment must have a unique DDname, and you need to code a DD for each possible real segment.

Output Segmentation Coding Examples

Example-1. The following listing contains nine real segments for an Insured Group master file. Segmentation occurs after 100 Document Packages. The listing shows only the JCL for the real segment DDs, not the other required JCL.

```
//INSSEG1 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG2 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG3 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG4 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG5 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG6 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG7 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG8 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSSEG9 DD SYSOUT=M,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//SYSIN DD *
FILEDEF NAME=INSSEG% -
        SEGMENT=(100 PACKAGES) -
        MAXSEGMENTS=9 -
        DEALLOCATE=Y
MERGE GROUP=INSURED -
      DDNAME=INSSEG% -
      MERGEDEF=9700 -
      ERRDDN=INSERR
//
```

Example-2. The following listing shows the coding required to create up to three segments for a first Insured Group master file, and the coding required to create up to five segments for a second Insured Group master file.

The MAXSHEETS parameter coded in the Example-2 listing also shows how output segmentation can be combined with routing-by-sheets.

DMGMERGE segments the first master file every 300 Document Packages, and it segments the second master file every 150 Document Packages. The listing shows only the JCL for the real segment DDs, not the other required JCL.

```
//INSA1 DD SYSOUT=X,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSA2 DD SYSOUT=X,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSA3 DD SYSOUT=X,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSB1 DD SYSOUT=Y,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSB2 DD SYSOUT=Y,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSB3 DD SYSOUT=Y,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSB4 DD SYSOUT=Y,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//INSB5 DD SYSOUT=Y,DCB=(BLKSIZE=3000,LRECL=155,RECFM=VBM)
//SYSIN DD *
FILEDEF NAME=INSA% -
        MAXSHEETS=100
        SEGMENT=(300 PACKAGES) -
        MAXSEGMENTS=3 -
        DEALLOCATE=Y
FILEDEF NAME=INSB% -
        SEGMENT=(150 PACKAGES) -
        MAXSEGMENTS=5 -
        DEALLOCATE=Y
MERGE GROUP=INSURED -
      DDNAME=(INSA% INSB%) -
      MERGEDEF=9700 -
      ERRDDN=INSERR
//
```

The FILEDEF Command and BTEXT-Generated Bar Codes for Xerox 4635 Printers

The Xerox BTEXT feature lets you determine if any pages were not sent to the printer by performing a reconciliation audit that compares your user-supplied page total with a count of the pages actually printed.

If your printer is a Xerox 4635 or similar printer, you can specify the FILEDEF BTEXTINIT parameter to request that Documerge do the following:

- Write BTEXT DJDE records for a FILEDEF-defined Document Package.
- Print a bar code on the last page of the Document Package using data taken from a reserved tag, sheet Overlay, and BTEXT generated DJDE records.

Guidelines for Generating BTEXT DJDEs with the BTEXTINIT Parameter

When specifying BTEXT processing with the BTEXTINIT parameter, use these guidelines:

- Each Documerge implementation of BTEXT processing has a maximum capacity of 65534 sheets per file — the Xerox limit. Therefore, for BTEXT processing Oracle recommends use of the Documerge page segmentation (e.g., SEGMENT=(60000 SHEETS). For an explanation of the segmentation feature, see "" on page 424).
- The BTEXT DJDE RRA (total of all previous PRA values used to validate that all pages of a Document Package were printed) is the computed reconciliation amount. Xerox requires that the RRA and NSE (number of sheets expected) subparameters reside in the last DJDE for a file. Documerge can only process the last DJDE and place a BTEXT RRA and NSE into the DJDE if a trailer is specified for the Document Package. Therefore, to get the final BTEXT RRA and NSE values, you must specify a trailer form. Documerge will not generate an error if a trailer is not specified.
- The trailer form does not have to contain the DMG.BTEXT.SEQ reserved tag, but if the trailer does have DMG.BTEXT.SEQ, then DMGMERGE generates a RBAR, BTEXT SEQ= DJDE, and bar code for the trailer, as it does for any form that contains DMG.BTEXT.SEQ.
- You can also define a banner page to contain the DMG.BTEXT.SEQ tag. If the banner does have DMG.BTEXT.SEQ, then DMGMERGE generates a RBAR, BTEXT SEQ= DJDE, bar code, but no PRA= value for the banner.

Implementing Xerox BTEXT For DMGMERGE

Before starting to implement BTEXT for Documerge applications, Oracle recommends that you refer to the Xerox 4635 printer manual for information about BTEXT DJDE keywords and bar codes.

For processing with DMGMERGE, Some BTEXT features are required — others are optional. The following topics address all the BTEXT features that can be implemented for DMGMERGE, starting with those features which are required.

For instructions about	See this topic
Coding the required BTEXTINIT parameter to get DMGMERGE to print the initial BTEXT DJDE values for a Document Package.	"To Code the BTEXTINIT Parameter" on page 431
Defining the required RBAR string that triggers BTEXT processing.	"To Define the RBAR String" on page 431
Coding the required DMG.BTEXT.SEQ reserved tag that triggers DMGMERGE to generate the BTEXT bar code, the RBAR text string, and the SEQ= parameter value.	"To Code the DMG.BTEXT.SEQ Reserved Tag" on page 432
Coding the optional OTEXT parameter that specifies a message to display on the Xerox operator's console, and/or suspends printing so that operator can perform bar code reader adjustment or other special preparation.	"" on page 433
Implementing the optional PRA parameter that contains a check-sum value which is written to the BTEXT DJDE and used to validate that all pages of a Document Package were printed.	"To Implement the PRA Parameter" on page 433
Implementing the optional TXT parameter that provides information that is helpful when reading the reconciliation report generated by BTEXT processing.	"To Implement the TXT Parameter" on page 433
Specifying that DMGMERGE generate the RRA (total of all previous PRA values) and NSE (number of sheets expected) values in the final BTEXT DJDE for a Document Package. These values are used to verify that all the pages of a Document Package were correctly printed.	"To Specify that DMGMERGE Generate the RRA and NSE Values" on page 433
Coding the optional (depending on your environment) BTEXTNSEADD parameter to adjust the NSE (number of sheets expected) value that DMGMERGE computes so that it also includes JES banner or trailer pages counted by Xerox.	"To Code the BTEXTNSEADD Parameter" on page 434
Making the other preparations required before DMGMERGE can perform BTEXT processing.	"To Prepare for BTEXT Processing" on page 434

To Code the BTEXTINIT Parameter

1 Update the DMGMERGE SYSIN with FILEDEFs for all the clean files (Document Packages) that need BTEXT processing.

IMPORTANT!

Documerge does BTEXT processing only for error-free (clean) Document Packages — not erred files. Do not specify the BTEXTINIT parameter for error files. The BTEXT SEQ values will not be correct for error files. If you write both clean and erred Document Packages to the same output file, the BTEXTINIT parameter will not work properly.

2 In each FILEDEF, code the BTEXTINIT parameter and its value, which consists of one or more of the standard Xerox BTEXT parameters such as PRD= (report date), DPT= (department), etc.

DMGMERGE uses the BTEXTINIT value to write the initial BTEXT DJDE records for the associated Document Package. All parameters are optional except the **RRA=** subparameter — **Xerox requires RRA= for BTEXT processing**.

For example, the following statements specify the BTEXTINIT parameter and subparameters in the FILEDEF for the INSCLEAN Document Package:

```
FILEDEF NAME=INSCLEAN -
          :
          :
         BTEXTINIT='RRA=INSURED,CJN=1,PRD=%%DATE%%'
```

For details about the BTEXTINIT parameter, see "BTEXTINIT=' '" on page 408.

To Define the RBAR String

1 Define a new PRINTDEF for the Xerox 4635 or other BTEXT enabled printer.

If you have an existing Xerox PRINTDEF, you can rename and modify a copy of it to get the new PRINTDEF for BTEXT processing.

2 Code the RBARFLAG and RBAROFF parameters for the PRINTDEF.

For example, the following statements specify the RBARFLAG and RPBAROFF parameters in a PRINTDEF for the Xerox 4635 printer:

```
PRINTDEF NAME=4635MET -
           :
           :
            RBARFLAG=X'01D9C2C1D9' - Hex 01 followed by "RBAR"
            RBAROFF=0 -
```

For details about the RBARFLAG and RBAROFF parameters, see "RBARFLAG=" on page 103, and "RBAROFF=" on page 103.

3 To catalog the new PRINTDEF for DMGMERGE, run the DPLDUTL utility.

IMPORTANT!

Ensure that you omit the MERGEDEF and PRINTDEF from the MERGE command so that DMGMERGE will use the PRINTDEF you specified in the FILEDEF.

After encountering the RBAR string, DMGMERGE will generate an RBAR record for any page that has a DMG.BTEXT.SEQ reserved tag.

To Code the DMG.BTEXT.SEQ Reserved Tag

1 In the form or forms (i.e., Overlay, trailer page, or other form) for which you want DMGMERGE to generate the BTEXT bar code, RBAR, and/or SEQ= DJDE records, code a DMG.BTEXT.SEQ BPSD with a maximum length of five characters.

You can code the DMG.BTEXT.SEQ BPSD in as many forms for as many pages in a Document Package as you desire. Or, you can omit this BPSD for all of the forms in a Document Package (all Merge Sets do not have to request BTEXT processing).

Although you can code the DMG.BTEXT.SEQ BPSD for any type of form (such as a check form), **Oracle recommends coding this BPSD in an Overlay** so that if you need to change the location of the bar code, you only have one Overlay form to change.

For example, if you use a composition system to code the DMG.BTEXT.SEQ BPSD correctly in a form, the procedure would consist of the following:

a Position the BPSD using the composition positioning commands.

You must code the DMG.BTEXT.SEQ BPSD so it prints the bar code in the correct place with the correct font for processing by the printer bar code scanner.

b Specify the font for the BTEXT bar code.

c Enter the BPSD definition: `BPSD NAME='DMG.BTEXT.SEQ' LENGTH=5.`

d As part of the form text, Code any asterisks before and after the BPSD.

According to Xerox, the bar code sequence number starts and ends with an asterisk.

For details about the DMG.BTEXT.SEQ reserved tag, see "DMG.BTEXT.SEQ" on page 295.

2 Load the form composed in step 1 to the EDL with the name *BTEXT.OVL* or other user-defined name of your choosing.

3 Specify the BTEXT.OVL or other form containing the DMG.BTEXT.SEQ reserved tag for each Document Package to receive BTEXT processing.

If you want only selected pages (based on DTN) of the Document Package to receive BTEXT processing, you can specify the Overlay(s) in the Rulebase. If all the pages receive BTEXT processing, it is simpler to specify a sheet Overlay in the FILEDEF.

For example, the following FILEDEF statement specifies BTEXT.OVL as a sheet Overlay:

```
OVL=(BTEXT.OVL SHEET)
```

For details about specifying sheet Overlays, see "xVL=( )" on page 401.

To Code the OTEXT Parameter

For those FILEDEFs that require operator intervention for bar code reader adjustment before BTEXT processing, add the OTEXT parameter that specifies a message to display on the operator's console, and/or suspends printing until the operator restarts the printer.

You can code only one OTEXT parameter for each FILEDEF.

For example, the following bolded statement specifies the OTEXT parameter in the FILEDEF for the INSCLEAN Document Package:

```
FILEDEF NAME=INSCLEAN -
         :
         :
         BTEXTINIT='RRA=INSURED,CJN=1,PRD=%%DATE%%'
         OTEXT=('--> SETUP FOR DOCUMERGE BTEXT <--'),WAIT)
```

For details about the OTEXT parameter, see "OTEXT=( )" on page 414.

To Implement the PRA Parameter

1. Define the DMG.PRA and DMG.PRA.*Groupname* reserved tags in a Rulebase Tag table, or the VDR can write these tags with the DMGVRFWR program.
2. Just as you would do for any other reserved tag, program the VDR to generate either the DMG.PRA or DMG.PRA.*Groupname* reserved tag, and then write the tag to the VRF.

   For details about these parameters, see "DMG.PRA" on page 320, or "DMG.PRA.Groupname" on page 320.

   DMGMERGE will total the PRA values for each Merge Set and write the RRA value in the final BTEXT DJDE that is used as a check sum to validate that all pages have printed.

To Implement the TXT Parameter

1. Define the DMG.TXT and DMG.TXT.*Groupname* reserved tags in a Rulebase Tag table, or the VDR can write these tags with the DMGVRFWR program.
2. Just as you would do for any other reserved tag, program the VDR to generate either the DMG.TXT or DMG.TXT.*Groupname* reserved tag, and then write the tag to the VRF.

   For details about the DMG.TXT or DMG.TXT.*Groupname* reserved tags, see "DMG.TXT" on page 324 or "DMG.TXT.Groupname" on page 324.

To Specify that DMGMERGE Generate the RRA and NSE Values

To generate the RRA (total of all previous PRA values) and NSE (number of sheets expected) values in the final BTEXT DJDE for a Document Package, specify a trailer page, preferably with the TRAILER= parameter for the FILEDEF.

The user-defined trailer page signals the occurrence of the final form in the Document Package, and DMGMERGE will automatically compute and write the RRA and NSE BTEXT DJDE parameters to print on the trailer. (You do not have to code the DMG.BTEXT.SEQ tag for the trailer page.)

For example, the following bolded TRAILER parameter specifies that the EDL member named INSCLEANTR (revision 2) is to be used as the trailer:

```
FILEDEF NAME=INSCLEAN -
          :
          :
          BTEXTINIT='RRA=INSURED,CJN=1,PRD=%%DATE%%'
          TRAILER=INSCLEANTR(2)
```

For details about user-defined trailer pages, see "TRAILER=" on page 417.

For more information about the TXT= parameter, see "To Implement the TXT Parameter" on page 433.

To Code the BTEXTNSEADD Parameter

If you need to adjust the DMGMERGE computed NSE (number of sheets expected) value because Xerox BTEXT internal processing counts JES banner or trailer pages, code the BTEXTNSEADD parameter for the FILEDEF command.

For example, the following bolded BTEXTNSEADD parameter specifies that 3 pages are to be added to the NSE value:

```
FILEDEF NAME=INSCLEAN -
          :
          :
          BTEXTINIT='RRA=INSURED,CJN=1,PRD=%%DATE%%'
          OTEXT=('--> SETUP FOR DOCUMERGE BTEXT PROCESSING NOW <--'),WAIT)
          TRAILER=INSCLEANTR(2)
          BTEXTNSEADD=3 -
```

For details about BTEXTNSEADD, see "BTEXTNSEADD=" on page 409.

To Prepare for BTEXT Processing

1. For the Xerox printer JDE/JDL, you need to code any required BTEXT parameters, such as RBAR.

   See your Xerox manual for JDE/JDL requirements.

2. Check that the bar code reader is aligned correctly.

   See your Xerox manual for alignment instructions.

When you run DMGMERGE, your output should contain the BTEXT DJDE record, the bar code, and the RBAR record.

The MERGE Command

The MERGE command starts the DMGMERGE program. One MERGE command is mandatory, and you can code more.

MERGE Parameters

Parameter	Value
BANNER=	Optional. The name of the form used as a banner sheet for the Group. **NOTE:** The BANNER= parameter overrides any value in the VRF for the DMG.BNR.*Groupname* Reserved Tag. You can use this parameter to print a banner sheet that's specialized for the Group. (The FILEDEF command's BANNER= parameter prints a banner sheet for the output file you define with the FILEDEF command.) You can also use this parameter to print a banner sheet for an error file and for a clean file. (These error files are defined in the MERGE command's ERRDDN= parameter or the VRF's DMG.ERDD.*Groupname* Reserved Tag.) Valid values are: **DEFAULT** The Documerge banner sheet, with asterisks. This is the BANNER= default value for clean DDs. **NONE** No banner sheet. This is the BANNER= default value for error DDs. **NOTE:** The MERGEDEF parameter MSGCTLx contains DJDEs that are placed at the beginning of the output file. This initial DJDE packet is separate from the DJDE packet in a form. Xerox processing requires printable characters between DJDE packets; otherwise, the packet in the first form of the output file is ignored. This condition can cause the output file to be unprintable. Therefore, if BANNER=NONE is specified for a clean Metacode file, DMGMERGE places a blank sheet before the first form in the output file. For an error DD, a blank sheet is not needed because the messages follow the initial DJDE packet and precede the first form. **formname** The name of the EDL member you want to use as the banner sheet. This value automatically specifies revision 0. If you want a banner sheet named DEFAULT or NONE, you must specify the revision level.
	NOTE: The MERGEDEF parameter MSGCTLx contains DJDEs that are placed at the beginning of the output file. This initial DJDE packet is separate from the DJDE packet in a form. Xerox processing requires printable characters between DJDE packets; otherwise, the packet in the first form of the output file is ignored. This condition can cause the output file to be unprintable. Therefore, if BANNER=formname is specified for a clean Metacode file, DMGMERGE places a blank sheet before the banner sheet. For an error DD, a blank sheet is not needed because the messages follow the initial DJDE packet and precede the banner sheet. **formname(revision)** The name and revision level of the EDL member you want to use as the banner sheet.

Parameter	Value
BANNERFEED=	(Optional) specifies the input tray name for user banner forms. Format: BANNERFEED=*trayname* Replace the *trayname* placeholder with one of the following: ▪ Xerox Metacode cluster name ▪ IBM AFP COPYGROUP (IMM) name, including the "C" prefix The default value is MAIN. To avoid printing errors, you must ensure that the selected cluster or COPYGROUP name exists. This parameter is ignored for the DMGMERGE default banner page and for line printer output. If you specify BANNERFEED in both the FILEDEF and MERGEDEF, the MERGEDEF takes precedence.
BVL=( )	(Optional) specifies a **Back Overlay** for all the even forms — except the banner and trailer forms — processed and written for the Group specified in the MERGE command. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.
CHAIN=	(Optional) the chain name to be used when selecting EDL members. The default chain name is based on the value of the PDEV parameter value in the PRINTDEF. Valid values are: **name** The chain name. **(name1 name2 ...name5)** Up to 5 chain names. DMGMERGE uses the sequence you specify to search the chains in the EDL member. DMGMERGE uses the first of these chains that it finds. With this option, you can mix Template Technology forms and other forms (or normalized AFP forms and other AFP forms) and keep the different chain names.
CONDITIONAL=	(Optional) available with Documerge 3.1.1 and higher levels. This specifies conditional Group processing. A conditional Group is one that prints only if at least one control Group in the same Merge Set prints to the same file as the conditional Group. Values are: **N** or **NO** This Group does not participate in condition Group processing; this is the default. **Y** or **YES** This is a conditional Group. **C** or **CONTROL** This is a control Group. See "MERGE Parameters" on page 435.
COPIES=	Indicates number of copies of the Document Package printed for the specified Group. This parameter overrides any value in the VRF for the DMG.GCPY.*Groupname* Reserved Tag. You can specify as many as 32,767 copies. If a Merge Set is in error, this value is ignored and one copy is produced. The MAXCOPY parameter value in the PRINTDEF must be equal to or higher than the COPIES= value.

Parameter	Value
DDNAME=	Specifies the file name(s) of the JCL control statement(s) that define the file(s) to hold a Group's printable Document Packages. Instead of a single DDname, you can code a list of DDnames in sequence with at least one blank separating each DDname, and all the DDNames placed in parentheses. For example, you could code **DDNAME=** to specify the following DD names: `DDNAME=(INSC0050 INSC0100 INSCREST)` **NOTE:** The MERGE parameter **DDNAME=** overrides the DMG.DD.*Groupname* Reserved Tag specified in a VRF.
DMFPPP=	(Optional) creates a DMFPPP file that interfaces Documerge output with other Oracle processes. If another program requires a DMFPPP file, you must use Documerge 3.1.2 or higher and this parameter to create the file. Format: `DMFPPP=`*`ddname`* Replace the *ddname* placeholder with the output DD filename for the DMFPPP file.
ERRDDN=	Specifies the file name(s) of the JCL control statement(s) that define the file(s) to hold a Group's printable output in error. Instead of a single DDname, you can code a list of DDnames in sequence with at least one blank separating each DDname, and with all the DDNames placed in parentheses. For example, you could code **ERRDDN** to specify the following DD names: `ERRDDN=(INER0050 INER0100 INERREST)` **NOTE:** The MERGE parameter **ERRDDN=** overrides the DMG.ERDD.*Groupname* Reserved Tag specified in a VRF.
ERRMSG=	Indicates that DMGMERGE is to echo (repeat) any Merge Set error message on the Message File as well as to the ERRDDN for this Merge Set. Valid values are: ▪ **NO** ▪ **YES**
EXTRAPAGE	Indicates that DMGMERGE produces a blank page at the beginning of a Group. Note that this parameter does not use an equals sign (does not have a value, but acts as a switch when coded). Specify EXTRAPAGE only if you want a blank page. The default is no blank page. **CAUTION:** Use this option only when completed Document Packages do not begin on the odd side of a sheet. Otherwise, this causes the output to begin on the incorrect side.
FGRPDEF=	For Metacode data streams, indicates the name of the Font Group Definition PEDEF to be used during Tumble or Imposition processing. **NOTE:** The MERGE parameter FGRPDEF= overrides the DMG.FDEF.*Groupname* Reserved Tag specified in a VRF. The DMG.FDEF.*Groupname* Reserved Tag overrides the FGRPDEF PEDEF specified in the PRINTDEF PEDEF.
FVL=( )	(Optional) specifies a **Front Overlay** for all the odd forms — except the banner and trailer forms — processed and written for the Group specified in the MERGE command. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.

Parameter	Value
GROUP=	**Required.** Indicates the name of the Group as defined in the Rulebase Library Group Table.
LANDSEP=	Indicates in tenths of inches the amount of space to be placed between concatenated landscape images. The default is (0) inches.
MERGEDEF=	Indicates the name of the MERGEDEF used from the PELIB. **NOTE:** The MERGE parameter **MERGEDEF=** overrides the DMG.MDEF.*Groupname* Reserved Tag specified in a VRF.
OVL=( )	(Optional) specifies an **Overlay** for all the forms — except the banner and trailer forms — processed and written for the Group specified in the MERGE command. For detailed rules and coding examples, see "xVL=( )" on page 401. For an explanation of package-level Overlays, see "Specifying Overlays with the DMGMERGE Commands" on page 474.
PACKAGE=	For Metacode data streams, indicates whether DMGMERGE generates the RSTACK value at the end of each Merge Set. Allows stapling the Merge Set as a single Document Package. **N** DMGMERGE does not generate the RSTACK value. **Y** DMGMERGE generates the RSTACK value. **NOTE:** The Y value can cause significantly slower printing. After generating the RSTACK value, DMGMERGE must reissue all DJDEs in EBCDIC.
PORTSEP=	Indicates in tenths of inches the amount of space to be placed between concatenated portrait images. This is optional. The default is (0) inches.
PRINTDEF=	The PRINTDEF name used from the PELIB. **NOTE:** The MERGE parameter PRINTDEF= overrides the DMG.PDEF.*Groupname* Reserved Tag specified in a VRF. The DMG.PDEF.*Groupname* Reserved Tag overrides the PRINTDEF PEDEF specified in the MERGEDEF PEDEF.

Parameter	Value
RESERVEDRT	Causes data in the following Reserved Tags to be right justified, with leading zeros and no commas (some finishing equipment requires this format), when merged into the BPSD on the form. **Reserved Tag Name For More Information** DMG.CURRENT.SHEET.COUNT Refer to page 305 DMG.HEX8BCC.COUNTRefer to page 314 DMG.ITEM.COUNT.VERIFY Refer to page 315 DMG.LINE.COUNT Refer to page 315 DMG.POL.COUNT Refer to page 319 DMG.SHEET.COUNT Refer to page 321 DMG.SHEET.NUMBER Refer to page 321 DMG.SSI.COUNT Refer to page 323 DMG.TOTAL.SHEETS Refer to page 323 This parameter does not use an equals sign (does not have a value, but acts as a switch when coded). **TECHNICAL NOTE:** For line-printer printed dash codes, RESERVEDRT supports bit processing for all of the above-listed reserved tags except DMG.LINE.COUNT and DMG.POL.COUNT. If the tag value ends with **.BIT.n** where **n** is any number from 0 to 9, or ends with **.BIT.nn**, where **nn** is any number from 00 to 31, then Documerge returns one of two characters, depending if the value of the requested bit is zero or one. BIT.0 or BIT.00 means the last (rightmost) bit; BIT.1 or BIT.01 is the bit just to the left of the last (rightmost) bit; etc. The MERGEDEF specifies the characters to use, based on the value of the bit — the DASHCODE-ON (bit value one) and DASHCODE-OFF (bit value zero) parameters. RESERVEDRT is not used for any other bit processing. **TIP:** You can use command tag processing to print a reserved tag value with a different justification format. To do this, use a command tag name, and not the original reserved tag name. For more information, see "DMG.C.xxx Reserved Tag Processing" on page 331.
RTJUSTIFY=	Indicates that you want to right-justify a Reserved Tag. The format of this parameter is `RTJUSTIFY=DMG.SET.NUMBER` Specify this on any MERGE command to cause BPSDs that use the DMG.SET.NUMBER tag to be right-justified with this BPSD area. Remember that right justification is on a character basis and generally requires a fixed-pitch font for desired results.
SEP=	Specifies the same value for both the LANDSEP= and PORTSEP= parameters simultaneously. For details, see LANDSEP= and PORTSEP= earlier in this table.

Parameter	Value

TAG=	A BPSD tag name and its user-defined variable data. This allows for printing of unique variable data on forms specified for a Group.
	For example:
	`TAG=(BPSD.TAG.NAME 'VARIABLE DATA')`
	The tag name cannot contain blanks. Variable data containing blanks must be enclosed in single quotes, variable data containing single quotes must be coded as two single quotes.The total characters inside the parentheses must be 70 characters or less, including any quotes.
	DMGMERGE performs any ***Delete*** processing previously defined for the tag name.
	For the TAG= parameter, the tag's value may span multiple SYSIN records as follows:
	▪ After the last character of the first line of the value, code an ending single quote. (The tag value must be in single quotes.)
	▪ Code a blank/hyphen (standard SYSIN continuation characters).
	▪ On the next record, code a quote, and continue the value. The data following this quote butts up to the data ended by the quote in the preceding record.
	▪ To end the value, code a single quote followed by a right parenthesis (standard for the TAG= parameter).
	▪ If more parameters follow for this MERGE command, remember to follow the right parenthesis with a blank/hyphen as usual.
	Example:
	`TAG=(TAGNAME 'This is the value for this tag which'-`
	`'is so large that we must use three SYSIN records' -`
	`'to code all of it.')`
	The priority for searching and processing TAG= is as follows:
	1 TAG= parameter from MERGE command
	2 TAG= parameter from FILEDEF command
	3 TAG= parameter from GLOBAL command
	4 VRF tags
	5 Internally generated tags (such as DMG.PAGE.NUMBER and DMG.CURRENT.DATE)
	NOTE: Documerge places no limit on the number of TAG= parameters per MERGE command.

Parameter	Value
TRAILER=	The name of the form used as a trailer sheet for the Group. Optional. Use this parameter to print a trailer sheet that's specialized for the Group. (The FILEDEF command's TRAILER= parameter prints a trailer sheet for the output file you define with the FILEDEF command.) Use this parameter to print a trailer sheet for an error file and for a clean file. (Error files are defined in the MERGE command's ERRDDN= parameter or the VRF's DMG.ERDD.*Groupname* Reserved Tag.) Valid values are: **DEFAULT** No trailer sheet. **NONE** No trailer sheet. This is the TRAILER= default value for both clean and error DDs. **formname** The name of the EDL member to use as a trailer sheet. This value automatically specifies revision 0. If you want a trailer sheet named DEFAULT or NONE, you must specify the revision level. **formname(revision)** The name of the EDL member you want to use as the trailer sheet. This value includes a revision level that you select. **NOTE:** The MERGE command's TRAILER= parameter overrides any value in the VRF for DMG.TLR.*Groupname*.
TRAILERFEED=	(Optional) specifies the input tray name for user trailer forms. Format: `TRAILERFEED=`*`trayname`* Replace the *`trayname`* placeholder with one of the following: ▪ Xerox Metacode cluster name ▪ IBM AFP COPYGROUP (IMM) name, including the "C" prefix The default value is `MAIN`. To avoid printing errors, you must ensure that the selected cluster or COPYGROUP name exists. This parameter is ignored for the DMGMERGE default trailer page and for line printer output. If you specify TRAILERFEED in both the FILEDEF and MERGEDEF, the MERGEDEF takes precedence.

The MERGE Command and Conditional Groups

Documerge 3.1.1 and higher support conditional Groups. A conditional Group is a Group that will not print depending on another Group in the same Merge Set. For coding details, see "CONDITIONAL=" on page 436.

Example of Conditional Group Processing

Assume we are printing an insured's policy. If the policy has 10 pages or less, we want to fold the policy in half so that the first page is in the middle and last page is on the outside, and we want to place it into a small envelope. If the policy is more than 10 pages, we will not fold the policy and put it into a large envelope. This can be done using

- MAXSHEETS to dynamically select the output file
- Finishing equipment that handles a given stack of paper (an output file).

So far this does not require conditional Groups. But suppose we also want to include a name and an address sheet to show in a window in the envelope. The problem is that if we fold the policy in half, the address sheet must print last and on the back side. But if we are not folding the policy, we want the address sheet to print first and on the front side.

Printing for this output is done using conditional Groups. Instead of one Group, we will request three Groups in DMGMERGE:

- Front address sheet
- The policy itself
- Back address sheet

DMGMERGE SYSIN Example

We won't print all three Groups for any one Merge Set. DMGMERGE will use only one of the address sheet Groups and ignore the other one. Here is an example of the DMGMERGE SYSIN that does this:

```
GLOBAL ALLERROR=Y
FILEDEF NAME=BIFOLD BANNER='BIFOLD-INSTRUCTION' MAXSHEETS=10
FILEDEF NAME=NOFOLD BANNER='NOFOLD-INSTRUCTION'

MERGE GROUP=FRONT.ADDRESS          -
      CONDITIONAL=YES              -
      DDNAME=NOFOLD                -
      MERGEDEF=MT97                -
      PORTSEP=0I                   -
      CHAIN=META                   -
      ERRDDN=ERROR1                -
      RESERVEDRT                   -
      ERRMSG=YES
MERGE GROUP=MAIN.POLICY            -
      CONDITIONAL=CONTROL          -
      DDNAME=(BIFOLD NOFOLD)       -
      MERGEDEF=MT97                -
      PORTSEP=0I                   -
      CHAIN=META                   -
      ERRDDN=ERROR1                -
      RESERVEDRT                   -
      ERRMSG=YES
MERGE GROUP=BACK.ADDRESS           -
      CONDITIONAL=YES              -
      DDNAME=BIFOLD                -
      MERGEDEF=MT97                -
      PORTSEP=0I                   -
      CHAIN=META                   -
      ERRDDN=ERROR1                -
      RESERVEDRT                   -
      ERRMSG=YES
```

Here is what the above DMGMERGE SYSIN does:

GLOBAL ALLERROR=Y

This causes all Groups to be in error if any one Group is in error. But more importantly, it causes DMGMERGE to do two passes on each Group for a each Merge Set.

This allows a Group to be conditional depending on a subsequent control Group. In this specific example, we want Group FRONT.ADDRESS to be conditional depending on the MAIN.POLICY Group.

Because FRONT.ADDRESS is specified prior to MAIN.POLICY, we need ALLERROR=Y. We also need ALLERROR=Y to keep all Groups together on either the clean or error files, since it does not make sense to print an address sheet to the clean file and print the main policy to the error file.

FILEDEF NAME=BIFOLD BANNER='BIFOLD-INSTRUCTION' MAXSHEETS=10

This causes file BIFOLD to be used only when the Document Package contains 10 sheets or less (a Document Package being one copy for a Group).

Since we have only requested one copy, we can equate a "Document Package" with a "Group" in this particular example.

The BANNER= is a form we have coded and loaded into the EDL with instructions on handling this stack of paper. We could even add Overlays using the OVL, FVL and/or BVL parameters to print bar codes on this file.

FILEDEF NAME=NOFOLD BANNER='NOFOLD-INSTRUCTION'

This allows different banner instructions on file NOFOLD, since we want to handle it differently from file BIFOLD. Also, we did not code MAXSHEETS, since this file gets "all other sheets" -- all sheets that do not get routed to BIFOLD (we could have coded MAXSHEETS=99999999 for the same results.)

We could even add Overlays using the OVL, FVL and/or BVL parameters to print bar codes on this file.

```
MERGE GROUP=FRONT.ADDRESS          -
      CONDITIONAL=YES              -
      DDNAME=NOFOLD                -
      MERGEDEF=MT97                -
      PORTSEP=0I                   -
      CHAIN=META                   -
      ERRDDN=ERROR1                -
      RESERVEDRT                   -
      ERRMSG=YES
```

This requests the front address sheet. Because CONDITIONAL=YES, the Group will be printed only if at least one control Group also prints to the same output file for this Merge Set. In this specific example, because DDNAME=NOFOLD, we will write this Group only if the MAIN.POLICY Group writes to file NOFOLD.

```
MERGE GROUP=MAIN.POLICY            -
      CONDITIONAL=CONTROL          -
      DDNAME=(BIFOLD NOFOLD)       -
      MERGEDEF=MT97                -
      PORTSEP=0I                   -
      CHAIN=META                   -
      ERRDDN=ERROR1                -
      RESERVEDRT                   -
      ERRMSG=YES
```

This requests the main policy. CONDITIONAL=CONTROL means this is a control Group — one that controls the processing of conditional Groups. More than one Group can specify CONDITIONAL=CONTROL if desired.

```
MERGE GROUP=BACK.ADDRESS            -
      CONDITIONAL=YES               -
      DDNAME=BIFOLD                 -
      MERGEDEF=MT97                 -
      PORTSEP=0I                    -
      CHAIN=META                    -
      ERRDDN=ERROR1                 -
      RESERVEDRT                    -
      ERRMSG=YES
```

This is similar to the front address sheet, except we will write this Group to its output file BIFOLD only if the MAIN.POLICY also writes to file BIFOLD.

Guidelines for Conditional Group Processing

NOTE

You can use segmentation with conditional Groups. Segmentation is independent of routing-by-sheets or conditional Groups. By design, segmentation occurs only at Merge Set boundaries, so you are always guaranteed that when two or more Groups specify the same DDNAME value, they will go to the same output file.

- Specify ALLERROR=Y (not absolutely required, but strongly recommended).
- Use the routing-by-sheets feature: Code at least one FILEDEF using MAXSHEETS.
- Decide if a Group is to be
  - a control Group
  - a conditional Group
  - does not participate in conditional processing at all
- For control Groups (such as MAIN-POLICY in our example above)
  - Specify CONDITIONAL=CONTROL.
  - Specify multiple DD names in the DDNAME= parameter to have DMGMERGE select the clean output file based on the number of sheets in the Document Package
- For conditional Groups (such as FRONT.ADDRESS and BACK.ADDRESS in our example above):
  - Specify CONDITIONAL=YES
  - Specify only one value in the DDNAME parameter. It still is possible to use routing-by-sheets here also, but probably not desired.
- For Groups that do not participate in conditional processing, specify CONDITIONAL=NO, or do not specify any CONDITIONAL parameter.

- Error message DMGMRG397W indicates that a conditional Group was found with no previous control Group processed. This occurs for one of the following reasons:
  - No Group specified CONDITIONAL=CONTROL
  - No Group with CONDITIONAL=CONTROL was found on the VRF in this Merge Set.
  - All Group with CONDITIONAL = CONTROL have zero copies
  - The control Group is specified after the conditional Group in the SYSIN, but ALLERROR=Y was not specified.
- A conditional Group prints if any control Group also prints to the same DDNAME.
- Conditional Groups with errors always print.

DMGMERGE Performance Considerations

This section describes Documerge techniques that may improve your overall Documerge run time.

WORKBUFF

As described under "WORKBUFF=" on page 377, the WORKBUFF EXEC parameter defines the buffer size for DMGMERGE to use for storing the data associated with one complete Document Package. (A Document Package is one copy for one Group for one Merge Set.)

Here are some facts you need to know about WORKBUFF:

- If the storage space allocated for WORKBUFF is not large enough, the performance of DMGMERGE might be significantly slower.

  Storage for WORKBUFF is allocated *above the line* — in 31-bit addressing space. Generally it is better to over-allocate than under-allocate the WORKBUFF space.

- DMGMERGE prints diagnostic statistics about the use of WORKBUFF and its corresponding *WRKFIL* overflow disk file. You should periodically review these statistics.

  The statistics show the buffer space used, buffer space needed.

- The memory allocation specified with WORKBUFF is actually used by the DMGLMMM program, which is the Documerge list and memory manager.

  DMGLMMM first uses this memory allocation as a working buffer. If the buffer becomes full, then the DMGLMMM uses WRKFIL disk file as an overflow area.

- Excessive writing and reading from WRKFIL significantly slows DMGMERGE execution time.

  Ideally, WRKFIL should not be used.

TIP

For WRKFIL, specify SPACE=(TRK,(1,30)). *1* is the minimum allocation. If WRKFIL is never used (the ideal situation), you only have taken up 1 track while DMGMERGE is running.

If WRKFIL is needed, the secondary space of "30" should suffice, because any secondary space can be allocated up to 15 times.

- The maximum WORKBUFF value is **1,073,741,823** bytes (commas shown for clarity.) This is 1024 * 1024 * -1, or one gigabyte minus one.

  The WORKBUFF maximum allocation exceeds the memory capacity of most computers, so practically there is not a Documerge imposed maximum, but a maximum limit imposed by the capacity of the computer.

NOTE

You can use the NUMAREAS EXEC parm as an alternative to specify the DMGLMMM buffer size. The value specified for NUMAREAS is multiplied by the WRKFIL block size to get the buffer size.

Docucorp recommends using WORKBUFF instead of NUMAREAS because WORKBUFF sets the actual buffer size, whereas you have to calculate the buffer size for NUMAREAS, and NUMAREAS makes the buffer size dependent on the block size of WRKFIL.

The statistics that DMGMERGE generates show the buffer space used, buffer space needed., the number of areas used, and the number of areas needed. So the statistics show the WORKBUFF and NUMAREAS values.

If you specify both WORKBUFF and NUMAREAS, NUMAREAS is used. And if you don't specify either one, WORKBUFF=200K is assumed.

Optimizing the Block Size of WRKFIL

The block size of the WRKFIL can make a difference in processing speed.The minimum block size is 10,000 bytes. A larger block size can reduce some internal DMGMERGE overhead. Oracle recommends the maximum block size of 32760.

While the maximum allocation might waste some space — ideally you never use this WRKFIL file anyway. If you specify BLKSIZE=0 (or omit the BLKSIZE parameter) in the WRKFIL DD, then Documerge determines the block size that gives the best track utilization (usually 2 blocks per track).]

NOTE

The AREASIZE EXEC parameter can also specify the block size of the WRKFIL file. However, Oracle recommends that the block size be coded directly on the WRKFIL DD, using the BLKSIZE parameter, and that AREASIZE be omitted from the EXEC PARM.

If BLKSIZE is coded on the WRKFIL DD, that is the block size used, and any AREASIZE value is ignored. If BLKSIZE is omitted, then the AREASIZE value is used. If both BLKSIZE and AREASIZE are omitted, a block size of 10,000 is used.

Block Size for DMGMERGE Files

The block size you specify for certain DMGMERGE files can make a significant difference in the performance of the VDR, DMGSORT, and DMGMERGE. These files are:

- WRKFIL
- DMGVRF1

See "DMGMERGE Files" on page 378 for descriptions of these files.

Generally, the larger the block size, the better the performance, because the number of I/O's is decreased. The largest block size you can specify for these DMGMERGE files is the smallest of the following:

- Any limits imposed by the operating system (MVS maximum is 32760).
- 32767 — This is the Documerge maximum block size allowed.
- The track capacity of the DASD device (if using DASD).

Remember that the block size is usually a trade-off between performance and DASD space. For example, on a 3380 device, a block size of 32760 wastes approximately 14,716 bytes per track, because the track capacity is 47,476 bytes. A block size of 23476 for 3380 devices gives exactly 2 blocks per track and wastes very little DASD space, but is not as run-time efficient as a block size of 32760. If you have ample unused DASD, try a block size of 32760 for best run-time performance.

FORMSBUFF

DMGMERGE attempts to keep an EDL member in memory once it has been read, to avoid repeatedly reading the form from the EDL (the VLAM library). Processing a form in memory is significantly faster than reading a form from the EDL. The size of the memory to hold the forms is set by the FORMSBUFF parameter in the EXEC statement.

DMGMERGE prints a line in the MESSAGE file informing you of the FORMSBUFF usage: how much was specified and how much was actually needed for this particular execution of DMGMERGE. Periodically, you should examine the MESSAGE report from DMGMERGE and evaluate the FORMSBUFF size. A FORMSBUFF value that is too small causes significant degradation in DMGMERGE run time when the same EDL members are being used by different Merge Sets and/or Groups.

NOTE

This parameter must be FORMSBUFF, in capital letters, and the value must be a whole number followed by the capital letter "K". For example:

```
FORMSBUFF=500K.
```

If you misspell this word or enter it in lowercase letters, then DMGMERGE will not recognize it, but instead will ignore it and use the default value of 100K. However, the resulting MESSAGE file will contain messages about any EXEC parms and the errors associated with them.

TAGBUFF and DATABUFF

If TAGBUFF or DATABUFF is specified, the amount of memory available to read in a Merge Set can be limited. If memory is not available, the List Manager/Memory Manager (LM/MM) system (internal storage-management programs) is invoked.

This can cause significant degradation in DMGMERGE run time. Therefore, we strongly recommend that you use the VRF Allocation (DMGVRFA) file and do not specify TAGBUFF or DATABUFF parameters.

VLMCONTROL

If there are a large number of EDL members to be processed by Documerge, specifying VLMCONTROL=KEEP in the EXEC parameter will improve Documerge run time by reducing the number of I/O's on the EDL. VLMCONTROL=KEEP may be specified for both the VDR step and the DMGMERGE step.

Note that specifying VLMCONTROL=KEEP will prevent any updates to the EDL during the VDR or DMGMERGE step. Jobs attempting to update the EDL, such as a VLMMAINT run, would be placed in a wait state, and would execute as soon as the VDR or DMGMERGE step completed.

NOTE

This parameter must be VLMCONTROL, in capital letters, and the value must be either a whole number or the word "KEEP" in capital letters. Example:

```
VLMCONTROL=KEEP
```

If you misspell this word or enter it in lowercase letters, then DMGMERGE will not recognize it, but instead will ignore it and use the default value of zero (not keeping the alphabetic index in memory) and no error message is generated.

DMG.GCPY.*Groupname* Reserved Tag

The DMG.GCPY.*Groupname* Reserved Tag (*Groupname* is the name of a merge Group) is useful to create a user-specified number of copies of a merge set for a Group. The value of the tag is the number of copies desired. Note that this tag's value can be set to "0" (zero), to bypass all merging operation for this Group's Merge Set. DMGMERGE will not invoke a DMGPRNT program if the DMG.GCPY value is zero for the Group for a Merge Set, saving significant execution time. If you have some optional Groups, you should set DMG.GCPY to zero if you do not want output for the Group for a Merge Set. Also, be sure to not specify the COPIES parameter in the DMGMERGE SYSIN, as this overrides the DMG.GCPY tag value.

Calling DMGMERGE Dynamically

This section explains how to dynamically call DMGMERGE — once or several times.

Guidelines and Rules for Preparing Calls to DMGMERGE

- You must use 3.1.0 or a later version of Documerge.
- Documerge must be able to access EXEC parms; therefore, you need to either
  - Code the JCL so that the application save areas chain back to the supervisor-save areas.

    -or-
  - Perform the "Using EXEC Parameters Under IMS" procedure in *Installing Documerge*. This procedure configures DMGMERGE so that it gets its EXEC parms only from the ISIPFIL file.

  Choose which procedure to use, based on the needs of your application. It is always safe to use the "Using EXEC Parameters Under IMS" procedure even if not using IMS, so you might want to do this regardless.
- The JCL should contain DDs for
  - WRKFIL
  - PEDEF (if it does not vary)
  - VLM2LIB (assuming it does not vary)
  - MESSAGE (assuming it is SYSOUT=something)
  - Parmfile: if PARMFILE=xxxxxxx in the EXEC parms

    If you completed the "Using EXEC Parameters Under IMS" procedure, then this is a fixed name of ***ISIPFIL***. Or, if its contents need to vary, you can can dynamically allocate the parm file.

To Dynamically Call DMGMERGE Multiple Times

1. Dynamically build and allocate the DMGMERGE data sets that need to vary. These include the
   - SYSIN
   - DMGVRF1
   - DMGVRFA
   - DMGMERGE output files
   - PEDEF (maybe; if it does not vary, then it can be in the JCL)
   - VLM2LIB (maybe; if it does not vary, then it can be in the JCL)
   - parmfile (maybe; if it does not vary, then it can be in the JCL)
2. Call DMGMERGE via ISICALL.

   This is similar to a VDR calling DMGRFMT; no parms need to be passed. For example (COBOL):

```
(data division)

01  DMGMERGE-CONSTANT.
    05  DMGMERGE  PIC X(8)  VALUE 'DMGMERGE'.


(procedure division)

    CALL 'ISICALL' USING DMGMERGE.
```

3 Handle the DMGMERGE return code, (e.g., write its value to an output file).

4 Delete all dynamic programs from memory. For example (COBOL):

```
(data division)

01  DELETE-ALL-PROGRAMS.
    05  DELETE-FLAG  PIC X(2)   VALUE '*D'.
    05  DELETE-RC    PIC S9(4)  COMP.
    05  ALL-FLAG     PIC X(4)   VALUE '*ALL'.

(procedure division)

    CALL 'ISICALL' USING DELETE-FLAG ALL-FLAG.
```

5 Move 'DMGMERGE' back into DMGMERGE field because ISICALL has changed this constant to something else (an internal table pointer to speed up processing). For example (COBOL):

```
(procedure division)

    MOVE 'DMGMERGE' to DMGMERGE.
```

6 Deallocate any dynamically allocated data sets, as required. For example (Assembler):

```
--deallocate data sets here--
DMGMERGE  DC  CL8'DMGMERGE'
          DS  OH         ALIGN NEXT ITEMS ON HALFWORD
DELETE    DC  CL2'*D'
DELETERC  DS  H ISICALL  RETURN CODE FROM DELETEALL REQUEST
ALL       DS  CL4'*ALL'
```

7 End. For multiple executions, you can repeat the procedure, beginning with step 1.

STATSFILE

STATSFILE Format

The STATSFILE can be fixed, variable, or undefined length records (RECFM = F, FB, V, VB, or U.) We recommend variable length, but fixed length can sometimes be easier to read by a user post processing program. If fixed length, trailing unused positions in a record are set to blanks (hex 40).

The following is the minimum LRECL based on STATSTYPE and RECFM:

STATSTYPE	RECFM=F	RECFM=U	RECFM=V
SUMMARY	56	56	60
SUMMFORM	56	56	60
FULL	120	120	124
FULLFORM	120	120	124
SUMMTAG	40	40	44
FULLTAG	120	120	124
SUMMBPSD	40	40	44
FULLBPSD	120	120	124
SUMMERROR	8	8	12
FULLERROR	120	120	124

If more than one STATSTYPE is selected, use the largest LRECL required.

BLKSIZE, as always, depends on the RECFM and LRECL:

```
If RECFM=F, BLKSIZE = LRECL

If RECFM=FB, BLKSIZE must be a multiple of the LRECL, or 0 (zero) to let MVS select the optimum
BLKSIZE.

If RECFM=V, BLKSIZE = LRECL plus 4

If RECFM=VB, BLKSIZE must be at least LRECL plus 4, or it can be zero to let MVS select the optimum
BLKSIZE.

If RECFM=U, BLKSIZE = LRECL
```

We suggested the following: DCB=(RECFM=VB,LRECL=124,BLKSIZE=0).

NOTE

Program DMGDELETE requires RECFM V or VB. See "DMGDELET" on page 458.

STATSFILE Layout

Layout of the STATSFILE, relative to one, is as follows (numbers in parentheses are the lengths of the fields):

For STATSTYPE=SUMMARY and STATSTYPE=SUMMFORM

Position	(Length)	Description
001	(1)	Constant "S" indicating a summary form statistical record.
002	(1)	Filler for alignment; value is hex zero.
003-010	(8)	DDNAME of EDL.
011-042	(32)	EDL member name (i.e.: form name).
043-044	(2)	Revision level. Binary halfword, value from 1 to 32767.
045-048	(4)	Chain name.
049-052	(4)	Use count. Binary fullword. The number of times this form was used during this DMGMERGE run. Technically, the number of times the first record was needed for this form.
053-056	(4)	Buffer space (FORMSBUFF) required for this form. Binary fullword.

For STATSTYPE=FULL and STATSTYPE=FULLFORM

Position	(Length)	Description
001	(1)	Constant "M" indicating an Merge Set form statistical record. More technically, this is information for one Group for one Merge Set.
002	(1)	Filler for alignment.
003-010	(8)	DDNAME of EDL.
011-042	(32)	EDL member name (i.e.: form name).
043-044	(2)	Revision level. Binary halfword, value from 1 to 32767.
045-048	(4)	Chain name.
049-052	(4)	Use count. Binary fullword. The number of times this form was used during this DMGMERGE run. Technically, the number of times the first record was needed for this form.
053-056	(4)	Buffer space (FORMSBUFF) required for this form. Binary fullword.
057-091	(35)	Mergeset ID (the value of the DMG.MERGESET.ID tag).
092	(1)	Filler for alignment.
093-096	(4)	Merge set sequence number (the value of DMG.SET.NUMBER). Binary fullword.
097-117	(21)	Group name.
118	(1)	Filler for alignment.
119-120	(2)	Group sequence number. Binary halfword. This field is the sequential number of the "MERGE" command in the DMGMERGE SYSIN. Use this field to distinguish between two MERGE commands that specify the same Group name.

For STATSTYPE=SUMMTAG and STATSTYPE=SUMMBPSD

Position	(Length)	Description
001	(1)	Constant "T" indicating a tag statistical record.
002	(1)	Constant "S" indicating a summary record.
003-032	(30)	Tag name.
033-036	(4)	Number of times this tag was referenced during this DMGMERGE run. Binary fullword.
037-040	(4)	Number of times this tag was used in a BPSD during this DMGMERGE run. Binary fullword. Note that this total is also included in the previous field in positions 033-036. Also note that for STATSTYPE=SUMMBPSD, this value will equal the value in positions 033-036 because only BPSD tags are reported.

For STATSTYPE=FULLTAG and STATSTYPE=FULLBPSD

Position	(Length)	Description
001	(1)	Constant "T" indicating a tag statistical record.
002	(1)	Constant "M" indicating a Merge Set record. More technically, this is information for one Group for one Merge Set.
003-032	(30)	Tag name.
033-036	(4)	Number of times this tag was referenced during this DMGMERGE run. Binary fullword.
037-040	(4)	Number of times this tag was used in a BPSD during this DMGMERGE run. Binary fullword. Note that this total is also included in the previous field in positions 033-036. Also note that for STATSTYPE=FULLBPSD, this value will equal the value in positions 033-036 because only BPSD tags are reported.
041-092	(52)	Filler; binary zeros. Alignment so that all "FULL" statstypes have their Merge Set ID start in the same relative position.
093-096	(4)	Merge set sequence number (the value of DMG.SET.NUMBER). Binary fullword.
097-117	(21)	Group name.
118	(1)	Filler for alignment.
119-120	(2)	Group sequence number. Binary halfword. This field is the sequential number of the "MERGE" command in the DMGMERGE SYSIN. Use this field to distinguish between two MERGE commands that specify the same Group name.

For STATSTYPE=SUMMERROR

Position	(Length)	Description
001	(1)	Constant "E" indicating an error statistical record.
002	(1)	Constant "S" indicating a summary record.
003-004	(2)	Error number. Binary halfword. See the error message manual for error numbers. This is the "nnn" portion of the message idenfier "DMGMRGnnnR". (R is the message severity.)
005-008	(4)	The number of times this error has occured in this DMGMERGE run. Binary fullword. This value will never be zero. If a given error message did not occur, no stats record will be generated for that error. If no errors occured, there will not be any "E" stats records.

For STATSTYPE=FULLERROR

Position	(Length)	Description
001	(1)	Constant "E" indicating an error statistical record.
002	(1)	Constant "M" indicating a Merge Set record. More technically, this is information for one Group for one Merge Set.
003-004	(2)	Error number. Binary halfword. See the error message manual for error numbers. This is the "nnn" portion of the message idenfier "DMGMRGnnnR". (R is the message severity.)
005-008	(4)	The number of times this error has occured for this Group for this Merge Set. Binary fullword. This value will never be zero. If a given error message did not occur, no stats record will be generated for that error. If no errors occured, there will not be any "E" stats records.
009-092	(52)	Filler; binary zeros. Alignment so that all "FULL" statstypes have their Merge Set ID start in the same relative position.
093-096	(4)	Merge set sequence number (the value of DMG.SET.NUMBER). Binary fullword.
097-117	(21)	Group name.
118	(1)	Filler for alignment.
119-120	(2)	Group sequence number. Binary halfword. This field is the sequential number of the "MERGE" command in the DMGMERGE SYSIN. Use this field to distinguish between two MERGE commands that specify the same Group name.

The actual use of this file is up to the client. The client might want to write post-processing programs to accumulate use counts for several DMGMERGE runs. Users of Oracle imaging program I.R.I.S. use the STATSFILE for I.R.I.S. maintenance; see "DMGDELET" on page 458 for more information.

The use count is incremented any time a form is requested. If multiple copies of a Group are requested, the use count reflects these multiple copies. (Multiple copies are requested either via the COPIES parameter parameter or the DMG.GCPY special tag.)

PROGRAMMER'S NOTE

FULLWORD is 4-bytes binary. COBOL: PIC S9(9) comp.

HALFWORD is 2-bytes binary. COBOL: PIC S9(4) comp.

Documerge Utilities

Documerge includes a number of utilities that increase its flexibility, adaptability and smooth operation. These stand-alone programs and utilities include:

- DMGOPNCL — opens, and then closes every file called by a Documerge DD statement.
- DMGDELET — facilitates the interface between Documerge and the Oracle I.R.I.S. imaging product.
- DMGVDRG (Generic VDR) — reprocesses Documerge VRFs for error correction with Docusolve.
- DMGZEROL — creates an index-packet-only form, or adds zero-length tags to an existing form at the beginning of its index packet.

DMGOPNCL

DMGOPNCL helps avoid unintended results from Documerge when MVS finds a file that has been allocated, but not opened or closed (no EOF record). DMGOPNCL simply opens, and then closes, every file called by a Documerge DD statement. In this way every dataset has an EOF record, and VSE handles Documerge correctly.

After you run your VDR, run DMGOPNCL, repeating each of the VRF DD statements from the VDR. Use the DISP=MOD parameter in your JCL; DMGOPNCL will add an EOF record to each file.

DMGOPNCL JCL Example

```
//DMGOPNCL  ** put your job card here **
//*
//*   DOCUMERGE V. 3.2 -- INSURES OUTPUT FILES HAVE EOF
//*   RUN AFTER THE DOCUMERGE STEP THAT ALLOCATES THE FILES
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGOPNCL EXEC PGM=DMGOPNCL
//*
//SYSIN  DD  *              <= FOLLOW WITH DDNAMES OF OUTPUT FI
FILE1
FILE2
/*
//FILE1 DD DSN=your.output.file one,
//            DISP=MOD
//FILE2 DD DSN=your.output.file.two,
//            DISP=MOD
//MESSAGE  DD SYSOUT=*,     <= DMGOPNCL MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//
```

DMGDELET

DMGDELET is a stand-alone Documerge utility designed primarily to assist in the interface between Documerge and Oracle I.R.I.S. imaging product.

DMGDELET reads the STATSFILE and deletes any members from a specified EDL DD name that are listed in the STATSFILE (i.e., that were printed during the last Documerge run). See "STATSFILE" on page 452 for more information.

Use DMGDELET with the EDLNAMES= EXEC parameter of DMGMERGE (see "EDLNAMES=" on page 375).

DMGDELET JCL Example

```
//DMGDELET  ** put your job card here **
//*
//*   DOCUMERGE V. 3.2 -- DELETE MEMBERS PROCESSED BY DMGMERGE
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGDELET EXEC PGM=DMGDELET,
//         PARM='edlname=libname'
//*
//LIBNAME  DD DSN=your.edl.whose.members.will.be.deleted,
//            DISP=SHR
//STATSFL  DD DSN=your.statsfile.from.dmgmerge,
//            DISP=SHR
//MESSAGE  DD SYSOUT=*,             <= DMGDELET MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//
```

Using DMGDELET with I.R.I.S.

Here is the suggested method for using DMGDELET to assist with a Documerge / I.R.I.S. interface.

1. Run the I.R.I.S Image Import routine to create an EDL member.
2. Run a Documerge cycle consisting of VDR, DMGSORT (optional), and DMGMERGE. Select the STATSFILE= control card when you run MERGE. (See "STATSFILE=" on page 397 for more information). Use a DD name of IRISEDL for the I.R.I.S. EDL.

   Oracle recommends using the slash (/) option to designate the I.R.I.S EDL(s) in the DMGMERGE EDLNAMES parameter. The slash tells Documerge not to include datasets from that EDL in the FORMSBUFF (see "EDLNAMES=" on page 375 for information about use of the slash with the EDLNAMES= parameter, and about the FORMSBUFF parameter).
3. Print the DMGMERGE output to verify correct Documerge operation.
4. Run DMGDELET. You should run DMGDELET before your next DMGMERGE run. Use the STATSFILE created during your last DMGMERGE run, with the following EXEC PARAMETER:

   ```
   PARM='EDLNAME=IRISEDL'
   ```
5. Back up the IRISEDL to tape, using VLMMAINT. VLMMAINT performs the actual deletions. You do not need to run VLMMAINT every time you use DMGDELET. In practice, most shops should find once a week to be sufficient.

The above procedure allows you to incorporate scanned images in your normal Documerge production runs, automatically tracking and marking for deletion those data files that you only want to use for a limited number of times. DMGDELET then purges your DASD of the unwanted data sets.

Notes on Usage

The DMGDELET EXEC parameter "EDLNAME=" should not be confused with the DMGMERGE parameter "EDLNAMES=". EDLNAME= is singular, referring to only one EDL at a time, to avoid accidental deletions. In addition, no parentheses are allowed with EDLNAME=, unlike EDLNAMES=, DMGDELET can only select one EDL DD name in a single DMGMERGE run.

DMGDELET marks the entire VLAM member for deletion from the EDL, including all chains for the member if the member carried multiple chains.

DMGDELET uses STATSFL for its STATSFILE DD name; this name is not user selectable. See "STATSFILE" on page 452 for more information.

DMGDELET uses the EDL DD name (as opposed to the actual dataset name) to select the VLAM EDL from which to delete members. You should ensure that the EDL name (DD name and DSN) selected by the DMGDELET EXEC parameter is identical to the EDL name specified in the DMGMERGE JCL.

DMGVDRG (Generic VDR)

Documerge 3.x provides a generic VDR (DMGVDRG) to reprocess a VRF. The generic VDR accepts a VRF as input. This VRF could be created by Docusolve, after the Docusolve user has corrected the errors from a Documerge run.

The VRF has all the data required to call DMGRFMT for forms processing and to call DMGVRFWR to write tags. The VDR copies the tags from the input VRF to an output VRF; then it calls DMGRFMT, passing the RFCB (DMGRFMT control block) and the explicit forms lists, which come from special tags in the input VRF. Documerge then proceeds as usual with DMGSORT, if desired, and DMGMERGE.

The following JCL runs the generic VDR:

Generic VDR (DMGVDRG) Example

```
//DMGVDRG   ** put your job card here **
//*
//*   DOCUMERGE V. 3.2 -- GENERIC VDR TO REPROCESS ERRVRF
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGVDRG  EXEC PGM=DMGVDRG,REGION=4M,
//            PARM='WORKBUFF=500K VLMCONTROL=KEEP VLMACCESS=RO'
//*
//VLM2LIB  DD DSN=your.documerg.v03r02.edl,
//            DISP=SHR
//*
//ERRVRF   DD DSN=your.error.vrf,    <= ERRVRF FROM DMGMERGE, OR
//            DISP=SHR              <= CORRECTED ERRVRF FROM DOCUSOLVE
//*
//ERRVRFA  DD DSN=your.error.vrfa,   <= ERRVRFA FROM DMGMERGE
//            DISP=SHR
//*
//*  Note:  If Docusolve or other product has changed the ERRVRF file,
//*         then the ERRVRFA file might not exactly match the ERRVRF
//*         file.  Program DMGVDRG will still run ok, utilizing
//*         WORKBUFF and WRKFIL space if needed to read in the
//*         ERRVRF file.
//*
//DMGVRF1  DD DSN=your.documerg.v03r02.vrf,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(15,15),RLSE),
//            DCB=BLKSIZE=23476                    HALF TRACK
//DMGVRFA DD DSN=your.documerg.v03r02.vrfa,
//           DISP=(NEW,CATALG,DELETE),
//           UNIT=sysda,
//           SPACE=(TRK,(1)),
//           DCB=(RECFM=F,BLKSIZE=80,LRECL=80)
//RBLIB    DD DSN=your.documerg.v03r02.rbl,
//            DISP=SHR
//*
//WRKFIL   DD DSN=&&WRKFIL,
//            DISP=(NEW,DELETE,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,30)),
//            DCB=BLKSIZE=23476                    HALF TRACK
//VDRGMSG  DD SYSOUT=*,                 <= DMGVDRG MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//*
//MESSAGE  DD SYSOUT=*,     <= DMGRFMT MESSAGES
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD SYSOUT=*
//
```

Generic VDR System Flow

Use the following procedure to correct Documerge errors with Documaker WS (Docusolve).

1. Run your normal VDR. Set switches RFCB-WRITE-RFCB and RFCB-WRITE-EXPLICIT-FORMS both to Y (see "The DMGRFMT Control Block (RFCB)" on page 218 for more information about these control block fields).
2. Optionally run DMGSORT.
3. If the ERRVRFA file (the Allocation file for the input VRF) does not exist, run DMGBLDVA against the ERRVRF input file to create the ERRVRFA file.

   Or, you can specify DMGVRFA=NO in the DMGVDRG EXEC parameter. This means that the DMGVRFA file is not created.
4. Run DMGMERGE, with ERRVRF processing and ALLERROR. DMGMERGE creates an error VRF for those merge sets in error. DMGMERGE routes all groups from a merge set to the ERRDDN if any group has an error.
5. Load the error VRF into Docusolve.
6. Use Docusolve to correct the errors in the VRF.
7. Use Docusolve to create a new VRF.
8. Run the generic Documerge VDR, specifying your desired EDL and RBL. Use the Docusolve created VRF as input. The output is a new VRF.
9. Optionally run DMGSORT.
10. Run DMGMERGE, again selecting ERRVRF and ALLERROR. If there are still errors, you can repeat the correction process with Docusolve as many times as necessary, beginning with step 4 above.

DMGZEROL

The DMGZEROL utility can perform the following operations:

- Create an index-packet-only (non-printing) form with zero-length BPSD tags.
- Insert zero-length tags in an existing form at the beginning of the index packet.

You can load the form containing zero-length tags to your EDL, and Documerge can then use the form as a *control form* to do the following:

- Reset deleted tags to control the processing of subsequent forms. For details, see "The Boilerplate Space Definition (BPSD)" on page 32.
- Generate internal statistics or audit records by calling a user-exit program (via a DMG.C.xxx tag). For details, see "DMG.C.xxx Reserved Tag Processing" on page 331.

DMGZEROL JCL Example

```
//DMGZEROL  ** put your job card here **
//*
//* ********************************************************************
//* ** DMGZEROL JCL executes DMGZEROL to add zero-length tag records **
//* ** to the index packet of an existing form or to build a stand-  **
//* ** alone index packet of zero-length tag records.                **
//* **                                                               **
//* ** This JCL illustrates the generation of a standalone index     **
//* ** packet containing one tag record.                            **
//* ********************************************************************
//*
//JOBLIB   DD DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//ZEROL    EXEC PGM=DMGZEROL,REGION=4M,
//  PARM=' / PDEV=AFP'
//INFILE   DD DUMMY
//OUTFILE  DD DSN=your.output.file,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=sysda,
//            SPACE=(TRK,(1,1),RLSE),
//            DCB=(RECFM=VB,LRECL=8205,BLKSIZE=10000)
//SYSOUT   DD SYSOUT=*
//MESSAGE  DD SYSOUT=*,
//            DCB=(RECFM=FBM,LRECL=133,BLKSIZE=1330)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  :BPSD NAME='your.tag.name' TYPE=O DELETE=N
/*
//
```

TheDMGZEROL Input Files

The DMGZEROL input files are SYSIN and INFILE, which can be an input form.

If INFILE is an existing form, DMGZEROL determines the data stream to produce based on the format of the incoming records. If there is no INFILE form, assign INFILE to DD DUMMY.

To produce an index-packet-only form from scratch, you must supply the following EXEC parm:

```
PDEV=printtype
```

and replace the *printtype* placeholder with one of the following values:

- `LINE` (line printer, the default)
- `META` (Metacode)
- `AFP` (IBM Advanced Function Presentation)

If INFILE is an existing form, DMGZEROL ignores the PDEV= parameter.

The DMGZEROL Output File

The DMGZEROL output file name is OUTFILE, which contains a VLAM-compatible form.

DMGEROL Input Data Format

SYSIN consists only of BPSD specifications, one per record, in the following format:

Position	Description
1	**:** (colon) To make the record a comment that is ignored by DMGZEROL, code a **-** (hypen) in position 1 instead.
2–5	BPSD (code the colon in column 1)
6	blank
7-	**NAME='**xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx' The only required parameter. The BPSD tag name can be a maximum of thirty characters enclosed in single quotes. Tag name characters can include letters, numbers, *at* sign (@), dollar sign ($), a period (.), but not a dash (-).
	blank
	First addtional optional parameter Code optional parameters in any sequence with one or more blanks between them. Optional parameters can be any of the following: ■ **LENGTH=** — if specified, the LENGTH value must be zero (0). ■ **DELETE=** — the value can be Y(ES), N(O), 1, or R. Defaults to **N** (do not delete). ■ **TYPE=** — the value can be O(PTIONAL) or M(ANDATORY). Defaults to **O** (optional). Other BPSD parameters (CHAR=, LINEEND=, MULTI=, GEN=, and CONT=) are not allowed because no actual print data is generated.
	As the last characters in a physical record, you can code a blank and a **-** (hyphen) as continuation characters that indicate that the logical record is to span addtional physical records.

The following examples illustrate valid and invalid BPSD tag records:

Example-1 (valid BPSD specification with additional parameters):

```
:BPSD NAME='ZERO.LENGTH' LENGTH=0 TYPE=M
```

Example-2 (invalid BPSD specification with LENGTH parameter value of 25):

```
:BPSD NAME='NON.NUMERIC.LENGTH' TYPE=M DELETE=N LENGTH=25
```

Documerge Overlay Forms

Overlay Concepts — Scopes, Levels, and Sets

In Documerge 3.1 and later versions, expanded Overlay options offer greater flexibility and control based on where and how you specify the Overlays. To take better advantage of these Overlay options, you first need to understand three concepts:

- **The *Overlay scope*** — in previous versions of Documerge, a form defined as an Overlay was associated with a particular DTN, and several DTNs were typically defined for a Document Package. In the following example, the Overlay TOPFIRST is within the scope of DTN 10, and we call this a *DTN-scope* Overlay.

```
STRUCTURE -
       RULE=((10 DUP OVL=('TOPFIRST' NONBLANK SHEET)) -
```

  For Documerge 3.1 and later versions, you can specify Overlays to apply to all pages in a Document Package. In the following example, the Overlay TOPALL has been specified after the STRUCTURE command, but before the RULE= subcommand. We call this a *Document Package-scope* Overlay.

```
STRUCTURE -
       OVL=('TOPALL' ALWAYS   SHEET) -
       .
       .
       RULE=((10   DUP OVL=('OVERLAY4')) -
```

  You can also define General or Document Package-scope Overlays in MERGE, FILEDEF, and GLOBAL commands. For more information, see "Specifying Overlays with the DMGMERGE Commands" on page 474.

  Use Document Package-scope Overlays to print bar codes, the current date and time (DMG.DATE and DMG.TIME Reserved tags), or any text that needs to appear on every sheet of a Document Package.

- The ***Overlay level*** — For convenience and flexibility, Documerge 3.1 and later versions let you specify Overlays at five different levels. Where (in which Documerge command or parameter) you code the Overlay specification determines its level.

The following table provides the name, coding location, scope, processing rules, and applications for each Overlay level.

Overlay Level	Where Specified	Scope	**Processing Rule** (for each level, whether or not a form receives an Overlay depends on the WHEN, WHERE, and APPLIESTO values specified) for that form)
DTN	Rulebase STRUCTURE RULE subcommand	DTN	The forms within the scope of the DTN get the Overlay
Package	Rulebase STRUCTURE command	Package	All forms in the specified Group get the Overlay
Group	MERGE command in the DMGMERGE SYSIN	Package	All forms printed for the specified Group get the Overlay
FILEDEF	FILEDEF command in the DMGMERGE SYSIN	Package	All forms printed to the file named in the FILEDEF get the Overlay
GLOBAL	GLOBAL command in the DMGMERGE SYSIN	Package	All forms printed in the DMGMERGE run get the Overlay

- An ***Overlay set*** can consist of one or both of the following:
  - ***DTN-scope Overlays*** — all the consecutively-defined Overlay names in a STRUCTURE RULE defined for a single DTN.
  - ***Package-scope Overlays*** — all GLOBAL-level, FILEDEF-level, MERGE-level, and Document Package-level Overlays combined.

Overlay Placement Options — APPLIESTO, WHEN, and WHERE

Specifying the Page Type for an Overlay — the APPLIESTO Option

The Print Option or DMGMERGE command BVL, FVL, and OVL parameters have an *APPLIESTO* option that you can use to specify the printing of Overlays on any of the following types of DMGMRGE-processed pages or sheets:

- **LOGICAL** — prints the Overlay on each logical page defined for the DTN (or physical page if no logical pages are defined).

  If the Overlay specification contains the WHEN option, the Overlay won't print unless a non-Overlay form printed on the associated logical page. For details, see "Print Option Parameters" on page 172.

  The logical Overlay will print for all logical pages defined for the DTN because any front (FVL) or back (BVL) Overlays refer to the physical page. For example, if you define two logical pages on the front side (FLPAGE), they both get any specified FVL. For details, see "LPGDEF Command Format" on page 160.

- **PHYSICAL** — prints the Overlay on the physical page containing the DTN.

  PHYSICAL is the default currently used by Documerge.

- **SHEET** — prints the Overlay on the side of the sheet of paper specified for the DTN.

  Use the SHEET option when you want

  - Documerge *not* to shift the Overlay. If you specify a LOGICAL page Overlay, Documerge will shift it by any specified FSHIFT or BSHIFT value. For details, see "LPGDEF Command Format" on page 160.
  - Documerge to print a bar code for the sheet of paper containing imposition layouts (IMPDEFs). For example, if the IMPDEF created two physical pages for each sheet side, and four physical pages per sheet, then you can use the SHEET option to print a single bar code for the sheet.
  - Documerge to place the Overlay last on the physical sheet, after all other forms are placed. For example, when you need to print a bar code, sheet number, or other sequential count on the sheet to make it easier to reassemble the sheets if they get scattered.

CAUTION!

To avoid corrupting Reserved tag values that report start or end of processing conditions, do not use the following Reserved tags in both a sheet Overlay and another form to be printed in the same Document Package:

- DMG.END.OF.SUBSET
- DMG.END.OF.SET
- DMG.BENCHMARK
- DMG.HASHMARK

Specifying the Form-Related Condition for Printing an Overlay — the WHEN Option

The Print Option or DMGMERGE command BVL, FVL, and OVL parameters have a *WHEN* parameter, which specifies the printing an Overlay when the DMGMERGE-processed page is controlled by other parameters or contains other forms.

- **DEFAULT** — use the default specified in the FVLDEFAULT, BVLDEFAULT, or OVLDEFAULT parameters for the GLOBAL command.
- **ALWAYS** — the Overlay prints on every page. ALWAYS is the default.
- **NONBLANK** — the Overlay prints only if a non-Overlay form is printed on the page.
- **ONLYBLANK** — the Overlay prints only if no other form is printed on the page.

Specifying the Printing Location of an Overlay — the WHERE Option

The Print Option or DMGMERGE command BVL, FVL, and OVL parameters have a *WHERE* option, which specifies one of the following printing locations for an Overlay in an Overlay set:

- **FIRST** — prints the Overlay only on the first page of the Overlay set.
- **NOTFIRST** — prints the Overlay on all pages except the first.
- **LAST** — prints the Overlay only on the last page of the Overlay set.
- **NOTLAST** — prints the Overlay on all pages except the last page.
- **MIDDLE** — prints the Overlay on all pages except the first and last pages.
- **ALL** — (Default) prints the Overlay on all pages except the banner and trailer pages.

> **TIP**
>
> Documerge does not apply General Overlays specified for the GLOBAL, FILEDEF, and MERGE commands to the banner and trailer pages.
>
> If you want the same information that you have specified in a General Overlay to print on the banner or trailer pages, implement a BPSD tag for the banner or trailer page which contains the same information and appears in the same location as the Overlay.

Following are the valid WHERE values, their associated actions, and some typical uses

Value	Action	Usage/Coding Rules
FIRST	Prints the Overlay only on the first page of the Overlay set. The definition of the first page depends on which of the Overlay parameters you code the WHERE option for: ▪ **OVL** (front and back Overlay) — the first page is the **initial front or back side**. ▪ **FVL** (front Overlay) — the first page is the **initial front side**. ▪ **BVL** (back Overlay) — the first page is the **initial back side**. Complies with other Overlay options such as WHEN and page parity.	To print a special heading on the first page of a group of forms (e.g., concatenated forms). To print a special beginning bar code on the first page of the Document Package.
NOTFIRST	Prints the Overlay on all pages except the first. Complies with other Overlay options such as WHEN and page parity.	To print a running header or footer (e.g., CONTINUED FROM PREVIOUS PAGE) on succeeding pages, some of which could contain different concatenated forms.

LAST	Prints the Overlay only on the last page of the Overlay set. The definition of the last page depends on which of the Overlay parameters you code the WHERE option for: ▪ **OVL** (front and back Overlay) — the last page is the **final front or back side**. ▪ **FVL** (front Overlay) — the last page is the **final front side**. ▪ **BVL** (back Overlay) — the last page is the **final back side**. Complies with other Overlay options such as WHEN and page parity.	To print a special heading or footer on the last page of a group of forms (e.g., END OF BENEFITS). To print a special ending bar code on the last page of the Document Package.
NOTLAST	Prints the Overlay on all pages except the last page. Complies with other Overlay options such as WHEN and page parity.	To print a running header or footer (e.g., CONTINUED FROM PREVIOUS PAGE) on the pages, some of which could contain different concatenated forms.
MIDDLE	Prints the Overlay on all pages except the first and last pages. Complies with other Overlay options such as WHEN and page parity.	To print a running header or footer on all pages except the first and last.
ALL	Prints the Overlay on all pages, complying with other Overlay options such as WHEN and page parity.	ALL is the default WHERE option.
For details about **WHEN** and other Overlay print options, see "xVL=( )" on page 178.		

Overlay Placement Rules and Guidelines

Overlay Placement Priorities for Different Scopes, Levels, and Options

At the start of a new package, all Overlays are reset — no Overlays are active. The order in which Documerge places Overlays or other forms on a page can sometimes impact how options specified for the forms will work.

If an Overlay specifies a WHEN option, Documerge prints it on the next available corresponding (front, back, or next available front or back) single-sided image (SSI). When placing Overlays specified for more than one level or scope, Documerge places the

(1) GLOBAL-level Overlays, in the sequence coded.

(2) FILEDEF-level Overlays, in the sequence coded.

NOTE

The FILEDEF applies Overlays only to clean files (DDNAME= or DMG.DD.groupname tags). The FILEDEF does not apply Overlays defined for packages that are erred (ERRDDN= or DMG.ERDD.groupname tags).

If Routing-by-Sheets processing is used, **only the first clean output file** defined for the FILEDEF is used for a LOGICAL or PHYSICAL Overlay. This is because the final output file is not known at the time LOGICAL and PHYSICAL Overlays are applied.

However, FILEDEF-specified SHEET Overlays will use **the final clean output file** because DMGMERGE SHEET Overlay processing occurs after the final-output clean output file created.

(3) MERGE-level Overlays, in the sequence coded.

(4) Package-level Overlays, in the sequence coded in the Rulebase (the sequence found in the DMG.FLST.*Groupname* tag).

(5) DTN-level Overlays, in the sequence coded in the Rulebase (the sequence found in the DMG.FLST.*Groupname* tag).

CAUTION!

Documerge merges tag data for sheet Overlays after it merges tag data for all other forms in a Document Package. Because of this fact, you should use caution when assigning BPSD tag attributes for non-sheet-Overlay and sheet Overlays in the same Document Package.

For example, if a tag's data has been deleted because DELETE=Y has been specified for the tag for both a non-Overlay form and a sheet Overlay, the tag data might not appear in the correct sequence.

To avoid corrupting reserved tag values that report start or end of processing conditions, do not use the following reserved tags in both a sheet Overlay and another form to be printed in the same Document Package:

- DMG.END.OF.SUBSET
- DMG.END.OF.SET
- DMG.BENCHMARK
- DMG.HASHMARK

The Affect of Overlays on Non-Overlay (Main) Forms

How an Overlay Can Cause a Non-Overlay Form to Shift Down the Page

When Documerge applies an Overlay, there are several conditions that can cause a main (non-Overlay) form to shift down the page:

- For the *APPLIESTO* options of PHYSICAL and LOGICAL, Documerge shifts a main form down the page by the size of the largest Overlay which has a WHERE option of ALL, FIRST, or NOTFIRST.

  An Overlay with a WHERE option of LAST, NOTLAST, or MIDDLE never causes a main form to shift. This is because this type of Overlay cannot be applied until all main forms have been applied.

- A PHYSICAL Overlay will shift any main form that is not part of a logical page.
- A LOGICAL Overlay will shift only a main form that is also on the Overlay's logical page. The intent of this shifting is to allow an Overlay to be added as a running header, thus shifting main forms down the page to make room for the running header.

  This shifting affects concatenation and might cause concatenated forms to print on a new page, or cause a form to shift off the page, generating the DMGMERGE error message DMGMRG536C.

  However, you can shift a form on a logical page from outside of the current logical page definition, without DMGMERGE generating an error. To produce the effect of downward shifting for a logical page that needs a running header Overlay, the logical page itself must be redefined with a lower vertical starting position.

How to Control the Shifting of A Non-Overlay Form Down the Page

Here are some guidelines you can use to avoid unanticipated shifting of main forms:

- To keep a main form from shifting, use the *APPLIESTO* option for a SHEET Overlay.
- If you specify it for a logical page, the *APPLIESTO* option for a PHYSICAL Overlay will not cause any main form to shift. (This is because logical pages are often used with perforated paper and must be placed exactly where specified.)
- Specify a logical bottom value of zero for a true Overlay form (a form that actually should print on top of an existing main form), so no main form is shifted.
- Specify the logical bottom for a running-header Overlay according to its true vertical dimension, so the main form will print lower on the page to make room for the running header.

Overlay Coding Methods

You can define Overlays in the VRF (usually using the Rulebase), or you can define them in the GLOBAL, FILEDEF, and/or MERGE commands in the DMGMERGE SYSIN in any combination desired.

Specifying Overlays with the DMG.OPT.*Groupname* Reserved Tag

The following explanation addresses the DMG.OPT reserved tag described on page 317.

The Overlay options in the DMG.OPT.*Groupname* tag are always 3 characters long. Possible values are

- ON (ON followed by one blank)
- OFF
- xVx

Main (non-Overlay) forms should have a value of ON or OFF. These values indicate if the currently active DTN-SCOPE Overlay set is to be printed.

For main forms, only the first form on the page (physical or logical or sheet) determines if any Overlays should print, by using the ON or OFF value of the Overlay option from the DMG.OPT.groupname tag. If subsequent main forms are placed on the same page (because of concatenation or logical pages), their Overlay option values are ignored.

An Overlay form uses a DMG.OPT.*Groupname* value of **xVx**. The second character of the value for Overlay forms is always a "V". The first and third characters define one of the Overlay options — page parity, APPLIESTO, scope, WHEN, or WHERE.

As described previously, the Overlay Set is all consecutive Overlays in the DMG.FLST.groupname tag. For example, if the DMG.FLST.INSURED tag has the following contents:

`FORM1` (Overlay)

`FORM2` (Overlay)

`FORM3` (main form)

`FORM4` (main form)

`FORM5` (Overlay)

`FORM6` (Overlay)

`FORM7` (Overlay)

`FORM8` (main form)

Then FORM1 and FORM2 is one Overlay set, and FORM5, FORM6, and FORM7 is another Overlay set.

VRF-defined Overlays can be DTN level or Package level:

- For DTN-level Overlays, only Overlays from the current (last found) Overlay set are used. In the example above, suppose FORM2 and FORM6 are DTN-scope. Then FORM8 would get only FORM6 and not FORM2, because FORM2 is not in the Overlay set that immediately precedes FORM8. In other words, when a new Overlay set is found, all DTN scope Overlays are reset.
- Package-level Overlays are never reset. Once a package-scope Overlay is found in an Overlay set, it remains active for the duration of the package. In the example above, suppose FORM1 was a package level Overlay. Then FORM8 could get FORM1 as an Overlay.

For any main form, the current active Overlays are

- All Overlays from the DMGMERGE GLOBAL command
- All Overlays from the clean file's DMGMERGE FILEDEF command. If using Routing-by-Sheets, PHYSICAL and LOGICAL Overlays use the first clean file; SHEET Overlays use the final selected clean file.
- All Overlays from the DMGMERGE MERGE command
- All previously encountered package-level Overlays
- All DTN-level Overlays from the immediately preceding Overlay set

From the list of active Overlays, DMGMERGE can fulfill the following additional specifications:

- APPLIESTO (PHYSICAL, LOGICAL, or SHEET)
- Page parity (FRONT, BACK, or BOTH)
- WHEN (ALWAYS, NONBLANK, or ONLYBLANK)
- WHERE (ALL, FIRST, NOTFIRST, LAST, NOTLAST, MIDDLE)

An Overlay is printed only when it meets all processing criteria.

As mentioned, Overlay forms have a DMG.OPT.*Groupname* option format of **xVx**. Following is description of the format's possible values:

byte 1: Represents the page parity, APPLIESTO option, or the SCOPE.

byte 2: Always 'V'.

byte 3: Represents the WHEN or WHERE option.

Following is the layout of byte 1. Across is the LEVEL and page parity option; down is the applies-to option:

	DTN-Level			Package-Level		
	OVL	BVL	FVL	OVL	BVL	FVL
LOGICAL	A	C	D	J	K	M
PHYSICAL	O	B	F	P	Q	R
SHEET	E	G	H	S	T	U

Following is the layout of byte 3. Across is the WHERE option; down is the WHEN option:

	LAST	NOTFIRST	NOTLAST	MIDDLE	ALL
ALWAYS	C	D	E	F	A
ONLYBLANK	I	J	K	M	O
NONBLANK	R	S	T	U	N
DEFAULT	X	Y	Z	G	L

Specifying Overlays with the DMGMERGE Commands

With Documerge, you can specify Package-scope Overlays in the FILEDEF, GLOBAL, or MERGE commands.

These *general* Overlay options, which apply to all the forms in the Document Package — except the banner and trailer pages — offer greater convenience, flexibility, and control. You can implement Overlays at the DMGMERGE command level without Rulebase or VDR changes.

TIP

Documerge does not apply General Overlays to the banner and trailer pages.

If you want the same information that you have specified in a General Overlay to print on the banner or trailer pages, implement a BPSD tag for the banner or trailer page which contains the same information and appears in the same location as the General Overlay.

Here are some suggested uses for DMGMERGE command Overlays:

- Using a FILEDEF command, place a bar-code on a sheet, which might vary in location or printer type depending on the definition for the output file. Use to implement bar-code changes when the bar-code is in a constant location for all pages for all output files.

  For example, if output is to be tri-folded, and a bar-code needs to go in a particular place, and other output (printed to a different output file) is not folded at all, and the bar-code needs to print in a different location for the second file.

- Using a FILEDEF, MERGE, or GLOBAL command, print a header or other Overlay on the first page only by specifying the FIRST value of the *WHERE* option. (For more information, see "Specifying the Printing Location of an Overlay — the WHERE Option" on page 468.

  For example, the first-page Overlay could contain a name-and-address for mailing, which again, might vary in location and even orientation depending on the output file.

- Using a GLOBAL command, print a header or other Overlay on every page of a test run.

  For example, the Overlay could contain the current run date and time from the DMG.DATE and DMG.TIME Reserved Tags.

For details about specifying GLOBAL, FILEDEF, or MERGE command Overlays, see "xVL=( )" on page 401.

Specifying Dynamic Overlays in VRF Tags

Documerge 3.1 and later versions offer a dynamic Overlay feature that lets you

- Define a VRF tag name that is also a dynamic Overlay name that will contain actual Overlay form name(s).

  For example, the following STRUCTURE RULE code defines a dynamic Overlay with the VRF tag name of OVERLAYTAG:

```
STRUCTURE -
        RULE=((10 DUP OVL=*OVERLAYTAG ))                -
```

  and the OVERLAYTAG has the following values:

Column	Value
01-32	OFRONT
33-37	00000
38	blank
39	F
40-50	blank
51-82	OBACK
83-87	00000
88	blank
89	B
90-100	blank

  Thus, one dynamic Overlay name defined as a VRF tag (beginning with an asterisk) in the Rulebase can point to two or more other Overlay names.

- Use the VDR to generate the contents of the tag (and therefore the Overlay sequence and associated options) just before run time.

  Your VDR can generate the contents of the tag, which can contain up to 1310 Overlay names.

- Use the VDR to specify different page parity, *WHEN*, *WHERE*, and other expanded Overlay options.

  For example, the following STRUCTURE RULE code defines a dynamic Overlay with the VRF tag name of OVERLAYTAG2:

```
STRUCTURE -
    RULE=((10 DUP OVL=*OVERLAYTAG2 ))
```

  and OVERLAYTAG2 has the following values:

Column	Value
01-32	OFRONT
33-37	00000
38	blank
39	F
40-50	blank

  The *F* value in column 39 overrides whatever page-parity Print Option value has been defined for the form in the Rulebase.

  For an explanation of expanded Overlay options, see "Coding a Structure Rule for Expanded Overlay Support" on page 167.

To Set Up a Dynamic Overlay

1. Define a tag name that you can prefix with the word "Overlay". The prefix designates that it is a special dynamic Overlay tag. You can create as many dynamic Overlay tag names as you need.

   In this procedure, examples refer to a dynamic Overlay tag called "Overlay.tag.name".

2. In a Rulebase STRUCTURE RULE, specify the following Print Option:

```
xVL=(*Overlay.tag.name)
```

   Replace the *x* with one of the following:

   - BVL — back side of the page only
   - FVL — front side of the page only
   - OVL — both front and back sides of the page

   For details about coding Overlay statements, see "xVL=( )" on page 178.

   TIP

   You can override any Rulebase-coded Overlay option with a dynamic Overlay, including front/back options.

   Replace the **Overlay.tag.name** placeholder with a name you defined in step 1.

TIP

You can also specify dynamic Overlays using the DMGMERGE SYSIN Overlay parameters, which let you request Overlays in DMGMERGE processing instead of (or in addition to) from the Rulebase.

For details about coding SYSIN Overlay statements, see "xVL=( )" on page 401.

You can also override any SYSIN Overlay parameter with a dynamic Overlay.

3 In a Rulebase Tag table, define the dynamic Overlay tag name and its length. The length must be a multiple of **50**, and each group of 50 bytes contains one Overlay name and its options. Because the maximum number of bytes for a single tag value is 65535, you can code a maximum of 1310 (65535 divided by 50, rounded down) Overlay names per tag.

TIP

You can define multiple occurrences of the same dynamic Overlay tag name to obtain more than 1310 dynamic Overlays if desired. You can also define more than one dynamic Overlay in an Overlay structure, just like you can define more than one Overlay in an Overlay structure.

If the ***Overlay.tag.name** tag occurs more than once in the VRF (or in the DMGMERGE TAG command for the Group), Documerge uses all values from all occurrences, and always starts with the first occurrence.

For details about defining tags, see "" on page 183.

4 To populate the tag with Overlay name values, write the tag values just like you would for any other tag by using the following methods:

- Use the DMGVRFWR program to build the value(s) for the tag.

  For details about DMGVRFWR, see "The DMGVRFWR Subprogram" on page 240.

- In the VDR, in the MSR1 (the Merge Set record of variable data), define an Overlay table which can contain up to 256 entries of 50 characters each. The number of entries should equal the result of dividing the tag length (specified in step 3) by 50.

NOTE

You can code an Overlay tag value with all blanks, which indicates ***No Overlay***, as if no Overlay was requested in the Rulebase or SYSIN.

If any of the Overlay tag values are all blank, then use WIB=Y for these Overlay tags. This tells DMGRFMT to write the tag to the VRF even if it is blank. For details about WIB=, see "WIB=" on page 187.

The entries in this Overlay table override any Overlay options specified in the **DMG.OPT.***groupname* Reserved tag, and also override any Overlay options generated by the DMGRFMT program.

Each 50-character entry has the following format:

Position	Item
01-32	VLAM explicit form member name
33-37	VLAM revision level; must be numeric, leading zeros. All zeros means *highest revision*.
38	The SCOPE value, which can be one of the following: **-** (dash) Don't change; keep Rulebase specification **blank** Don't change; keep Rulebase specification **P** Package scope **D** DTN scope

Position	Item
39	The page parity value, which can be one of the following: **-** (dash) Don't change; keep Rulebase specification **blank** Don't change; keep Rulebase specification **O** Both front and back sides **F** Front side only **B** Back side only
40	**NOTE:** Documerge ignores the *WHEN* option if the *APPLIESTO* value is **LOGICAL PAGES** because the WHEN option is for PHYSICAL or SHEET only, it checks to see if the physical page or sheet has any main (non-Overlay) forms. If there are no main forms, then the page or sheet is considered to be blank. The *WHEN* option value, which can be one of the following: **-** (dash) Don't change; keep Rulebase specification **blank** Don't change; keep Rulebase specification **N** NONBLANK **O** ONLYBLANK **A** ALWAYS
41	The *APPLIESTO* option value, which can be one of the following: **-** (dash) Don't change; keep Rulebase specification **blank** Don't change; keep Rulebase specification **P** Physical page **L** Logical page **S** Sheet
42	The *WHERE* option value, which can be one of the following: **-** (dash) Don't change; keep Rulebase specification **blank** Don't change; keep Rulebase specification **A** ALL **F** FIRST **N** NOTFIRST **L** LAST **T** NOTLAST **M** MIDDLE
43–50	Blanks (hex 40); reserved for future use

5 Before the start of the VDR processing for each new Merge Set, ensure that the VDR moves spaces (hex 40) to the working storage area for the table defined in step 4.

TIP

You can invoke a user-exit program to dynamically build Overlay names by starting the tag name with the string ***DMG.C...***. Documerge uses DMG.C to evoke DMG.C.xxx Reserved tag processing. For details, see "DMG.C.xxx Reserved Tag Processing" on page 331.

Alternatively, you can invoke a user-exit program to dynamically build Overlay names by coding the tag value with a starting three-character **CTAGTRIGGER**. For details, see "GLOBAL Parameters" on page 386.

Dynamic Overlay Coding Examples

The following coding examples illustrate how you would code dynamic Overlay specifications to print an Overlay called *OFRONT* on the front side of a page, and *OBACK* on the back side of the same page.

Example-1. Dynamic Overlays, using the FVL parameter and one VRF tag defined for dynamic Overlays, which also contains an *F* in column 39 and the *B* in column 89 for setting the page parity Print Option:

```
OVL=*OVERLAYTAG
```

Values for OVERLAYTAG:

Column	Value
01-32	OFRONT
33-37	00000
38	blank
39	F
40-50	blank
51-82	OBACK
83-87	00000
88	blank
89	B
90-100	blank

You could have used the FVL= or BVL= parameter instead of the OVL= parameter to define the Overlay; however, we recommend the OVL= parameter.

Example-2. This is the same as example 1, but you would define two instances of the same-named OVERLAYTAG tag and write them to the VRF:

```
OVL=*OVERLAYTAG
```

Values for 1st OVERLAYTAG occurrence:		Values for 2nd OVERLAYTAG occurrence:	
Column	Value	Column	Value
01-32:	OFRONT	01-32:	OBACK
33-37:	00000	33-37:	00000
38:	blank	38:	blank
39:	F	39:	B
40-50:	blank	40-50:	blank

Installing Documerge Trouble Ticket Fixes

On a regular basis, Oracle prepares Trouble Ticket fixes to Documerge programs. These changes can be for reported problems, or internally discovered problems, as well as enhancements. This appendix describes how to install a Trouble Ticket fix for your Documerge software.

For MVS, Oracle always delivers Trouble Ticket fixes as load modules in IEBCOPY format. You will run the IBM IEBCOPY program to recreate a Trouble Ticket fix load library.

As a rule, never directly replace any Documerge program in a production load library, but instead create a new Trouble Ticket fix load library, and concatenate this load library ahead of the production load library in your JOBLIB or STEPLIB. That way, if any other issues arise, Oracle knows you are using a Trouble Ticket fix.

TIP

All Documerge Trouble Tickets have a trouble ticket number, starting with the letters "TT". Include this trouble ticket number in your Trouble Ticket fix loadlib data set name. Sample data set name: `DOCUMERG.V312.TT001234.LOADLIB`

The Trouble Ticket-fix load library can be pre-allocated, or it can be allocated during your IEBCOPY execution. All Documerge loadlibs are **`BLKSIZE=6144,LRECL=0,RECFM=U`**

Usually, only one directory block is required for Trouble Ticket-fix load libraries. After library creation, you can use the TSO 3.4 option to free unused space in the Trouble Ticket-fix load library.

There are two ways Oracle can send the Trouble Ticket fix to you.

- By tape (either cartridge or reel tape)
- By Email (new for Documerge 3.2)

Installing Updates Received by Tape

If you receive a fix by tape, the tape will be unlabeled.

To Create a Trouble Ticket-Fix Load Library Received by Tape

- Run IEBCOPY.

  The output is your Trouble Ticket-fix load library as shown in the following example.

Trouble Ticket-fix Load Library from Tape JCL Example

```
//TAPEFIX   ** put your job card here **
//*
//*     DOCUMERGE V.3.2 -- CREATE IEBCOPY INPUT FILE FOR TROUBLE TICKET
//*      FIXES. THIS JCL ALLOCATES AND CREATES THE OUTPUT PDS FILE
//*
//IEBCOPY   EXEC  PGM=IEBCOPY
//SYSPRINT DD SYSOUT=A
//INFILE   DD  DSN=descriptive.info.for.computer.operator,
//   UNIT=TAPE,
//   LABEL=(1,NL),
//   VOL=SER=volser,       --any volser; might depend on your system--
//   DISP=(OLD,KEEP)
//OUTPDS   DD  DSN=your.Trouble Ticket.fix.load.library;
//   DISP=(NEW,CATLG),
//   SPACE=(TRK,(10,10,1),RLSE),
//   UNIT=SYSDA,
//   DSORG=PO,
//   DCB=(BLKSIZE=6144,RECFM=U)
//SYSUT3  DD  DSN=&&SYSUT3,UNIT=SYSDA,DISP=(NEW,DELETE,DELETE),
//   SPACE=(TRK,(20))
//SYSIN   DD  *
   COPY OUTDD=OUTPDS,INDD=INFILE
/*
//
```

Installing Updates Received by Email

If you receive a fix by Email, it will have a copy of the IEBCOPY input data attached.

To Create a Trouble Ticket-Fix Load Library Received by Email

1. Save the attachment.
2. If the attachment is zipped, unzip the attachment.
3. Transfer the unzipped file to the mainframe.

TIP

The exact transfer procedure depends on your transfer protocol. You may need to contact someone in your informations services area for help

The transfer must be

- binary (no CR/LF)
- no ASCII to EBCDIC translation
- fixed length, unblocked
- record size 80, block size 80

Trouble Ticket-fix Load Library from Email Attachment Example

```
//EMAILFIX   ** put your job card here **
//*
//*  DOCUMERGE V.3.2 -- CREATE TROUBLE TICKET FIX LOADLIB FROM EMAIL ATTACHMENT
//*   THIS JCL ALLOCATES AND CREATES THE OUTPUT PDS FILE
//*
//JOBLIB  DD  DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGJOINR  EXEC  PGM=DMGJOINR
//INJOIN    DD  DSN=your.transfered.unzipped.Email.attachment.file,DISP=SHR
//OUTJOIN   DD  DSN=&&OUTJOIN,DISP=(NEW,PASS),UNIT=SYSDA,SPACE=(TRK,30),RLSE)
//*
//IEBCOPY   EXEC  PGM=IEBCOPY
//SYSPRINT DD SYSOUT=A
//INFILE   DD  DSN=&&OUTJOIN,DISP=(OLD,DELETE),
//OUTPDS   DD  DSN=your.Trouble Ticket.fix.load.library;
//   DISP=(NEW,CATLG),
//   SPACE=(TRK,(10,10,1),RLSE),
//   UNIT=SYSDA,
//   DSORG=PO,
//   DCB=(BLKSIZE=6144,RECFM=U)
//SYSUT3  DD  DSN=&&SYSUT3,UNIT=SYSDA,DISP=(NEW,DELETE,DELETE),
//   SPACE=(TRK,(20))
//SYSIN   DD  *
   COPY OUTDD=OUTPDS,INDD=INFILE
/*
//
```

4. Run the DMGJOINR program, which is described in the next topic. Input is the MVS file created in step 3. The sequential output file can be a temporary (&&) file.
5. Load the file output from step 4 as input to IEBCOPY. The output is your Trouble Ticket fix.

DMGJOINR

The DMGJOINR program re-creates a MVS sequential data set that was "split" into 80-byte fixed length records by the DMGSPLTR program.

You can use the DMGJOINR program when Oracle Emails Trouble Ticket fixes to you. DMGJOINR converts a Trouble Ticket fix you received (as an Email attachment) to the proper IEBCOPY input format.

DMGJOINR JCL Example

```
//DMGJOINR   ** put your job card here **
//*
//*     DOCUMERGE V.3.2 -- CREATE IEBCOPY INPUT FILE FOR TROUBLE
//*      TICKET FIXES
//*
//JOBLIB  DD  DSN=documerg.v03r02.loadlib,DISP=SHR
//*
//DMGJOINR  EXEC  PGM=DMGJOINR
//INJOIN    DD  DSN=file.created.by.program.dmgspltr,DISP=SHR
//OUTJOIN   DD  DSN=your.output.file,DISP=(NEW,CATLG),
//  UNIT=SYSDA,SPACE=(TRK,(30,30)),RLSE)
//
```

DMGSPLTR

The DMGSPLTR program creates the input used by DMGJOINR by separating a MVS sequential data set into 80-byte fixed-length records. DMGSPLTR also creates a header that describes the DCB (LRECL, BLKSIZE, and RECFM) for the output file that contains the records.

For example, to send a fixed Trouble Ticket program via Email, Oracle does the following:

(1) Runs IBM's IEBCOPY program to create a sequential file backup of the PDS modules.

(2) Runs DMGSPLTR to create the 80-byte fixed length records.

(3) Emails the resulting DMGSPLTR output file to the Documerge user.

The user who receives the email runs DMGJOINR to recreate the IEBCOPY backup file, and then runs IEBCOPY to recreate the load library.

Users can also employ DMGSPLTR to transfer large MVS sequential files to Oracle in one or more Emails.

CAUTION!

Your Email system probably has an upper limit on the size of the Email attachment you can send (typically 5MB or less, but the actual size varies). Therefore, a very large file usually cannot be sent via Email.

DMGSPLTR JCL Example

```
//DMGSPLTR   ** put your job card here **
//*
//*    DOCUMERGE V.3.2 -- CREATE 80-BYTE RECORDS TO EMAIL FILES TO
DOCUCORP
//*
//JOBLIB  DD  DSN=documerg.v03r.2.loadlib,DISP=SHR
//*
//DMGSPLTR  EXEC  PGM=DMGSPLTR
//INSPLIT  DD  DSN=sequential.file.to.be.sent,DISP=SHR
//OUTSPLIT DD  output.file.to.be.Emailed,DISP=(NEW,CATLG),
//  UNIT=SYSDA,SPACE=(TRK,(30,30)),RLSE)
//
```

Index

C

D

E

F

G

H

I

J

K

L

M

N

R

S

T

U

V

W

X

Y

Z