precisely

EngageOne Enrichment

Reprint User Guide

Version 7.4

Table of Contents

1 - Reprint overview

In this section

What is Reprint?

Reprint enables you to extract selected documents or pages from an existing EngageOne Enrichment output or other print stream and place the extracted data into one or more output streams. Your input to Reprint is not changed.

Reprint provides a quick solution when only a portion of a large print run requires reprinting. For example, Reprint is useful in the following situations:

- When specific documents or pages must be reprinted due to low print quality, printer problems, or damage in finishing or envelope insertion.
- When quality control procedures require sample sets of documents or pages to be printed. Reprint allows you to reprint a range of pages, one or more documents, or a sampling of documents or pages as necessary. You can even reprint every nth document.
- When individual documents are required for audits. You can specify specific documents to print on demand.
- When the security of original documents must be ensured. Because Reprint does not alter its input, many people can use it to print the documents or pages they need, without fear of changing original documents.
- With Reprint, you can reprint documents and pages from any print stream format (including line printer, AFP line data, AFPDS, MO:DCA, Xerox DJDE, Xerox Metacode, PCL or PostScript).

How is Reprint different from Enrichment?

Reprint and Enrichment both use a control file and a rule file to control processing. However, there are fundamental differences between the products in the purpose and uses of these files.

- Enrichment modifies and enhances print streams, while Reprint extracts pages from an existing print stream.
- Reprint cannot change the content or layout of a print stream. In Enrichment, variable values are generally set from data on the input pages using the Field tag group. In Reprint, all variable values that come from the Reprint Index fields are not available.
- The control file in both Enrichment and Reprint controls the overall processing flow. At a minimum, Enrichment requires only the control file and one input print stream to create an output stream, while Reprint requires the control file, a source print stream, a Reprint Index, and a rule file to create an output stream.
- In Enrichment, the optional rule file is generally used to determine the type of processing to perform on each document within a print stream. In Reprint, the rule file contains the selection criteria by

which documents or pages are extracted from a print stream. There are no specific rule file sections nor is there a pagerule file in Reprint.

Because of these differences, experienced Enrichment users should pay particular attention to the following sections in this reference:

- The <OUTPUT> command is used differently in Reprint rule files. For information on the <OUTPUT> command, see **<OUTPUT> statements** on page 27.
- Two types of output controls not available in Enrichment, <DOCPAGES> commands and PAGERANGE definitions, can be used in the Reprint rule file. Reprint is compliant with all other rule file structures, including math operations, concatenation, and function calls. For more information on these two controls, see **<DOCPAGES> statements** on page 28 and **PAGERANGE definitions** on page 28.
- The Reprint Output tag group is a subset of the Enrichment Output tag group. Output group tags that in Enrichment change content or layout are not available in the Reprint Output tag group. For more information, see **Output Tag group** on page 38.
- The ability to create multiple outputs is common to both Enrichment and Reprint.
- In both Enrichment and Reprint, you can create any number of side files. The Reprint Index used by Reprint can be a side file from a previously run Enrichment application. Since Reprint can produce side files, you can use these side files as the Reprint Indexes in subsequent Reprint applications.

Sample Reprint application

The following application is a very simple Reprint example. Two control files are required to run this sample: `bankdemo.con` and `bankdemrepr.con`. The first control file (`bankdemo.con`) is the initial Enrichment job which creates the index file `repridx.txt` using the *%%AccountNumber* field.

> **Note:** While there are other fields in the document, such as *%%Address1*, these are not used for the index file in this example. If you needed to create an index using these fields they could be added to the <SIDEFILE> tag group.

The second control file (`bankdemrepr.con`) reads the original production print file and selects documents to reprint based on account number.

`Bankdemo.con`

```
<! ------------------------------------------------------- >
<!   Control File: bankdemo.con                            >
<!   Purpose: Create a reprint index file from line data.  >
<!   Customer: Any customer that uses Reprint    >
<! ------------------------------------------------------- >

<INPUT>
  <NAME>INPUT1
  <FILE>'BANKDEMO.LIN'
```

```
  <type> I AA
  <DOC> T %%AccountNumber C
  <document> T %%PageIndicator = 1
  <FIELD>%%AccountNumber
    <REF> ' ' 'Account Number:' 47 E
    <LOC> 0 2 7
    <textuntil> S ' '
  </FIELD>
  <FIELD>%%Address1
   <loc> 14  16  25
  </FIELD>
  <FIELD>%%Address2
   <loc> 15  16  25
  </FIELD>
  <FIELD>%%Address3
   <loc> 16  16  25
  </FIELD>
  <FIELD>%%PageIndicator
   <REF> ' ' 'PAGE' 61 E
   <LOC> 0  2  1 LINE 7
  </FIELD>
</INPUT>

<! ------------------------------------------------------------ >
<! Write output and create reprint index sidefile              >
<! %%OUT_RECORD AND %%TOTAL_RECORDS are Required Fields for   >
<! reprint                                                     >
<! ------------------------------------------------------------  >

<output>
   <name>OUTPUT1
   <file>bankdemo.out
   <sidefile> 'repridx.txt'
      <sidepart> %%AccountNumber 7
      <sidepart> 1
      <sidepart> %%OUT_RECORD         3 R 0
      <sidepart> %%TOTAL_RECORDS      3 R 0
      <sidepart> X'0D'
   </sidefile>
</output>
```

`Repridx.txt`

The print index file is a flat file containing field information as well as record information used to retrieve documents. The <SIDEFILE> created in the original Enrichment job created this reprint index. The Reprint control file reads the index to reference requested documents.

```
48184151000066
48187001066132
48189251198057
```

`Bankdemrepr.con`

```
<! ------------------------------------------------------------ >
<!   Control File:  bankdemrepr.con                            >
<!   Purpose:       Reprint based on Account Number            >
<!   Customer:      Any Reprint Customer          >
<! ------------------------------------------------------------ >
<!   Input Data file from original Enrichment Run            >
<!   Associated Enrichment Control file name is bankdemo.con>
<! ------------------------------------------------------------ >
<REPRINT>
   <NAME> INPUT1
```

```
   <FILE> 'bankdemo.out'
   <TYPE> I AA
</REPRINT>

<! ------------------------------------------------------- >
<!   Definition of the index file created by bankdemo.con     >
<! ------------------------------------------------------- >
<INDEX>
   <FILE> 'repridx.txt'
   <VAR> %%AccountNumber  1   1  7 R
   <VAR> %%OUT_RECORD     1   9  3 R
   <VAR> %%TOTAL_RECORDS  1  12  3 R
</INDEX>


<! ------------------------------------------------------- >
<!   Rule File to select Account number to reprint            >
<!   Selected document(s) gets written to OUTPUT1             >
<! ------------------------------------------------------- >
<RULE>
   <CONTENT>
   IF %%AccountNumber = 4818700 THEN
       <OUTPUT> OUTPUT1
   ENDIF
   </CONTENT>
</RULE>
<! ------------------------------------------------------------>
<!   Output file where selected document(s) are written       >
<! ------------------------------------------------------------>
<OUTPUT>
   <NAME> OUTPUT1
   <FILE> 'select.lin'
</OUTPUT>
```

2 - Running Reprint

In this section

Input and Output

Reprint requires four files for input:

1. One or more source print streams. The source print streams can be output from Enrichment or from any other application as long as the print streams are in AFPDS, AFP line data, AFPmixed data, impact, DJDE, metacode, PCL format, or PostScript.
2. A flat file, called the Reprint Index, that indexes the documents and pages in the source print streams. The Reprint Index contains one or more lines of information about each document in each input print stream. The only required information is the record number after which each document begins in the print streams, and the total number of records in each document. Reprint can use any other information in the Reprint Index as criteria to select the document to extract. Other information, such as beginning and ending page counts, may be required for specific Reprint functions. For more information on creating a Reprint Index, refer to the *Enrichment Language Reference Guide*.

   Note: The records in the Reprint Index file need to be in the same sequence as the documents in the input print stream.

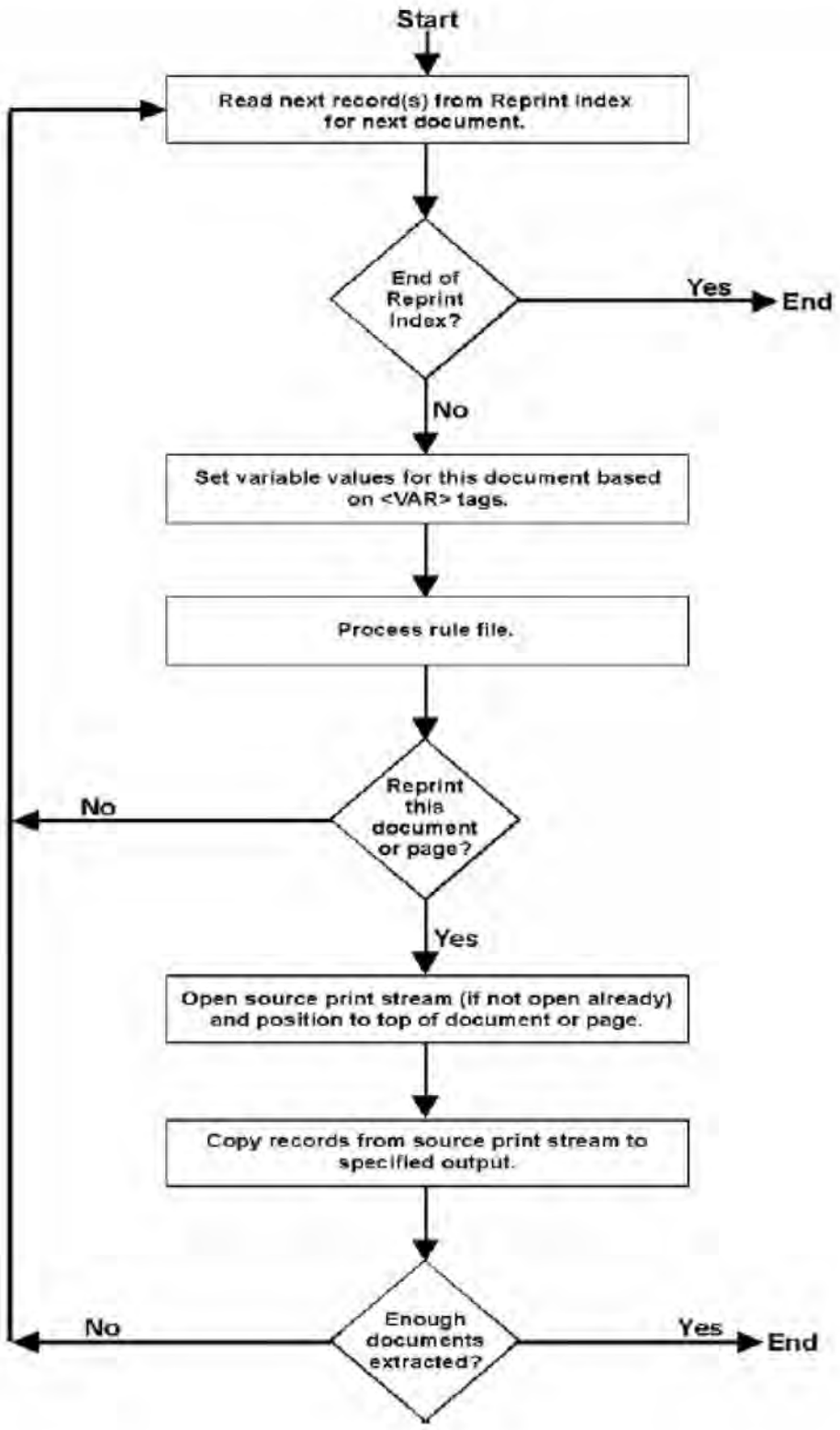
   The Reprint Index is easily created during Enrichment processing by using the Sidefile tag group. Enrichment writes one record to the Reprint Index for each document.
3. A Reprint control file. For more information on the control file, see **The Control file** on page 30.
4. A rule file. For more information on the rule file, see **Rule File overview** on page 16.

From these inputs, Reprint creates a Reprint report and one or more output streams.

Optional outputs from Reprint include multiple side files for each output that contain user-specified information about each document in the matching output.

Processing flow

Reprint is optimized for highly efficient document extraction. As shown in the diagram below, Reprint only reads the Reprint Index to determine which documents or pages to reprint. It does not analyze the source print stream to make this determination. Reprint only reads the source print stream when extracting a document or page. The rule file controls which documents and/or pages are extracted.


Start
Read next record(s) from Reprint index for next document.
End of Reprint Index?
Yes
End
No
Set variable values for this document based on <VAR> tags.
Process rule file.
Reprint this document or page?
No
Yes
Open source print stream (if not open already) and position to top of document or page.
Copy records from source print stream to specified output.
Enough documents extracted?
No
Yes
End

Run requirements

On mainframe

Reprint on mainframe systems requires JCL to run. Sample Reprint JCL is shown below.

```
//*your job card
//*****************************************************************
//** This is a sample Reprint job.
//** Change SYS3.C370 to the high level qualifier for C/370 runtime.
//** Change PDR.STREAMW to the high level qualifier for Enrichment.
//** Change all DDs to reference your particular data set names.
//*****************************************************************
//JOBLIB DD DSN=PDR.STREAMW.LOAD,DISP=SHR
//       DD DSN=SYS3.C370.SEDCLINK,DISP=SHR
//       DD DSN=SYS3.C370.SIBMLINK,DISP=SHR
//*
//*****************************************************************
//*  This step runs Reprint.
//REPRINT EXEC PGM=PDRR0000
//*  Reprint control file can be either instream data or
//*  a separate data set as the example show.
//CONTROL  DD DSN=control.file,DISP=SHR
//*
//*  Input print stream for Reprint job.
//INPUT1   DD DSN=input.data,DISP=SHR
//*
//*  You must have a Reprint RULE file.
//RULE     DD DSN=rule.file,DISP=SHR
//*
//*  Reprint output enhanced print stream
//OUTPUT1  DD DSN=output.file,DISP=SHR
//*
//*  Reprint Summary Report and Messages
//REPORT   DD DSN=message.summary,DISP=SHR
//
```

On UNIX and Windows

Reprint on UNIX and Windows requires scripts to run.

Windows script

To run Reprint on Windows, run the following command:

```
reprint -c=control.file -m=message.summary
```

Unix Script

```
# This script links the file input.data to the pseudo-file DD:INPUT1
#     then executes Reprint (called reprint in this example)
#     with the command options -c=control.file and -m=message.summary
#     A file DD:OUTPUT1 is created and then renamed to output.file.
```

```
#
# This example assumes that all files are in the current directory
#     of execution and that reprint is in the execution path of the
#     current user.
#
ln input.data DD:INPUT1
reprint -c=control.file -m=message.summary
mv DD:OUTPUT1 output.file
```

Run-time arguments

You can provide arguments to Reprint at run time to supplement or override tag values specified in the control file. Specify these optional run-time arguments (also called switches) in the Reprint JCL PARM statement. Code the PARM statement as follows:

```
PARM='Cruntime//switch'
```

where:

Cruntime	One or more C/370 run-time options to use during Reprint processing. Often, there are no C/370 run-time options specified. Refer to the *IBM C/370 Programming Guide* for more information on C run-time options.
switch	A Reprint run-time argument. Each Reprint switch must begin with a slash (/) (in addition to the initial slash required to separate the C/370 run-time arguments from program arguments).

The following table lists and explains valid Reprint run-time switches.

Argument	Description
/C=dsname	Specifies the control file to use for Reprint processing. If you do not use this switch, Reprint assumes the control file is DD:CONTROL. If you have not defined DD:CONTROL on your system, processing stops.

Argument	Description
/E=level	Specifies the level of error at which Reprint processing stops, where level is one of the following: **S** Stops processing only if Reprint issues a severe message. **W** Stops processing if Reprint issues a warning or severe message. **E** Stops processing when Reprint issues an informational, warning, or severe message. If you do not use this switch, Reprint processing stops only when a severe message occurs.
/L=length	Specifies the width, in characters, in which Reprint outputs messages, where length is one of the following: **S** Messages break to the next line every 80 characters. **L** Messages break to the next line every 132 characters. If you do not use this switch, Reprint breaks messages to the next line every 80 characters.
/M=dsname	Specifies the file into which Reprint places processing messages and reports. If you do not use this switch, Reprint assumes the message file is DD:REPORT. If you have not defined DD:REPORT on your system, Reprint routes messages to JES output (as if you specified SYSOUT=*) and generates a warning.
/R=print	Specifies which messages Reprint should include in the processing report, where print is one of the following: **N** Report no messages. **S** Report only severe messages. Severe messages reflect conditions that cause Reprint processing to stop. **W** Report warning and severe messages. Warning messages reflect error conditions that need not cause Reprint to stop, but may produce unexpected processing results. **I** Report information, warning and severe messages. Reprint displays information messages to indicate progress during processing. Information messages require no corrective action. **P** Report processing, information, warning, and severe messages. Processing messages are more specific than information messages, and may be helpful as troubleshooting aids. **D** Report detail, processing, information, warning, and severe messages. Detail messages present trace-level information.

Argument	Description
/S=switch	Specifies whether Reprint places message numbers in the reports it generates, where switch is one of the following: **E** Message numbers appear in the report. **N** Message numbers do not appear in the report. If you do not use this switch, Reprint places processing, information, warning, and severe messages in the report. The document *Enrichment Messages Reference Guide* lists messages by number. If you let /S default to Y, it is easier to find message explanations in that book.
/T=level	If you do not use this switch, Reprint assumes /S=Y. Specifies the level of trace information to include in the Reprint Report, where level is one of the following: **I** Include an intermediate amount of trace information. **F** Include full trace information. This switch is useful for troubleshooting, especially if there appear to be problems with Reprint output. However, using /T can affect performance and creates very large Reprint reports.
/V	Validates the control file for correct tagging and syntax, but does not process input or output files. This switch is useful for troubleshooting.

3 - The Rule file

In this section

Rule File overview

With Reprint, rules are used to select which documents and pages to extract, to set or change variable values, and to control which output each document or page is routed to.

You can store rules for use with Reprint applications in two places:

- In a rule file identified to Reprint by the Rule group <FILE> tag. For more information, see **<FILE>** on page 49.
- Within a Content tag group in the Rule group. For more information, see **Content Tag group** on page 35.

Reprint processes rules once per document. If you use the Content tag group rather than a rule file, simply include your rules between the <CONTENT> and </CONTENT> tags as if you were creating a rule file. The syntax of Reprint rules is explained in this chapter. Rule syntax is the same in the Content tag group as in a rule file.

The Reprint rule file consists of the following:

- Comments
- Set statements
- Conditions
- <OUTPUT> statements
- <DOCPAGES> statements
- PAGERANGE definitions
- Declarations for user-written functions, explained in the *Enrichment Language Reference Guide*.

Comments

Use comments in your rule file to include information about the functions performed. All rule comments must begin with the characters /* and must end with the characters */. The following figure shows a rule file that contains comments in the header and in the body of the file:

```
/*************************************************************/
/*  System: Reprint                        */
/* Version: 5.0                                              */
/* Purpose: Example rule data set for separating output      */
/*          based on a user-defined variable in a rule file. */
/* (c) Copyright 1995-2013 Precisely                 */
/*************************************************************/
IF %%Area_Code = 502 THEN           /* Determine whether document*/
   <output> AreaCode502 * 1         /*  goes in AreaCode502 or   */
```

```
ELSEIF %%Area_Code = 606 THEN       /*  in AreaCode606 output.   */
   <output> AreaCode606 * 1 ENDIF
```

Reprint ignores text or symbols between the comment begin and end characters. You can nest comments, but the number of comment begin sequences must always equal the number of comment end sequences. You can place comments anywhere in a rule file, but you should devise a standard way of commenting the files. Each rule file should begin with comments that explain its purpose.

> **Note:** Rule file command and control characters are shown in this guide using a U.S. character set (code page). Your actual characters may differ if you do not use a U.S. code page.

Comments can be included in any of the following common programming styles:

/* comment text */	Comments can span multiple lines up to the closing "*/". This is helpful if you need to temporarily comment out a block of lines for testing.
<! comment text >	Comments can span multiple lines up to the first closing ">".
// comment text	Comments the rest of the current line after the two slashes.

All three comment styles work in both the rule file and the control file. To comment out blocks of code, use the /**/ style, since the tag syntax <tag> may lead to incorrect closing of the <!> block of comments.

Set statements

Set statements assign a value to a variable based on the value of another variable, or based on the results of a function call, math operation, or condition. You can use set statements in a rule file outside of the conditions or as the results of conditions.

When you place set statements outside conditions in the rule, Reprint sets their values automatically during processing in the order they appear in the rule file.

When you use one or more set statements in the result of a condition, Reprint sets their values when the condition is evaluated as true. Set statements in the result of a condition occur after a THEN or ELSE condition in the rule file.

Reprint rules can contain four types of set statements:

- Simple set statements
- Math operations
- Function calls

- Concatenations

Simple set statements

A simple set statement sets a variable to constant value or to another variable. Simple set statements have the following syntax:

```
%%varname = constant
```

or

```
%%varname = %%varname2
```

You can also use simple set statements within conditions to evaluate variable values. For example:

```
IF %%age = 21 THEN %%eligibility = Yes
```

In the example, `%%eligibility = Yes` is a simple set statement that sets a variable as the result of a condition (`IF %%age = 21`).

To set a variable to nothing, use a blank character within quotation marks (' '). You must delimit variable values that contain spaces with single quotes (`'908 West 7th St.'`) or double quotes (`"908 West 7th St."`). If the value contains single or double quotes, you must double the internal quote marks (`'Guy''s Auto Hut'` or `"Bill's ""Hi-Fi"" Stereos"`) or Reprint assumes they terminate the value.

Math operations

Reprint rules use math operators to carry out arithmetic on numbers or numeric variable values. The following table lists and briefly describes the valid Reprint math operators. Reprint evaluates mathematical expressions from left to right according to the operator precedence indicated in the table. If a mathematical expression contains parentheses, Reprint evaluates expressions within the parentheses first.

Operator	Description	Precedence
+	Addition	3
-	Subtraction	3
*	Multiplication	2

Operator	Description	Precedence
%	Integer division. Returns the integer result when one value is divided by another.	2
#	Remainder. Returns only the remainder when one value is divided by another.	2
( )	Open and close parentheses.	1

You can nest parentheses as deeply as necessary, as long as every open parenthesis has a complementary closing parenthesis. The following table shows how the results of Reprint math evaluation can differ based on the evaluation order.

Expression	Result
(2 + 5) * 10 - 4 % 2	68
(2 + 5 * 10) - 4 % 2	50
2 + 5 * 10 - 4 % 2	50
2 + 5 * (10 - 4 % 2)	42
2 + (5 * 10 - 4) % 2	25
(2 + 5) * (10 - 4) % 2	21
2 + (5 * (10 - 4)) % 2	17
(2 + (5 (10 - 4))) % 2	16

Function calls

Reprint provides a rich set of functions for use in the rule file for string manipulation, conversion, formatting, and file I/O. Some examples of functions are:

CHECKSUM	Calculates a checksum for a specified string.

COMPARE	Evaluates a specified number of characters from two strings to determine if they are the same.
COPIES	Returns a specified number of copies of a string.
DATE	Returns the current date in the specified format.

For a complete list of Enrichment functions, see the *Enrichment Language Reference Guide*.

What are Function calls?

Function calls are used in the rule file and have the following set statement syntax:

```
%%answer = function(arg1, arg2,…)
```

The data and options passed to a function are called arguments. Arguments can be any of the following:

- A constant string. If the string contains blanks, you must enclose the string in single or double quotation marks. If the string contains quotation marks, surround it with single or double quotation marks and double the internal quotation marks.
- A constant numeric value.
- A constant hexadecimal value.
- A constant binary value.
- A variable of any type valid for Reprint.

Some arguments have default values. If you do not explicitly specify such an argument, Reprint uses its default value. For example, if you specify a function call as follows:

```
%%answer = FUNCTION(arg1,,arg3)
```

Reprint sets the value for *arg2* to its default. Note that no value was entered for *arg2* between the parentheses. Likewise, you could set *arg2* and *arg3* to their default values as follows:

```
%%answer = FUNCTION(arg1)
```

No separators (that is, commas) are required in the above function call since only the first argument is specified. If a function is called using only default argument values, or if there are no arguments, the parentheses are still required. For example:

```
%%answer = FUNCTION()
```

Functions return their answer to a variable. In the example below, REVERSE is the name of the function (which reverses the content of the argument). The argument is a variable called *%%Barcode*. The answer is stored back into the original variable.

```
%%Barcode = REVERSE(%%Barcode)
```

Rule file compilation

Reprint automatically evaluates and compiles the rule file before processing the first document. This greatly increases performance.

During compilation, Reprint evaluates functions to ensure that proper variable types are used, all required arguments are specified, and all constant values are valid. If Reprint encounters errors, it generates messages and stops processing.

Function return codes and return values

In addition to returning an answer in a variable, functions also return data in a return code and a return value.

Return codes are integer values that indicate the success or failure of the function call. While each function may set specific return codes based on its result, all such functions follow this convention:

- A return code of 0 indicates that the function was successful.
- A negative return code indicates that invalid arguments were used or a severe error occurred.
- A positive return code indicates that the function was not successful or was partially successful.

Reprint places the return code from each function call in the *%%RC* system variable. *%%RC* contains the return code from the most recent function call. As shown below, you can use *%%RC* in a rule to specify alternate processing.

```
%%AMOUNT = FINDNUM(%%LINE1) IF %%RC <> 0 THEN  %%AMOUNT = FINDNUM(%%LINE2)  IF %%RC <> 0
 THEN   %%AMOUNT = 12  ENDIF ENDIF
```

Return values provide data that supplements the answer returned from the function call. The data stored in the return value depends on the function. For example, the return value from FINDZIP identifies what type of ZIP Code was found (ZIP, ZIP+4, or ZIP+4+2). Reprint stores the actual ZIP Code in the return. Reprint stores return values in the *%%RV* system variable. As shown below, you can use *%%RV* in the rule file in the same way you use *%%RC*.

```
%%ZIP = FINDZIP(%%ADDRESS)
/*FIND ZIPCODE  */ IF %%RV = 5 THEN
/*OUTSORT 5 Digit  */  <OUTPUT> 5DIGIT ELSE  <OUTPUT> ZIP4 ENDIF
```

Concatenation

You can use a concatenation character (|) as follows to concatenate two or more strings or variables into a single variable as shown below.

Note: You can use either a solid vertical bar (|) or a broken vertical bar (¦) as a concatenation character.

```
%%varname = value1 | value2 | value3 |...
```

where:

%%varname	Is a variable name whose value is the result of the concatenation. *%%varname* can be up to 50 characters in length (including %%) and cannot include spaces.
value1, *value2*, and so on	Are the values to concatenate. These can be variable names whose values are set elsewhere or constants. Reprint does not place spaces between concatenated values, so you must specify spaces surrounded by single quotation marks as necessary to concatenate blanks into the value. If you specify a variable as one of the values to concatenate, leave a space between the *%%varname* and the concatenation character so Reprint does not include the concatenation character as part of *%%varname*.

Conditions

Conditions are combined with set statements and <OUTPUT> statements to create rules. The following table lists the conditions valid for Reprint rules.

Condition	Explanation

IF/ENDIF	IF begins a rule or a series of rules. If Reprint evaluates one or more of the rules as true, THEN a result happens. Each IF requires a matching ENDIF to terminate the rule or series of rules. For example:
	`IF %%age >= 21 THEN <OUTPUT> Eligible * 1`
	`ENDIF`

ELSEIF	ELSEIF offers an alternative to the rule or series of rules given by IF. If Enrichment evaluates one or more of the rules as true, THEN a result happens. For example:
	`IF %%age >= 21 THEN <OUTPUT> Eligible *`
	`1  ELSEIF %%military = Yes THEN <OUTPUT> Eligible * 1`
	`ELSE <OUTPUT> NotEligible * 1`
	`ENDIF`

Condition	Explanation

THEN	THEN defines the result of an IF or ELSEIF rule. For example:
	`IF %%age >= 21 THEN <OUTPUT> Eligible * 1`
	`ELSEIF %%military = Yes THEN <OUTPUT> Eligible * 1`
	`ELSE <OUTPUT> NotEligible * 1`
	`ENDIF`

ELSE	ELSE defines the result if Enrichment evaluates no IF or ELSEIF rules as true. For example:
	`IF %%age >= 21 THEN <OUTPUT> Eligible * 1`
	`ELSEIF %%military = Yes THEN <OUTPUT> Eligible * 1`
	`ELSE <OUTPUT> NotEligible * 1`
	`ENDIF`
	Note: An ELSE is not required for each IF.

You can use conditions to make a simple comparison like this:

```
IF %%age = 21 THEN...
```

or with multiple comparisons that are grouped with the logical connectors AND and OR and that use parentheses ( ) to enclose information:

```
IF %%age = 21 AND (%%smokes = yes OR state @ 'KY WV OH') THEN...
```

Each comparison contains a variable on the left, a comparative operator, and a value or variable on the right. It can contain more than one of each.

Comparison operators

A comparison operator compares a variable with a value (or another variable) to obtain a true or false result. The comparison operators listed in Table 3-5 are valid for all comparisons in a rule file.

Operator	Name	Explanation

>	Greater than	The variable value is greater than the value to which it is compared, as in:
		`IF %%age > 21 THEN… or`
		`IF %%name > Martin THEN…`

Operator	Name	Explanation

<	Less than	The variable value is less than the value to which it is compared, as in:
		`IF %%age < 21 THEN… or`
		`IF %%name < Martin THEN…`

>=	Greater than or equal to	The variable value is greater than or equal to the value to which it is compared, as in:
		`IF %%age >= 21 THEN… or`
		`IF %%name >= Martin THEN…`

<=	Less than or equal to	The variable value is less than or equal to the value to which it is compared, as in:
		`IF %%age <= 21 THEN… or`
		`IF %%name <= Martin THEN…`

=	Equal to	The variable value is equal to the value to which it is compared, as in:
		`IF %%age = 21 THEN… or`
		`IF %%name = Martin THEN…`

!= \= <>	Not equal to	The variable value is not equal to the value to which it is compared, as in:
		`IF %%age != 21 THEN… or`
		`IF %%name \= Martin THEN…`

@	Included in	The variable value is equal to some part of the string inside the quotation marks, as in:
		`IF %%age @ '16 18 21 40' THEN… or`
		`IF %%state @ 'KY CA NC CT' THEN…`
		When @ is used in a comparison, Reprint uses any subset of the characters or blanks enclosed in the quotation marks as a separate value by which to compare. For example, the above condition is true if %%state equals any of the following:
		`K KY  C Y  C KY  CA CA`
		and so on.

Operator	Name	Explanation

@=	Case-sensitive
	Similar to with Wild Cards
		@= enables you to specify approximate comparison values by using * or $ as wild card characters to broaden the effect of the operator. For example:
		`IF %%STATE @= 'K*' THEN…`
		is true if the value of %%STATE begins with an uppercase K and is followed by any other character or characters. Likewise, the following syntax:
		`IF %%STATE @= 'K$' THEN…`
		is true if the value of %%STATE begins with an uppercase K and is followed by any other single character.
		@= is case sensitive. Therefore, the capitalization of the variable values it returns matches the rule file entry exactly.

*=	Non-case-sensitive Similar to with Wild Cards	Identical to @=, except that the capitalization of the variable values returned need not match the rule file entry exactly.

Comparison values

Comparison values are compared with variable values to obtain a result. A comparison value can be a literal string, a number, or another variable.

Literal strings

If you do not specify <CHARACTERS> in your Environment group, Enrichment uses the binary values of characters to compare strings. On mainframe systems, characters are represented by the EBCDIC character set. In EBCDIC, the lowercase characters have lower binary values than the uppercase characters.

- On UNIX and Windows, characters are represented by the ASCII character set.
- In ASCII, the lowercase characters have greater values than the uppercase characters. For example, the word cat would compare as shown in the following figure.

Note: Trailing blanks are ignored when comparing strings. Therefore, 'ABC ' equals ABC.

As the following figure shows, the EBCDIC hexadecimal value of the word cat may be X'8381A3', X'C381A3', or X'C3C1E3' while the ASCII hexadecimal value of the word may be X'636174', X'436174', or X'434154', each depending upon how the word is capitalized.

cat < Cat < CAT	cat > Cat > CAT
88A C8A CCE	667 467 445
313 313 313	314 314 314
EBCDIC Hexadecimal Translation	ASCII Hexadecimal Translation

You can use the <CHARACTERS> tag to avoid the hexadecimal translation differences and to enable proper comparison of international characters.

Numbers

Numbers are always compared as numbers. Field variables are considered numeric if they only contain blanks or numbers. Leading zeros are ignored, therefore, the string ’0005’ equals 5.

Variables

You can compare variables with other variables. For example:

- `IF %%age = %%retirement_age THEN...`

  or

- `IF %%Rider \= %%Rider1 THEN...`

Connectors

You can also connect comparisons to make a result depend on the values of more than one comparison. The two connectors valid in Reprint rule files are **AND** and **OR**.

AND is used when more than one comparison must be true to obtain a result. For example:

```
IF %%age = 21 AND %%smokes = yes THEN...
```

To obtain the result, *%%age* must be set to 21 and *%%smokes* must be set to *yes*. If both comparisons are true, the result is obtained. If either of the comparisons is not true, the result is not obtained.

OR is used when one of a series of comparisons must be true to obtain a result. For example:

```
IF %%age = 21 OR %%smokes = yes THEN...
```

To obtain the result, either *%%age* must be set to *21* or *%%smokes* must be set to *yes*. If either comparison is true, the result is obtained. If neither comparison is true, the result is not obtained.

You can nest as many of these connectors as you want to form complex rules, and you can use parentheses to group comparisons and connectors in any order or arrangement, as long as a comparison precedes and follows each connector. If you do not use parentheses, Reprint processes

comparisons and connectors from the beginning of the IF or ELSEIF condition to the THEN condition. For example,

```
IF %%age = 21 AND %%smokes = yes OR %%state @ 'KY WV OH' THEN...
```

To make the above rule true, either *%%age* must be set to *21* and *%%smokes* must be set to *yes*; or *%%state* must be set to any sequential part of *'KY WV OH'*.

If we add parentheses to the rule, for example:

```
IF %%age = 21 AND (%%smokes = yes OR %%state @ 'KY WV OH') THEN...
```

The rule is true only if *%%age* is set to *21* and either *%%smokes* is set to *yes* or *%%state* is any sequential part of *'KY WV OH'*.

<OUTPUT> statements

<OUTPUT> statements in the Reprint rule file specify an output file in which to place extracted documents and/or pages. <OUTPUT> statements have the following format:

<OUTPUT> *name maximum frequency*

where:

Parameter	Description
name	Duplicates the name of a Reprint Output tag group.
maximum	Specifies the maximum number of documents to place in the output. Use * if Reprint is to place all documents resulting from the rule into the output. * is the default maximum setting.
frequency	Specifies which documents, that result from the rule, Reprint places in the output. For example, if you set frequency to 1, Reprint places every document that results from the rule in the output, up to the number specified by maximum. If you set frequency to 10, Reprint places every tenth document that results from the rule in the output, up to the number specified by maximum.

For example, as a result of the rule shown below, Reprint places up to three five-page documents in the output `Five_Pagers`. Reprint places the tenth, twentieth, and thirtieth five-page documents

into `Five_Pagers` and stops processing. If there are only 25 five-page documents, Reprint places the tenth and twentieth documents in the output and stops at document 25.

```
IF %%TOTAL_PAGES = 5 THEN
     <OUTPUT> Five_Pagers 3 10
ENDIF
```

<DOCPAGES> statements

<DOCPAGES> statements in the Reprint rule file specify specific pages to place in output from each document. <DOCPAGES> statements have the following syntax:

<DOCPAGES> *first last*

where:

Parameter	Description
first	Specifies the first page of the document you want placed in output.
last	Specifies the last page of the document you want placed in output. Set last to * to place to the end of the document.

For example, as the result of the rule shown below, Reprint would place from page 2 to the end of all `five-page` documents in the output `Five_Pagers`.

```
IF %%TOTAL_PAGES = 5 THEN
  <OUTPUT> Five_Pagers * 1
  <DOCPAGES> 2 *
ENDIF
```

PAGERANGE definitions

PAGERANGE definitions in the Reprint rule file specify an exact range of pages from an input to place in an output. PAGERANGE definitions are used within IF/THEN or ELSEIF/THEN conditions and have the following syntax:

PAGERANGE (*begin,end,value*)

where:

Argument	Description
begin	Specifies the beginning count of the page range (greater than or equal to the `%%OUT_TOT_L_PAGE` value).
end	Is one of the following: • The user-assigned value associated with the input from which to print the pages in the range. • The `%%OUT_FILE` value associated with the input from which to print the pages in the range. • Blank if there is only one input to Reprint.

Note: For information about the %%OUT_FILE, %%OUT_TOT_L_PAGE, and %%OUT_TOT_L_PAGE_BEGIN system variables, see **Index Tag group** on page 35.

For example, you might code a rule that resembles the one shown below to place pages 2000 through 2200 of a single input into the output FINAL_OUT.

```
IF PAGERANGE(2000,2200) THEN
  <OUTPUT> FINAL_OUT
ENDIF
```

Pages are determined only by carriage control '1'; therefore, pages defined by user definition or other printstream definitions are not recognized. In this case, only entire documents can be selected.

4 - The Control file

In this section

Overview

To use Reprint, you must create a control file that contains instructions about the files to process and how to process them. The Reprint control file contains tag groups, tags, and values that define a Reprint application.

Tag groups are subsets of instructions within the control file, each of which has a specific function. In order to define how Reprint performs a particular function, you must define values for the tags that make up the tag groups. Some tags are required, meaning that you must supply a value for them. Others are optional, meaning that if you do not specify the tag in the control file, Reprint supplies an assumed, or default, value for it. Some tag groups must be used within other tag groups, and others can be used alone or within a tag group.

Tag groups

The Reprint control file can contain a number of different tag groups. You can use some of the tag groups more than once in a single Reprint control file. You can use others only once in each control file. The table below lists the tag groups and briefly describes their Reprint functions. The table also tells whether you can use each tag group more than once in a control file, and where you can find more information about each tag group and its tags.

Tag group	Description	Usage	In groups	Page
Content	Delimits information in the control file for Reprint to use instead of a separate rule file.	Once	Rule	5-3
Index	Identifies a file called the Reprint Index from which to extract variable values for substitution in an input.	Once	Alone	5-4
Output	Identifies one or more files for Reprint to create, and the parameters with which to create them.	Multi	Alone	5-7
Reprint	Identifies one or more input files to process, and the parameters with which to process them.	Once	Alone	5-10

Tag group	Description	Usage	In groups	Page
Rule	Identifies a file used to control how documents and/or pages are placed in output.	Once	Alone	5-17
Sidefile	Creates a flat file for use by Reprint or other applications. The data in the file retains the order of the output whose tag group defines it.	Multi	Output	5-19

You can place the tag groups in the control file in any order, but each group must begin and end with the proper tags (for example, <REPRINT> and </REPRINT> begin and end a Reprint tag group).

Further, while it is easier to read a control file in which each tag occupies its own line, such as:

```
<REPRINT>
  <NAME>MONTHLY
  <FILE>DD:SYSIN
  .
  .
  .
 </REPRINT>
```

with the exception of the Content tag group you may also enter the data in a continuous stream, such as:

```
<REPRINT><NAME>MONTHLY<FILE>DD:SYSIN</REPRINT>
```

Comments

You can use comments in your control files to include information about the control file or the Reprint application the control file defines. All comments must begin with the character sequence <! and must end with the character >. For example, the example below shows a portion of a control file that contains comments in the header and in the body of the file.

> **Note:** Control file command and control characters are shown in this guide using a U.S. character set (code page). Your actual characters may differ if you do not use a U.S. code page.

```
<!*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*->
<!--  System: Reprint           ->
<!-- Version: 5.2.6                                      ->
<!-- Purpose: Example control data set for reprinting  ->
<!-- pages 47-91.                                        ->
<!-- (c) Copyright 1995-2013,Precisely ->
<!*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*->
<reprint>
```

```
    <name> Letters
    <file> DD:INPUT1
</reprint>
```

Reprint ignores any information between the comment begin and the comment end sequences. You can place comments anywhere in a control file, but you can avoid confusion by devising a standard way of commenting control files. You should begin each control file with comments explaining its purpose.

Comments can be included in any of the following common programming styles:

- /* comment text */ - Comments can span multiple lines up to the closing */ This is helpful if you need to temporarily comment out a block of lines for testing.
- <! comment text > - Comments can span multiple lines up to the first closing >
- // comment text - Comments the rest of the current line after the two slashes.

All three comment styles work in both the rule file and the control file. To comment out blocks of code, use the /**/ style, since the tag syntax <tag> may lead to incorrect closing of the <!> block of comments.

5 - Reprint language reference

In this section

Content Tag group

The Content tag group specifies information that Reprint uses as conditions for rule-based processing. Rather than specifying a <file> tag in the Rule tag group, you can include the data directly in the control file by placing such information between the <CONTENT> and </CONTENT> tags. The Content tag group is optional and can only be used within the Rule tag group.

<CONTENT> and </CONTENT>

Explanation

Identifies the beginning and end of the Content tag group within the Rule tag group. The Content group must begin with the <CONTENT> tag and end with the </CONTENT> tag.

Syntax

<CONTENT> *conditions* </CONTENT>

where conditions is the information Enrichment would normally use the Rule group <file> tag to locate.

Index Tag group

The Index tag group identifies a file (called the Reprint Index) from which to extract information during Reprint processing, and defines where in the Reprint Index to find variable values. The Reprint Index is generally a flat file created by Enrichment or another application. During processing, Reprint substitutes the extracted information for variable names in the rule file. The Reprint control file must contain one Index tag group.

<INDEX> and </INDEX>

Explanation

Identifies the beginning and end of the Index tag group in the control file. The Index group must begin with the <INDEX> tag and end with the </INDEX> tag.

Syntax

```
<INDEX>
<file>
<var>
</INDEX>
```

<FILE>

Explanation

Identifies the location and name of a file associated with the Output group. Reprint places processing results into an output file according to the <filemax> tag. You can use up to 55 characters after the <FILE> tag to identify the file. Each Output tag group can include as many <FILE> tags as necessary. <FILE> is a required tag. Reprint processing cannot proceed unless you set at least one <FILE> tag.

Reprint places extracted documents into the data set defined by the first <FILE> tag in the Output tag group until it reaches the limit specified by the <filemax> tag. Then Reprint places documents into the data set defined by the second <FILE> tag in the Output tag group, and so on.

If the <filemax> tag size value exceeds the number of <FILE> tags defined for an output, Reprint places all extra documents in the last file defined for the output.

Syntax

Any of the following:

<FILE> dd:ddname	where *ddname* is the data definition of the output file.
<FILE> dd:ddname(member)	where *ddname* is the data definition of the output PDS and member is the member name.
<FILE> 'qual.qual.qual'	where *qual* is each qualifier under which the output sequential file is stored. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.
<FILE> 'qual.qual.qual(member)'	where *qual* is each qualifier under which the output PDS is stored and member is the member name. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.

<VAR>

Explanation

Identifies from where in the Reprint Index to extract variable values. You can specify more than one <VAR> tag and value in the Index tag group.

All variable values extracted from the Reprint Index must be defined by <VAR> tags in the Index group. However, not all columns in the Reprint Index must be mapped using <VAR> tags.

You must define either the *%%DOCUMENT_NO* or the *%%OUT_RECORD* system variable with an Index group <VAR> tag. You must also use a <VAR> tag to define the *%%TOTAL_RECORDS* system variable and any other variables necessary for rule file processing.

If you did not use Enrichment to create the Reprint Index, you must still define *%%OUT_RECORD* and *%%TOTAL_RECORDS* in <VAR> tags. However, you must also have created appropriate values for those tags.

Similarly, if the rule file contains <DOCPAGES> statements or PAGERANGE definitions, you must define *%%OUT_TOT_L_PAGE* and *%%OUT_TOT_L_PAGE_BEGIN* in <VAR> tags. Again, you must have created appropriate values for those tags.

If you specify more than one Reprint group <file> tag, you must define the *%%OUT_FILE* system variable with a <VAR> tag, or use the <idvar> tag in the Reprint group.

> **Note:** Reprint allows you to extract only three system variables from the Reprint Index: *%%OUT_RECORD*, *%%TOTAL_RECORDS*, and *%%OUT_FILE*. If other Enrichment system variables are extracted from the Reprint Index, you must rename them in the <VAR> tag and when using the Reprint rule file.

Syntax

<VAR> *%%varname record column length strip*.

Parameter	Description	Default
%%varname	Is up to 50 characters that identify the variable name. The variable name must begin with %%.	None
record	Is the record number within a group of records for each document in the Reprint Index on which the value of *%%varname* is found. The record value is always 1 if the Reprint Index was created from a previous Enrichment run, since a Enrichment side file only contains one record per document.	None

Parameter	Description	Default
column	Is the column number in the Reprint Index where the value of *%%varname* begins.	None
length	Is the length, in characters, of the value *%%varname*. The maximum width of the Reprint Index is 6,144 bytes.	None
strip	Indicates how Reprint strips blanks from the *%%varname* value, as follows: **R** Strip blanks to the right of the value **L** Strip blanks to the left of the value **B** Strip blanks on both sides of the value	None

Output Tag group

The Output tag group defines one or more files in which Reprint is to store documents and/or pages extracted for reprint. Each Reprint control file must contain at least one Output tag group, but you can specify as many as are necessary.

<OUTPUT> and </OUTPUT>

Explanation

Identifies the beginning and end of an Output tag group in the control file. Each Output group must begin with the <OUTPUT> tag and end with the </OUTPUT> tag.

> **Note:** Your JCL must properly define all output files when running Reprint. You must define all output files with the same properties as the source input files.

Syntax

```
<OUTPUT>
  <name>
  <file>
  <filemax>
  <sidefile>
   .
   .
   .
```

```
  </sidefile>
</OUTPUT>
```

<NAME>

Explanation

Identifies a unique name for the Output group. <NAME> is a required tag. Reprint processing cannot proceed unless you set the <NAME> tag.

Syntax

<NAME> name

where name is a unique name of up to 50 characters. The name value cannot contain spaces.

<FILE>

Explanation

Identifies the location and name of a file associated with the Output group. Reprint places processing results into an output file according to the <filemax> tag. You can use up to 55 characters after the <FILE> tag to identify the file. Each Output tag group can include as many <FILE> tags as necessary. <FILE> is a required tag. Reprint processing cannot proceed unless you set at least one <FILE> tag.

Reprint places extracted documents into the data set defined by the first <FILE> tag in the Output tag group until it reaches the limit specified by the <filemax> tag. Then Reprint places documents into the data set defined by the second <FILE> tag in the Output tag group, and so on.

If the <filemax> tag size value exceeds the number of <FILE> tags defined for an output, Reprint places all extra documents in the last file defined for the output.

Syntax

Any of the following:

<FILE> dd:ddname	where *ddname* is the data definition of the output file.
<FILE> dd:ddname(member)	where *ddname* is the data definition of the output PDS and member is the member name.
<FILE> 'qual.qual.qual'	where *qual* is each qualifier under which the output sequential file is stored. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.

<FILE> 'qual.qual.qual(member)'	where *qual* is each qualifier under which the output PDS is stored and member is the member name. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.

<FILEMAX>

Explanation

Specifies the maximum amount of data that Reprint should place in one output file. When the indicated value is reached during processing, Reprint completes the current document, then begins placing data in the next file identified by an Output group <file> tag. If data is left over after all named files are filled, Reprint places that data in the last file named. Output files always contain complete documents, regardless of the type setting.

Syntax

<FILEMAX> size type

Parameter	Description	Default
size	An integer from 1 to 2147483648 that specifies the maximum number of bytes, lines, pages, or documents that Reprint should place in each output file.	None
type	One of the following that specifies the unit in which the size parameter is measured: **B[YTES]** Adds data to output files up to the number of bytes specified by the size value. When the limit is reached, Reprint places the remainder of the current document in the output file before placing data in the next output file. **L[INES]]** Adds data to output files up to the number of lines specified by the size value. When the limit is reached, Reprint places the remainder of the current document in the output file before placing data in the next output file. **P[AGES]]** Adds data to output files up to the number of pages specified by the size value. When the limit is reached, Reprint places the remainder of the current document in the output file before placing data in the next output file. **D[OCUMENTS]]** Adds data to output files up to the number of documents specified by the size value.	PAGES

Reprint Tag group

The Reprint tag group defines one or more print streams to use as input to Reprint, and the parameters to use to process them. The print streams identified by the Reprint tag group are generally outputs from a previous Enrichment processing run. You can specify multiple Reprint tag groups in a single Reprint control file.

<REPRINT> and </REPRINT>

Explanation

Identifies the beginning and end of a Reprint tag group within a Reprint control file. Each Reprint group must begin with the <REPRINT> tag and end with the </REPRINT> tag.

Syntax

<REPRINT> *<name>* *<file>* *<idvar>* *<sequential>* *<type>* *<header>* *<multup>* *<duplex>* *<outtop>* </REPRINT>

<NAME>

Explanation

Identifies the unique name for the Reprint tag group. <NAME> is a required tag. Reprint processing cannot proceed unless <NAME> is explicitly set.

Syntax

<NAME> *name*

where *name* is a unique name of up to 50 characters. The *name* value cannot contain spaces.

<FILE>

Explanation

Identifies the location and name of a source print stream to process. Reprint extracts documents and/or pages from the print stream as defined in the remainder of the control file, and places the result in the appropriate output file. You can use up to 55 characters after the <FILE> tag to identify

the input print stream. If you use the Reprint group <idvar> tag or you use the *%%OUT_FILE* system variable in an Index group <var> tag, you can specify multiple <FILE> tags. <FILE> is a required tag. Reprint processing cannot proceed unless you explicitly set at least one <FILE> tag.

If you use a single <FILE> tag in the Reprint group, the Reprint Index must contain the *%%OUT_RECORD* and *%%TOTAL_RECORDS* system variables.

If you use multiple <FILE> tags in the Reprint group, you must also use the <idvar> tag. The Reprint Index must also contain the *%%OUT_RECORD*, *%%OUT_FILE*, and *%%TOTAL_RECORDS* system variables.

If you use multiple <FILE> tags in the Reprint group but you did not use the Output group <filemax> tag to create the inputs in Enrichment, you can define a variable similar to the *%%OUT_FILE* system variable. Thus, for multiple inputs to Reprint your Reprint group might contain tagging that resembles that shown below, where *IDn* is the variable value as it appears in the Reprint Index, and the <idvar> tag defines the name of the variable as it appears in the Index group <var> tag.

```
<FILE> Name1 ID1 <FILE> Name2 ID2 <FILE> Name3 ID3 <IDVAR> %%varname
```

Syntax

<FILE> *filename [ID]*

Parameter	Description	
filename	Is one of the following:	
	dd:ddname	where *ddname* is the data definition of the file
	dd:ddname(member)	where *ddname* is the data definition of the PDS and member is the member name
	'qual.qual.qual'	where *qual* is each qualifier under which the sequential file is stored. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked and is inserted as the first qualifier
	'qual.qual.qual(member)'	where *qual* is each qualifier under which the PDS is stored and member is the member name. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier
[ID]	Defines the identifier used for the specified print stream. Must correspond to a user-defined variable name listed in an <idvar> tag. ID is required only if you use a user-defined variable instead of *%%OUT_FILE*.	

<IDVAR>

Explanation

Allows you to identify multiple Reprint inputs by identifying the file ID in the Reprint Index.

Syntax

<IDVAR> *%%varname*

where *%%varname* is a variable defined in an Index group <var> tag that specifies the input ID in the Reprint Index.

The *%%varname* value must be defined in an Index group <var> tag.

Sequential

Explanation

Specifies whether the Reprint group <file> tags are in the same order as the records in the Reprint Index. If so, Reprint opens files only as they are needed and closes them as they are processed so that only one file is open at a given time. When the Reprint group <file> tags are not in the same order as records in the Reprint Index, Reprint opens each file as it is needed and closes all opened files at the conclusion of the Reprint run.

Syntax

<SEQUENTIAL> *value*

Parameter	Description		Default
value	One of the following:		YES
	Y[ES]	The input print streams listed in the Reprint tag group are in the same order as the records that define them in the Reprint Index	
	N[O]	The input print streams listed in the Reprint tag group are not in the same order as the records that define them in the Reprint Index. Setting <SEQUENTIAL> to NO increases Reprint processing speed slightly since all opened files remain open until processing has finished.	

<TYPE>

Explanation

Identifies the type of print stream used as input.

Syntax

<TYPE> type

Parameter	Description		Default
type	**A**	The input is an AFPDS print stream	A
	L	The input is a line-printer, AFP line-data, Xerox DJDE, or Xerox Metacode print stream	
	P	The input is a PCL print stream.	

<HEADER>

Explanation

Identifies the lines of header information to skip during Reprint processing, and whether to add the header information to the output stream. Header information is the data that precedes the first document in the input print stream.

If you set *keepYN* to *YES* for an AFP input and the header contains the BDT, Enrichment does not add the Enrichment BDT. If *keepYN* is set to *NO* or the header does not include the BDT, Enrichment replaces the BDT with the Enrichment BDT.

Syntax

<HEADER> *records [keepYN]*

or

<HEADER> *string startcolumn [keepYN]*

Parameter	Description	Default
records	The number of records at the top of the input print stream to ignore.	0

Parameter	Description	Default
string	A string which can be specified in any of the normal forms to indicate that the end of the header has been reached.	None
startcolumn	The column the specified string must start to mark the end of the header.	None
[keepYN]	One of the following: **Y[ES]** Keep the header information, copying it to the top of each output print stream **N[O]** Discard the header information (that is, do not place it in the output)	YES

<MULTUP>

Explanation

Identifies the number and ordering of multiple-up logical pages on a single physical page in a line-printer input print stream. Each Reprint group can contain one <MULTUP> tag.

The <MULTUP> tag is only necessary when the rule file contains <DOCPAGES> statements and/or PAGERANGE definitions for multiple-up line-printer inputs that are not handled by a page format (that is, non-AFP line-data inputs).

Syntax

<MULTUP> *across down order channel*

Parameter	Description	Default
across	The number of logical pages horizontally on a physical page.	See note.
down	The number of logical pages vertically on a physical page.	See note.
order	One of the following: **R[OW]** If logical page 2 follows logical page 1 horizontally on the physical page **C[OLUMN]** If logical page 2 follows logical page 1 vertically on the physical page	See note.

Parameter	Description		Default
channel	One of the following:		See note.
	C[C]	If your line data contains carriage controls but not TRCs	
	N[OCC]	If your line data contains neither carriage controls nor TRCs	
	T[RC]	If your line data contains carriage controls and TRCs.	

Note: If you set the <MULTUP> tag across and down parameters but you do not set the order and channel parameters, Reprint assumes an order value of ROW and a channel value of TRC.

<DUPLEX>

Explanation

Defines whether the input print stream is duplex or simplex. The <DUPLEX> tag is only necessary when the rule file contains <DOCPAGES> statements and/or PAGERANGE definitions for multiple-up duplex inputs.

Syntax

<DUPLEX> *duplex flip*

Parameter	Description		Default
duplex	One of the following:		NO
	Y[ES]	If the input prints on both sides of a sheet of paper (duplex)	
	N[O]	If the input prints on only one side of the paper (simplex)	
flip	One of the following:		NO
	Y[ES]	If the multiple-up duplex input has the back logical pages on the front of the physical sheets of paper	
	N[O]	If the logical pages are in the normal consecutive order on the physical sheets of paper	

<OUTTOP>

Explanation

Specifies a number of records in the input print stream after which Reprint adds a record that indicates top of document. This is for multiple-up documents so Enrichment or another program can delete extraneous pages once Reprint processing is complete.

Reprint adds the text specified by the <OUTTOP> tag after the record number specified under the following conditions:

- A new physical page has begun; and
- The previous document ended on the previous physical page.

If the data is flipped multiple-up duplex, you must also set the <duplex> tag flip parameter to YES.

Syntax

<OUTTOP> *records text*

Parameter	Description	Default
records	An integer greater than 0 that indicates the number of records after which Reprint is to add a record that indicates top of document.	None
text	Is the string to insert at the records position. The string can be alphanumeric or hexadecimal characters. If an alphanumeric string contains spaces, it must be surrounded by quotation marks. Hexadecimal strings cannot contain spaces, and must be specified using hexadecimal notation (for example, x'C1C3E3').	None

Rule Tag group

The Rule tag group defines a file that contains conditions used by Reprint to control reprint. You must specify one Rule tag group in each Reprint control file.

<RULE> and </RULE>

Explanation

Identifies the beginning and end of the Rule tag group in the control file. The Rule tag group must begin with the <RULE> tag and end with the </RULE> tag.

Syntax

```
<RULE>
  <name>
  <file> or <content>
                .
                .
                .
            </content>
</RULE>
```

<NAME>

Explanation

Specifies a unique name for the Rule tag group. <NAME> is a required tag. Reprint processing cannot proceed unless you explicitly set the <NAME> tag.

> **Note:** This chapter discusses control file tags used to identify a rule file to Reprint. For information on the rule file, see **The Rule file** on page 15.

Syntax

<NAME> *name*

where *name* is a unique name of up to 50 characters. The *name* value cannot contain spaces.

<FILE>

Explanation

Identifies the location and name of the rule file associated with the Rule tag group if the Content tag group is not used. Reprint processes the file you specify in this tag and places the result in the appropriate output file. You can use up to 55 characters after the <FILE> tag to identify the rule file. The Rule group can contain one <FILE> tag. The Rule group requires the <FILE> tag if the Content tag group is not used. You cannot use a <FILE> tag in a Rule group that contains the Content tag group.

Syntax

Any one of the following:

<FILE> DD:ddname	where *ddname* is the data definition of the rule file.
<FILE> DD:ddname(member)	where *ddname* is the data definition of the rule file PDS and member is the member name.
<FILE> 'qual.qual.qual'	where *qual* is each qualifier under which the sequential rule file is stored. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.
<FILE> 'qual.qual.qual(member)'	where *qual* is each qualifier under which the rule file PDS is stored and member is the member name. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.

Sidefile Tag group

The Sidefile tag group is used within the Output tag group to identify a flat file to which Reprint writes extracted data for reporting. You can specify multiple Sidefile tag groups in each Output tag group in the control file to produce multiple side files.

<SIDEFILE> and </SIDEFILE>

Explanation

Identifies the beginning and end of the Sidefile tag group within an Output tag group. In addition, the <SIDEFILE> tag identifies the location and name of the flat file associated with the Sidefile tag group. You can use up to 55 characters after the <SIDEFILE> tag to identify the file.

The Sidefile tag group must begin with the <SIDEFILE> tag and end with the </SIDEFILE> tag.

Syntax

```
<SIDEFILE>
  <sidepart>
  <file>
</SIDEFILE>
```

<SIDEPART>

Explanation

Identifies a variable that Reprint is to write to the side file. You should specify one <SIDEPART> tag for each variable to write. Reprint writes one record to the side file for each document in the output.

Reprint assembles records in the order in which the corresponding <SIDEPART> tags appear in the Sidefile tag group, from left to right.

Syntax

<SIDEPART> *%%varname length [justify pad]*

Parameter	Description	Default
%%varname	Up to 50 characters (including %%) that identify a variable name whose information Reprint is to add to the side file. The *%%varname* value must be one of the following: • A valid system variable. • Defined in the Reprint Index. You must also define the variable in an Index tag group <var> tag.	None
length	Specifies the length in characters of the variable value identified by *%%varname*. If the actual length of the variable value is less than the length you specify, Reprint uses the justify and pad values to correctly position and lengthen the variable value. If the actual length of the variable value is greater than the length you specify, Reprint uses the justify value to truncate the variable value.	None
[justify]	Specifies how Reprint positions a variable value shorter or longer than length, as follows: **L** Left-justifies the variable value. If the variable value is shorter than length, places pad characters after the variable value. If the variable value is longer than length, truncates the variable value from the right. **R** Right-justifies the variable value. If the variable value is shorter than length, places pad characters before the variable value. If the variable value is longer than length, truncates the variable value from the left. **N** Does not pad the variable value. If the length of the variable value exceeds length, truncates the variable value from the right.	See hints.
[pad]	Specifies a single character for Reprint to use to pad a variable value to length. The pad character can be a letter, number, special character, hexadecimal character, or blank. A space surrounded by single quotation marks (' ') indicates a blank pad character. Do not specify a pad value if justify is set to N.	See hints.

Note: Reprint and other applications require that each field in the side file begin in the same column for each document. If your application has such a requirement, or the application requires the side file to be in Fixed Block format, you should not set justify to N.

In most cases, *%%varname* values are variables defined in the Reprint Index. They may also be variable values defined only in the rule file.

You can also use the system variables listed in the following table in the <SIDEPART> tag.

Note: Values for the *%%OUT_RECORD* and *%%TOTAL_RECORDS* system variables must be in the Reprint Index and defined in Index group <var> tags for successful Reprint processing. If you use PAGERANGE definition or <docpages> statement in the rule file or if you are using

multiple inputs from a single Enrichment Output tag group, you may also define *%%OUT_FILE*, *%%OUT_TOT_L_PAGE*, and *%%OUT_TOT_L_PAGE_BEGIN*.

System variable	Description
%%DOCUMENT_NO	The overall document number, regardless of which output contains the document (or case).
%%INPUT_PAGES	The total number of logical pages in the document or case.
%%L_PAGE_NO	The page number of the current logical page.
%%L_TOTAL_PAGES	The total number of logical pages in the document or case.
%%OUT_BYTE	The number of bytes before the current document in the output.
%%OUT_FILE	The number of the output file that contains the current document. Required if the Reprint tag group contains multiple <file> tags when the Reprint group <idvar> tag is not used.
%%OUT_L_PAGE	The current logical page number in the output.
%%OUT_L_PAGE_BEGIN	The number of logical pages before the current document in the output.
%%OUT_PAGE	The logical front page number in the output.
%%OUT_PAGE_BEGIN	The number of logical front pages before the current document in the output.
%%OUT_RECORD	The number of records before the current document in the output. Required for all Reprint applications.
%%OUT_TOT_BYTE	The number of bytes before the current document in the output.
%%OUT_TOT_L_PAGE	The number of logical pages in the output. Required if <docpages> statements and/or PAGERANGE definitions are used in the rule file.
%%OUT_TOT_L_PAGE_BEGIN	The number of logical pages before the current document in the output. Required if <docpages> statements and/or PAGERANGE definitions are used in the rule file.
%%OUT_TOT_PAGE	The number of logical front pages in the output.
%%OUT_TOT_PAGE_BEGIN	The number of logical front pages before the current document in the output.

System variable	Description
%%OUT_TOT_RECORD	The number of records before the current document in the output.
%%PAGE_NO	The logical front page number of the current physical page in the document.
%%SEQUENCE_BEGIN	0 if beginning a new sequence, or 1 if continuing a sequence. You cannot use *%%SEQUENCE_BEGIN* to replace the value of a variable.
%%SEQUENCE_NO	The logical front page number of the current physical page in the document.
%%TOTAL_PAGES	The total number of logical front pages in the document or case.
%%TOTAL_RECORDS	The number of records per document. Required for all Reprint applications.

Hints

1. For numeric system variables:
   - Default *justify* value is R
   - *justify* value of L or R works like R
   - Pads with 0 regardless of the specified pad character
   - *justify* value of N is not padded
2. For non-numeric system variables and user-defined variables:
   - Default *justify* value is L
   - Default pad character is a blank
   - *justify* value of N is not padded and implies Left justification. Enrichment gives up to the full length of the variable if the specified length is longer than the variable's actual length

<FILE>

Explanation

Identifies a file associated with the Sidefile tag group.

Syntax

Any of the following:

<FILE> DD:ddname	where *ddname* is the data definition of the rule file.
<FILE> DD:ddname(member)	where *ddname* is the data definition of the rule file PDS and member is the member name.
<FILE> 'qual.qual.qual'	where *qual* is each qualifier under which the sequential rule file is stored. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.
<FILE> 'qual.qual.qual(member)'	where *qual* is each qualifier under which the rule file PDS is stored and member is the member name. If you do not enclose the tag value in single quotes, the user ID from which Reprint is invoked is inserted as the first qualifier.

Notices

2 Blue Hill Plaza, #1563
Pearl River, NY 10965
USA

www.precisely.com