Advantage™ VISION:Results™ for z/OS

Reference Summary

r6

Contents

Chapter 1: Introduction

Program Layout . . . 11
Automatic Cycle Without Sort . . . 12
Automatic Cycle . . . 13
Syntax Conventions . . . 14
General Rules for Command Operands . . . 15
Summary of Commands . . . 17
Contacting CA Technical Support . . . 26

Chapter 2: Commands

$COBOL . . . 27
Examples . . . 27
$DEFAULT . . . 27
Examples . . . 27
$GDEF . . . 28
Examples . . . 28
$GSET . . . 28
Examples . . . 28
$IF . . . 28
Examples . . . 28
$IFVALUE . . . 29
Examples . . . 29
$PRINT . . . 29
Examples . . . 29
$SET . . . 29
Examples . . . 29
%macroname . . . 30
Examples . . . 30
%sublibrary . . . 30
Examples . . . 30
ACCEPT . . . 30
ALLOCATE . . . 30
Examples . . . 31

ARRAY 31
Notes 31
Examples 31
BINSEARCH 31
Examples 31
CALL 32
Examples 32
CASE 32
CLOSE 32
Notes 32
COMBINE BITS 33
Examples 33
CONTROL 33
Examples 33
COPY 33
Notes 33
Examples 34
COPYDB2 34
Examples 34
dataname definitions 34
Examples 34
dataname assignment 35
Examples 35
DELETE 35
Examples 35
DOUNTIL 35
DOWHILE 36
DYLaaaaaaa 36
Examples 36
EDIT 36
Examples 36
EXEC SQL 37
EXITCASE 37
EXITDO 37
FILE (BDAM Files) 37
Notes 38
Examples 38
FILE (EXIT Files) 38
Notes 38
Examples 38
FILE (ISAM Files) 39
Notes 39
Examples 39
FILE (PDS Files) 40
Examples 40
FILE (Sequential Files) 40
Notes 41
Examples 41

FILE (SSL Files) 41

Examples 41

FILE (VSAM Files) 41

Notes 42

Examples 42

FILLER 42

Examples 43

FIN 43

GOTO 43

Examples 43

GETCOM 43

HEX 43

Examples 44

HEXPRINT 44

Examples 44

IF 44

Examples 44

Examples 45

Examples 45

Examples 45

Examples 46

Examples 46

Examples 47

IQFILE 47

IQREAD 47

ITERATE 47

LCPRINT 48

Examples 48

LETTER 48

Notes 48

Examples 48

LINEAR 48

Notes 49

Examples 49

LIST 49

Notes 50

Examples 50

LTD 50

LTH 50

Notes 50

Examples 52

MATCH 53

Examples 53

MERGE 53

Examples 53

MOVE . . . 54
  Examples . . . 54
  Examples . . . 54
  Examples . . . 54
  Examples . . . 54
  Examples . . . 55
  Examples . . . 55
MQBACK . . . 55
MQBEGIN . . . 55
MQCLOSE . . . 55
MQCMIT . . . 56
MQCONN . . . 56
MQCONNX . . . 56
MQDISC . . . 56
MQGET . . . 56
MQINQ . . . 56
MQOPEN . . . 57
MQPUT . . . 57
MQPUT1 . . . 57
MQSET . . . 57
NEWPAGE . . . 57
NEXT . . . 57
ON . . . 58
  Examples . . . 58
OPTION . . . 58
  Examples . . . 59
PCFILE . . . 59
PCWRITE . . . 59
PERFORM . . . 60
  Notes . . . 60
  Examples . . . 60
PICNSAVE . . . 60
  Notes . . . 60
PRINT . . . 60
  Examples . . . 61
PRINTGEN . . . 61
PRINTNOGEN . . . 61
PUTCOM . . . 61
  Examples . . . 61
QUIT . . . 61
QUITALL . . . 62
READ . . . 62
  Examples . . . 62
READBOOK . . . 62
  Examples . . . 62
READDIR . . . 62
  Examples . . . 62
READMEM . . . 63
  Examples . . . 63

REDEFINE . . . 63
Examples . . . 63
REGRESSION . . . 63
Examples . . . 63
REJECT . . . 63
RELSORT . . . 64
REPORT . . . 64
Examples . . . 64
REPORTFILE . . . 65
Examples . . . 65
RETRIEVE . . . 65
Notes . . . 65
Examples . . . 65
RETSORT . . . 65
SAMPLE . . . 66
Notes . . . 66
Examples . . . 66
Notes . . . 66
Examples . . . 66
SCATTER . . . 67
Notes . . . 67
Examples . . . 67
SEARCH . . . 67
Notes . . . 68
Examples . . . 68
SORT . . . 68
Notes . . . 68
Examples . . . 68
STATEOFF . . . 69
STATEON . . . 69
STOP . . . 69
STOPALL . . . 69
STORE . . . 69
Notes . . . 70
Examples . . . 70
tagname . . . 70
Examples . . . 70
TABLE . . . 70
Notes . . . 70
Examples . . . 71
TRANSLATE . . . 71
Examples . . . 71
TREND . . . 71
Notes . . . 72
Examples . . . 72
TSORT . . . 72
Examples . . . 72

Tn 73
Notes 73
Examples 73
USE 73
Notes 73
Examples 74
VCLOSE 74
Examples 74
VDELETE 74
Examples 74
VFILE 74
VOPEN 75
Examples 75
VREAD 75
Examples 75
VREADDIR 76
Examples 76
VRENAME 76
Examples 76
VWRITE 76
Examples 76
WORKAREA 77
Examples 77
WRITE 77
Examples 77
WRITEDIR 77
Examples 77
WRITEMEM 78
Examples 78
XMLFILE 78
Examples 78
XMLWRITE 78
Examples 78

Chapter 3: DYLINSTL Macro

Mandatory Parameters 79
Optional Parameters 80
Considerations for Large Programs 111
Additional Considerations for Extremely Large Programs 113
Calling COBOL Programs. 114
DYLINSTL Macro Example and Options List 114
. 115
Run CUSTMJCL to Change the DYLINSTL Macro Parameters 115
Creating Different Functional Versions of VISION:Results 117

Chapter 4: Reserved Words

Symbols 120

A 120
B 120
C 120
D 120
F 123
G 123
H 123
I 123
J 123
K 124
L 124
M 124
N 124
O 125
P 125
Q 125
R 125
S 125
T 126
U 126
V 126
W 126
X 127
Y 127
Z 127
Numbers 127
Definitions 127

Chapter 5: Programming Tips

Punctuation Limitations 133
Colons and Periods 133
Asterisks and Semicolons 133
Continuation Rules 134
Naming Conventions 134
Data Names 134
File Names 135
Tag Names 135
Number of Characters per Input Line. 135
Literals 136
Character or Alphanumeric Literals. 136
Moving Character Literals 137
Comparing Character Literals 137
Assigning Character Literals to Fields 137
Listing Literals in a Report. 138
Printing a Literal in File Print 138
Numeric Literals. 138
Moving Numeric Literals 139
Comparing Numeric Literals 139

Using Numeric Literals in Arithmetic Expressions . . . 139
Assigning Numeric Literals to Numeric, Packed, or Binary Fields . . . 139
Hexadecimal Literals . . . 139
Moving Hexadecimal Literals . . . 140
Comparing Hexadecimal Literals . . . 140
Assigning Values to Fields Using Hexadecimal Notation . . . 140
Rules for Comparing Fields . . . 140
Character/Character . . . 140
Character/Numeric or Packed . . . 140
Character/Binary . . . 140
Numeric/Numeric . . . 141
Numeric/Binary . . . 141
Numeric/Packed . . . 141
Packed/Packed . . . 141
Packed/Binary . . . 141
Binary/Binary . . . 141
Rules for Moving Fields . . . 141
Character/Character . . . 141
Character/Numeric . . . 142
Character/Packed . . . 142
Character/Binary . . . 142
Numeric/Character . . . 142
Numeric/Numeric . . . 142
Numeric/Packed . . . 142
Numeric/Binary . . . 142
Packed/Character . . . 143
Packed/Numeric . . . 143
Packed/Packed . . . 143
Packed/Binary . . . 143
Binary/Character . . . 143
Binary/Numeric . . . 143
Binary/Packed . . . 144
Binary/Binary . . . 144
Numeric and Packed Key Considerations . . . 144
Edit Codes . . . 145

Chapter 1: Introduction

This guide contains the syntax of all commands for Advantage™ VISION:Results™ for z/OS, as well as information about reserved words, JCL, and other programming considerations.

Note: Advantage VISION:Results will be referred to as VISION:Results throughout this guide.

Program Layout

Suggested steps:

1. OPTION statement(s).
2. REPORT statement.
3. FILE and WORKAREA definitions.
4. Advantage™ VISION:Excel™ free-form statements.
5. CONTROL statement.
6. If sorting, unsorted file processing followed by SORT command.
7. Detail-time processing.
8. Subroutines (make sure detail-time processing logic is kept separate from the subroutines).
9. ON END OF SORTING

   ON END OF INPUT.
10. ON CHANGE IN dataname

    ON FINAL.
11. TITLE information.

WORKAREA definitions can occur after the OPTION statement, but fields can be referenced only after they are defined.

OPTION COLUMNS (changes free-form columns) can be anywhere in a program.

Notes

- An asterisk (*) in position 1 indicates a comment.
- A semicolon (;) begins a comment.
- Data that is not in the free-form columns is considered to be a comment.

Automatic Cycle Without Sort

Note: In this flowchart, when VISION:Results reads all input files and writes all output files, this applies only to those files that do not have READ/WRITE commands.

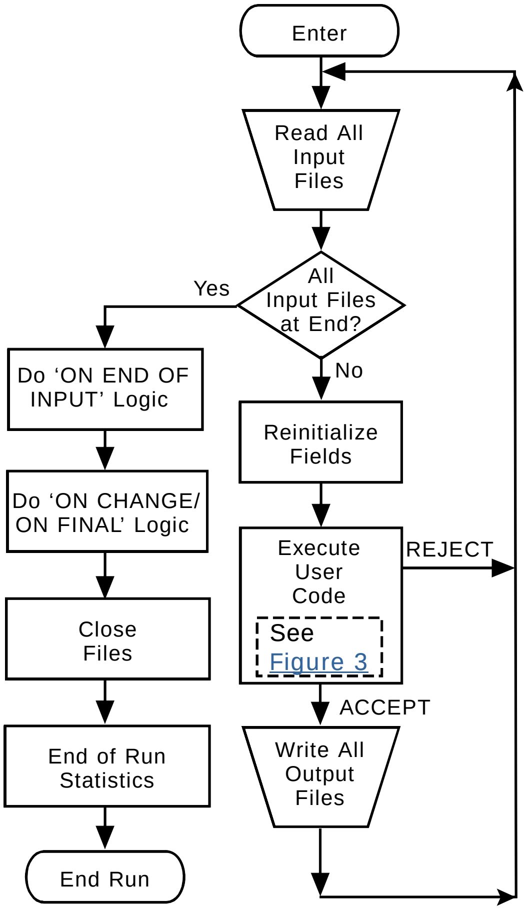

Figure 1 Flowchart A

Automatic Cycle

Note: In this flowchart, when VISION:Results reads all input files and writes all output files, this applies only to those files that do not have READ/WRITE commands.

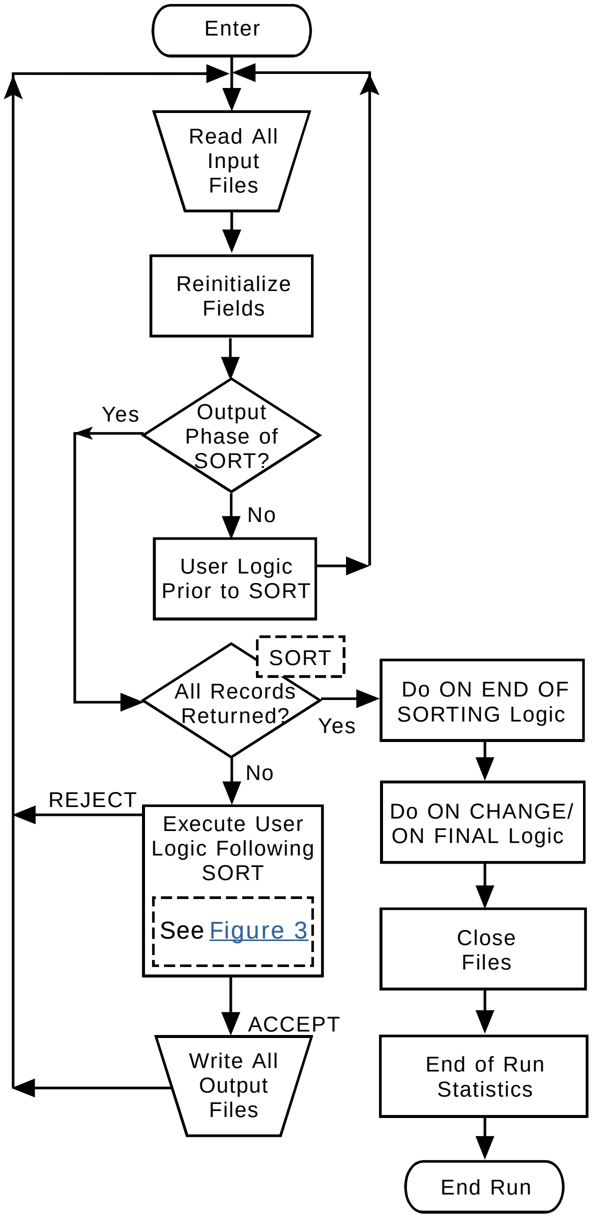

Figure 2 Flowchart B

If there are any detail LIST statements, the following takes place whenever the first LIST statement is executed (before the detail report line is produced). If there are no detail LIST statements, the following takes place as part of the automatic cycle at the same point as where output files are written.

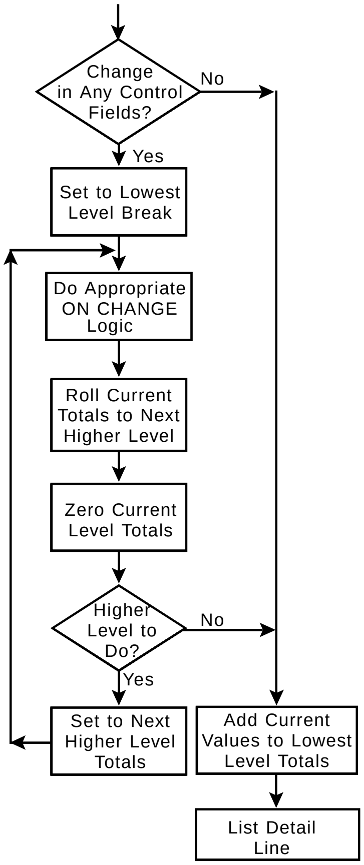
Change in Any Control Fields?
No
Yes
Set to Lowest Level Break
Do Appropriate ON CHANGE Logic
Roll Current Totals to Next Higher Level
Zero Current Level Totals
Higher Level to Do?
No
Yes
Set to Next Higher Level Totals
Add Current Values to Lowest Level Totals
List Detail Line

Figure 3   Flowchart C

Syntax Conventions

- A vertical bar (|) separates alternatives. Only one alternative can be selected.
- An underscore (_) indicates a default value.
- Braces ({ }) are used to group related items that are alternatives. One item must be chosen.
- Brackets ([ ]) are used to indicate optional items. Everything within the brackets is optional and can be omitted.
- Uppercase letters, words, and special characters must be specified exactly as shown.

- Lowercase letters and words require a specific value to be substituted in their place.
- Parentheses must be entered exactly as shown.
- Keywords and operands between a set of ! and !... indicate that repetition of what is between the ! and !... is allowed.
- Bolded text indicates VSE only.

General Rules for Command Operands

Rule	Description
alphanumeric	Must begin with an alphabetic character and can be followed by any of the characters A to Z, 0 to 9, or $ % ! ? # ' [ ] { } – .
alphanumeric literal	When used to give an initial value to a data name (for example, VALUE “literal”), it can be 1-255 characters. When used in a Tn statement, it can be up to the report width (in both cases, however, it must be physically broken up into pieces that can be contained on one free-form line). When used in a MOVE statement, it can be 1-70 characters. Wherever it is used, the alphanumeric literal begins and ends with a single or double quotation mark. If it begins with a single quotation mark and a single quotation mark is to be part of the literal, all single quotation marks that are part of the literal must be repeated (the syntax analyzer strips out the repeated quotation marks). For example, the literal IT'S can be written as “IT'S” or “IT”s'. The same is true of literals beginning with a double quotation mark that contain a double quotation mark.
blocksize	1 to 5 numeric characters defining the size of a block. The maximum is 32,767. Variable-length maximum is 32,760.
column heading	1 to 30 characters to be printed for the data name in the column heading.
dataname	2 to 50 alphanumeric characters. The name is used in a record or work area field definition for a VISION:Results reserved word (begins with DYL). The installation default is 50 characters.
dataname2	2 to 10 alphanumeric characters. The name is used in a FILE, TABLE, or ARRAY statement following a keyword. This is a self-defining data name that must not be defined in record or work area field definitions.

Rule	Description
dataname3	2 to 10 alphanumeric characters. This is the name of a record or work area field definition data name that is specified in the parameter list of the keyword PARM of a MODIFY or EXIT FILE definition.
decimals	One numeric character specifying the number of digits to the right of the assumed decimal point in the field.
filename	One to eight (one to seven for VSE) alphanumeric characters specifying the name of a file.
hex literal	5 to 21 literals that specify a hexadecimal value. It is in the form X'nn', where nn is pairs of hex digits (a hex digit is 0 to 9 and A to F).
idname	See filename.
imperative	One or more commands that accomplish conditional selection. An imperative cannot contain another IF, except in STRUCTURED programming.
index	Either (INW), (INX), (INY), (INZ), or USERINXnn.
length	One to five numeric characters specifying the length of a data name.
literal	See numeric literal, alphanumeric literal, and hex literal.
membername	The name of a member in the PDS specified by the filename DD statement.
modulename	One to eight alphanumeric characters specifying the name of an external load module to be called.
n	A number whose attributes are determined by the command or keyword that uses it.
nK	Amount of memory required in 1 KB increments, where 1 KB is 1024 bytes. The default is 4 KB.
numeric	Contains only the characters 0 to 9.
numeric literal	1 to 20 characters specifying a value. It can contain the digits 0 to 9, one decimal point, commas to denote thousands, and a leading minus sign.
password	One to eight alphanumeric characters specifying a password (must be in single quotation marks).
printdecimals	One numeric character specifying the number of digits to be printed to the right of the decimal point.

Rule	Description
printdigits	One to two numeric characters specifying the number of digits to be printed to the left of the decimal point. Maximum is 26.
printsize	One to two numeric characters specifying the print size of a numeric field using an edit code in the text of the letter writing facility.
recordsize	One to five numeric characters specifying the size of a record. The maximum is 32,767. Variable-length record maximum is 32,752.
start	One to five numeric characters specifying the starting location of a data name within a record or work area.
tagname	1 to 10 characters (starting with a letter or number and ending with a colon) or 1 to 10 digits specifying the name to be associated with the following commands: GOTO, PERFORM.
title	One to n characters specifying the title to be used. If the title is wider than 70 characters, the title specification is repeated.

Summary of Commands

Command	Description
$COBOL	Indicates that instream COBOL statements immediately follow.
$ECOBOL	Terminates instream COBOL statement processing (valid only with $COBOL).
$DEFAULT	Indicates that the default values of symbolic variables follow.
$DEND	Terminates the definition of the symbolic variables (valid only with $DEFAULT).
$ELSE	Allows the inclusion of an alternate path in a $IF or $IFVALUE clause.
$ENDGDEF	Terminates the $GDEF process.
$ENDGSET	Terminates the $GSET process.
$GDEF	Indicates that the default values of macro global symbolic variables follow.

Command	Description
$GSET	Indicates that what follows this command changes the value of one or more macro global symbolic variables.
$IF	Controls conditional compilation in a COPY member or macro based upon whether or not the value of a symbolic variable is null.
$IFE	Terminates the $IF and $IFVALUE macro commands.
$IFVALUE	Controls conditional compilation in a macro based upon the relationship between a symbolic variable and a literal value.
$PRINT	Prints the macro symbolic variables and checks their name, length, value, if active, local or global status, and how they were set.
$SET	Indicates that what follows this command changes the value of one or more macro local symbolic variables.
%macroname	Invokes a macro.
%sublibrary	Used to establish an alternative to the D sublibrary qualifier for invoking a macro (VSE only).
ACCEPT	Exits program logic and enters the output phase of the automatic cycle.
ALLOCATE	Invokes Dynamic Allocation, a z/OS service that allows input and output files to be allocated without having to provide the JCL to do so.
ARRAY	Defines the attributes of a two- to four- dimensional array.
BINSEARCH	Performs a binary search on a fixed length entry table.
CALL	Executes an external subroutine.
CASE	Designates the start of a case structure and specifies the subject of all subordinate WHEN keywords.
CLOSE	Closes the specified file (valid only for exit-processed files).
COMBINE BITS	And, or, exclusive, or data.
CONTROL	Designates data names that represent report control breaks.

Command	Description
COPY (COPYE) COPYC COPYL COPYP COPYN	Includes precoded source code from a source library.
COPYDB2	For any given DB2 table, copies the column names from the DB2 catalog as field names into a VISION:Results program.
datanamex = xxxx + yyyy	Adds yyyy to xxxx and places the result in datanamex. Only the contents of datanamex are altered.
datanamex = xxxx - yyyy	Subtracts yyyy from xxxx and places the result in datanamex. Only the contents of datanamex are altered.
datanamex = xxxx * yyyy	Multiplies yyyy by xxxx and places the result in datanamex. Only the contents of datanamex are altered.
datanamex = xxxx / yyyy	Divides yyyy into xxxx and places the result in datanamex. Only the contents of datanamex are altered.
datanamex = xxxx ** yyyy	Raises xxxx to the power yyyy and places the result in datanamex. Only the contents of datanamex are altered.
DELETE	Deletes a table or an array.
DOUNTIL	Designates the start of a program loop and indicates that control should remain in the loop until the specified condition is true. The loop is executed at least once.
DOWHILE	Designates the start of a program loop and indicates that control should remain in the loop while the specified condition is true. The loop might not be executed.
DYACCEPT	Exits program logic and enters the output phase of the VISION:Results automatic cycle (Advantage™ VISION:Results Interface™ to CA-IDMS/DB only).
DYNEXT	Directs the execution of an IF statement for either a true or false condition (VISION:Results Interface to CA-IDMS/DB only).
DYSTORE	Stores an entry in a table or array (VISION:Results Interface to CA-IDMS/DB only).
EDIT	Formats a numeric field according to a specified mask.
ELSE	Allows the inclusion of an alternate path in an IF or CASE statement (valid only with IF and CASE).

Command	Description
ENDCASE	Indicates the end of the code being controlled by a case structure (valid only with CASE).
ENDDO	Indicates the end of the code to be included in the loop being controlled by a DOWHILE or DOUNTIL (valid only with DOWHILE and DOUNTIL).
ENDEXEC	Terminates the execution of SQL statements (valid only with EXEC DBC and EXEC SQL).
ENDIF	Terminates an IF statement (valid only with IF).
ENDLTD	Terminates the heading and body of a letter definition.
ENDONE	Terminates ON ONE statements (valid only with ON ONE).
EXEC DBC/SQL	Executes SQL statements. Valid only with Advantage™ VISION:Results Interface™ to DB2 or DBC (z/OS) or Advantage™ VISION:Results Interface™ to SQL/DS (VSE and CMS).
EXITCASE	Immediately transfers control to the first statement following the current ENDCASE.
EXITDO	Immediately transfers control to the first statement following the current ENDDO.
FILE	Defines files to be processed in a VISION:Results program.
FILLER	Indicates non-referenced fields in field definitions.
FIN	Denotes the end of the VISION:Results program. Must be present if instream data follows the program.
GETCOM	Causes 1-256 bytes to be moved from the JCA to dataname (VSE only).
GOTO	Branches to the logic designated by a tagname.
HEX	Prints the designated record, field, index, or literal in vertical hex format.
HEXPRINT	Prints the designated record, field, index, or literal in vertical hex and graphics format.
IF	Designates the start of a compare statement. Can test for equal to (EQ), not equal to (NE), less than (LT), less than or equal to (LE), greater than (GT), greater than or equal to (GE), positive, negative, numeric, bits on, and bits mixed conditions.

Command	Description

IF MATCHED,
IF ADVANCED,
IF DUPLICATE,
IF FIRSTDUP,
IF LASTDUP, and
IF REPLICA
	Supports the MATCH facility.

IF SAMPLING	Invokes interval or random selection, based on user-supplied criteria.
ITERATE	Causes the initiation of the next iteration of a DOWHILE or DOUNTIL loop.
IQFILE IQREAD	Allows users to embed Advantage™ VISION:Inquiry® syntax in their VISION:Results code to access databases or files through VISION:Inquiry.
LCPRINT	Prints the designated record, field, index, or literal in graphics format. Does not translate lowercase characters to blanks.
LETTER	Invokes automatic text edited letter writing facility.
LINEAR	Defines requirements for linear regression analysis and/or scatter diagrams. Invoked in procedure logic by the REGRESSION nn command, where nn is unique for each regression statement.
LIST	Reports on designated field, index, or literal.
LTD	Defines the body of a letter format in a program. It cannot be specified unless LTH is previously specified.
LTH	Defines the heading of a letter format in a program. Invoked in procedure logic by the LETTER nn command, where nn is unique for each letter described.
MATCH	Declares a match operation on two or more files.
MERGE	Declares a merge operation on two or more files.
MOVE	Transfers data from one field to another. Converts data and aligns decimals if necessary. Can also move numeric bits, zone bits, sums, and literals, move with offset, and move unsigned, absolute, and normalized data.
MOVE CORRESPONDING	Moves all fields belonging to one qualifier to corresponding data name fields of another qualifier.
MQBACK	WebSphere MQ API that backs out a unit of work.

Command	Description
MQBEGIN	WebSphere MQ API that begins a unit of work.
MQCLOSE	WebSphere MQ API that closes a specific queue.
MQCMIT	WebSphere MQ API that commits a unit of work.
MQCONN	WebSphere MQ API that connects the application to a WebSphere MQ Queue manager.
MQCONNX	WebSphere MQ API that connects with standard or FastPath bindings.
MQDISC	WebSphere MQ API that disconnects the application from a queue manager.
MQGET	WebSphere MQ API that retrieves a message from a queue.
MQINQ	WebSphere MQ API that inquires about the properties of an object.
MQOPEN	WebSphere MQ API that opens a specific queue for message access.
MQPUT	WebSphere MQ API that places a message into a queue.
MQPUT1	WebSphere MQ API that serves as a combination of MQOPEN, MQPUT, and MQCLOSE to handle a single message for a specified queue.
MQSET	WebSphere MQ API that sets properties of an object.
NEWPAGE	Skips to the top of the next page before continuing to print the source listing.
NEXT	Directs the execution of an IF statement for either a true or false condition.
ON CHANGE IN	Precedes logic that is to be entered when the designated data name changes in value.
ON END OF INPUT	Precedes logic that is to be entered when all sequentially read (except skip sequential or sequential IO) input files have been read to end of file.
ON END OF SORTING	Precedes logic that is to be entered when the SORT has returned all sorted records.
ON FINAL	Precedes logic that is to be entered for GRAND TOTAL processing.
ON ONE	Precedes logic that is to be executed only once when encountered.

Command	Description
OPTION	Precedes overrides to VISION:Results system defaults.
OTHERWISE	Allows the inclusion of an alternate path in an IF or CASE statement (valid only with IF and CASE).
PCFILE PCWRITE	Allows data to be written to the Advantage™ VISION:Journey® for Windows VSAM download file (z/OS only).
PERFORM tagnamex TO tagnamey	Enters the logic preceded by tagnamex and then returns to the next instruction after the PERFORM statement when tagnamey is encountered.
PICNSAVE	Selects and saves the designated data to be retrieved with a USE in another VISION:Results request.
PRINT	Prints the designated record, field, index, or literal in graphics format.
PRINTGEN/PRINTNOGEN	Allows or suppresses the printing of all instructions generated for each macro.
PUTCOM	Causes 1-256 bytes to be moved from the dataname to the JCA (VSE only).
QUIT	Terminates existing request processing without executing ON CHANGE IN or ON FINAL logic.
QUITALL	Terminates existing request processing and subsequent request processing without executing ON CHANGE IN or ON FINAL logic.
READ	Reads the designated file.
READBOOK	Reads sublibrary book of a source statement library (VSE only).
READDIR or READIR	Reads partitioned data set directory (z/OS only).
READMEM	Reads member of a partitioned data set library (z/OS only).
REDEFINE	Resets the location counter to the specified data name to respecify data definitions from there on.
REGRESSION	Invokes linear regression, trend line analysis, and/or scatter diagrams.
REJECT	Exits user logic, but does not enter output phase of the automatic cycle.
RELSORT	Invokes the SORT command and releases a record to sort. Control returns to the next sequential statement.

Command	Description
REPORT	Precedes overrides to VISION:Results report defaults, such as line width and depth. Also used to specify other reporting formats, such as HTML output or delimiting specifications.
REPORTFILE	Prints file, data, index, or literal immediately in graphics format. Treats the first character as a carriage control (VSE only).
REPORTnnn	Indicates multiple reports, where nnn is the report number.
RETRIEVE	Retrieves an entry from a table or array.
RETSORT	Invokes the SORT command to return a sorted record from sort. Control returns to the next sequential statement.
SAMPLE	Defines the type of selection of test data (interval or random) that you want. Invoked in procedure logic by an IF SAMPLING nn command, where nn is unique for each sampling statement.
SCATTER	Defines requirements for scatter diagrams for either trend line or linear regression analysis. Invoked in procedure logic by a REGRESSION nn command, where nn is unique for each regression statement.
SEARCH	Performs a sequential search on a table.
SORT	Sorts records or data using the designated keys.
STATEOFF	Discontinues printing source statements after this command.
STATEON	Resumes printing source statements after this command.
STOP	After completing the reports and print run statistics, and closing all files, VISION:Results terminates the run.
STOPALL	Terminates the run for the current request and bypasses any subsequent requests.
STORE	Stores an entry in a table or array.
SUBTOTAL	Designates a data name that represents report control breaks.
TABLE	Defines the attributes of a one-dimensional table.

Command	Description
tagname	Identifies a specific location in the program logic to which the logic will branch using the commands GOTO and PERFORM.
TRANSLATE	Translates the contents of a field from one set of values to another.
TREND	Defines the requirements for trend line analysis and/or scatter diagrams. Invoked in procedure logic by a REGRESSION nn command, where nn is unique for each regression statement.
TSORT	Sorts entries in a fixed length entry table.
Tn	Defines the contents and placement of title or footing statements for a report.
USE idname or filename	Uses the file and field definitions for a file that was defined using the RETAIN or PICNSAVE command.
USE tablename or arrayname	Uses the table or array that was defined using the RETAIN command.
VCLOSE	Closes a virtual diskette file (Advantage™ VISION:Vlink™ only).
VDELETE	Deletes a virtual diskette file (VISION:Vlink only).
VFILE	Defines a virtual diskette library and a virtual diskette in that library (VISION:Vlink only).
VOPEN	Opens a virtual diskette file (VISION:Vlink only).
VREAD	Reads from a virtual diskette file (VISION:Vlink only).
VREADIR or VREADDIR	Reads the directory of a virtual diskette (VISION:Vlink only).
VRENAME	Renames a virtual diskette file (VISION:Vlink only).
VWRITE	Writes to a virtual diskette file (VISION:Vlink only).
WHEN	Indicates under what conditions the code following WHEN is to be executed. Valid only when subordinate to CASE.
WORKAREA	Temporary fields that can be created to be used in the program.
WRITE	Writes the designated file.
WRITEDIR	Writes to the designated partitioned data set directory record (z/OS only).

Command	Description
WRITEMEM	Writes to the designated partitioned data set member record (z/OS only).
XMLFILE XMLWRITE	Allows the creation of XML data and schema documents.

Contacting CA Technical Support

For online technical assistance and a complete list of locations, primary service hours, and telephone numbers, contact Technical Support at http://ca.com/support.

Chapter 2: Commands

$COBOL

```
$COBOL
        COBOL field definition statements
        $ECOBOL                        Must begin in column 8 or later.
```

Figure 4 $COBOL Command Syntax

Examples

```
$COBOL
    01  ARFIELDS.
          02  AR-ACCTCODE  PIC XX.
          02  AR-ACCOUNT   PIC 9(5).
          02  AR-AMOUNT    PIC S9(5)V99 USAGE COMP-3.
$ECOBOL

$COBOL
    01  ARFIELDS.
          02  AR-AMOUNT   PIC 9(5) PACKED-DECIMAL.
          02              PIC X(10)
$ECOBOL
```

$DEFAULT

```
$DEFAULT
 default statements
 $DEND
```

Figure 5 $DEFAULT Command Syntax

Examples

```
$DEFAULT #1=NEW  #2=OLD
$DEND
```

$GDEF

```
$GDEF #variable symbol $ENDGDEF
```

Figure 6 $GDEF Command Syntax

Examples

```
$GDEF
$GDEF  #RATE='13.5'  $ENDGDEF
```

$GSET

```
$GSET #variable symbol $ENDGSET
```

Figure 7 $GSET Command Syntax

Examples

```
$GSET #NAME='ABC INC.' $ENDGSET
```

$IF

```
$IF #variable symbols
  macro or COPY statements
$ELSE
  macro or COPY statements
$IFE
```

Figure 8 $IF Command Syntax

Examples

```
$IF #1
  PERFORM #2 TO #3
$ELSE
  PERFORM #4 TO #5
$IFE
```

$IFVALUE

```
$IFVALUE #variable symbol {{EQ | NE | LT | LE | GT
  | GE value} | NEGATIVE | POSITIVE | NUMERIC}
  macro statements
$ELSE
  macro statements
$IFE
```

Figure 9 $IFVALUE Command Syntax

Examples

```
$IFVALUE #VALUE1 EQ 2
  MOVE #FLD1 TO #FLD2
  #FLD3 = FLD4 + 5
$ELSE
  MOVE #FLD5 TO #FLD6
$IFE
```

$PRINT

```
$PRINT [#variable symbol]
```

Figure 10 $PRINT Command Syntax

Examples

```
$PRINT
```

$SET

```
$SET #variable symbol  $ENDSET
```

Figure 11 $SET Command Syntax

Examples

```
$SET #DIVISION=42 $ENDSET
```

%macroname

SSL is the default value for VSE.

```
%macroname [PDS | SSL | LIB | PAN | CON | NDV]!variable symbols!
```

Figure 12 %macroname Command Syntax

Examples

```
%SRT
%SRT #KEY=YES
%SRT LIB #SEQ2=YES #SEQ3=NO
```

%sublibrary

The default sublibrary is D.

```
%[sublibrary.] bookname [SLL|LIB|PAN|CON] variable symbol
```

Figure 13 %sublibrary Command Syntax

Examples

```
%A.SRT
%A.SRT PAN
```

ACCEPT

```
ACCEPT
```

Figure 14 ACCEPT Command Syntax

ALLOCATE

```
ALLOCATE [FILE | FILES] [DYNAMICALLY] dataname
  STATUSFLAG dataname
```

Figure 15 ALLOCATE Command Syntax

Examples

```
ALLOCATE FILE  DYNAMICALLY INFILE
STATUSFLAG INSTATUS
```

ARRAY

```
ARRAY arrayname STATUS dataname  F
element-length {DIMENSION | DIM} (n1 n2 [n3]
[n4])
[FILL fillchar] [RETAIN | NORETAIN]
```

Figure 16 ARRAY Command Syntax

Notes

Command Syntax Item	Item Information
arrayname	1 to 8 alphanumeric characters.
length	An integer between 1 and 32767.
[n4]	An integer between 1 and 999999.
fillchar	One alphanumeric character or hexadecimal literal.

Examples

```
ARRAY HISTORY  F 30  STATUS HISTAT  DIM (10 12)
```

BINSEARCH

Data name of a field containing the search value.

```
BINSEARCH tablename argument-name
```

Figure 17 BINSEARCH Command Syntax

Examples

```
BINSEARCH ACCTS ACCTNO
```

CALL

```
CALL subroutine name [nK]  [CDLOAD|NOCDLOAD]
[USING !|dataname|literal|D’dataname|E’dataname
|L’dataname|O’dataname|P’dataname|T’dataname!...]
```

[nK]—VSE default is 4K.

Figure 18   CALL Command Syntax

Examples

```
CALL CHECKER
CALL CONVA 8K USING TRANS BALANCE
```

CASE

```
CASE dataname [index] !WHEN condition
imperative!...   {ELSE | OTHERWISE} imperative]
ENDCASE
```

Condition is the same as for the IF statement.

Figure 19   CASE Command Syntax

CLOSE

```
 CLOSE filename
```

Figure 20   CLOSE Command Syntax

Notes

Command Syntax Item	Item Information
filename	Used only for EXIT-supported files.

COMBINE BITS

```
COMBINE BITS {dataname [index] | literal}
{EXCLUSIVE OR | EXOR | OR | AND}
  {dataname [index] | literal}
  [INTO dataname [index]]
```

Figure 21 COMBINE BITS Command Syntax

Examples

```
COMBINE BITS X'F0F0' OR ACCOUNT
COMBINE BITS X'000000000C' AND ACCOUNT INTO SAVACCT (INZ)
```

CONTROL

Data names are specified in minor to major order.

```
{CONTROL | SUBTOTAL} [BY] !dataname!...
```

Figure 22 CONTROL Command Syntax

Examples

```
CONTROL ACCOUNT
SUBTOTAL BY TRANS ACCOUNT
```

COPY

```
{COPY | COPYE | COPYC | COPYP | COPYL| COPYN}
 { membername |   [sublibrary.] bookname }
 [[!{#a  | #n} = {dataname  | literal}!...] |
[COBOL] {REPLACING operand1 BY operand2}]
```

Figure 23 COPY Command Syntax

Notes

Command Syntax Item	Item Information
sublibrary. bookname	The default sublibrary is D.
REPLACING operand1 BY operand2	See $COBOL, $ECOBOL.

Examples

```
COPY ARDEF
COPY EMPMSTR COBOL
COPY SLSOR101 COBOL REPLACING XYZ by ’C101-’
COPYE PAYDEF #1 = 279   #2 = HOUR
COPYP - copy from CA-PANVALET library
COPYL - copy from CA-LIBRARIAN library
COPYC - copy from CONDOR library
COPYN - copy from ENDEVOR library
```

COPYDB2

Requires the IBM DB2 subsystem.

```
COPYDB2 tablen
ame
```

Figure 24 COPYDB2 Command Syntax

Examples

```
COPYDB2 EMPTABLE
```

dataname definitions

```
dataname length [start] [CH | NU | PD | BI]
[decimals] [[E | A | Z | B | P | D | S] |
[NB | Q]| [W | NP] | [U | NZ] | [Y | NE] |
[X | NA] | [F | DE] | [G | NDE] | [K | DNE] |
[H | DA] | [J | NDA] | [L | DNA] | [D4 |V]]
[{[printdigits].printdecimals | printdigits}]
[{ROUND | ROUNDED}] [(columnheading)]
[LEFT | RIGHT] [VALUE {nnn | ’xxx’ | X’xx’ | ALL
’x’ | ALL X’xx’ | NULL | LOWVALUES |
HIGHVALUES} [REINIT]] [{SUM |NOSUM}]
```

Figure 25 dataname definitions Command Syntax

Examples

```
SACCOUNT 2
ACCOUNT 2 182
NAMADR 75 85 (NAME AND ADDRESS)
INSTLBAL 6 191 PD 2 A (INSTALL’BALANCE)
COMMENT 3 (BALANCE’GE & 300.00’OR’LT $0.00) VALUE ’ ’ REINIT
JBALANCE 5 10 PD 2 A 8.0 ROUND (BALANCE WITH’SURCHARGE) VALUE 0 REINIT
```

dataname assignment

SUM, SUMn, and index are mutually exclusive.

```
                                                      {+}
dataname[index] = [!(!...]{[SUM }] dataname[index]}  !{-}
                          {[SUMn}]                 }   {*}
                          {literal                 }   {/}
                                                      {**}
{[SUM }]1 dataname[index]}!...][!)!...) [{ROUND|ROUNDED}]
{[SUMn}]                 }
{literal                 }
```

Figure 26 dataname assignment Command Syntax

Examples

```
JBALANCE = JBALANCE * 1.10
AVG1 = SUM JBALANCE/TALLY
DIFFP = (SUM DIFF/SUM ADJBUD) * 100
ADJBUD = YTDBUDGET * (INFLA/1.12)
TOTAL = (AVG1 + 10) * ((BAL/7)-20)
AVG1 = SUM BALANCE/TALLY ROUNDED
```

DELETE

```
DELETE {tablename |  arrayname}
```

Figure 27 DELETE Command Syntax

Examples

```
DELETE ACCTS
```

DOUNTIL

Condition is the same as for the IF statement. See page 44.

```
DOUNTIL conditional imperative  ENDDO
```

Figure 28 DOUNTIL Command Syntax

DOWHILE

Condition is the same as for the IF statement.

```
DOWHILE conditional imperative ENDDO
```

Figure 29 DOWHILE Command Syntax

DYLaaaaaaa

```
DYLaaaaaaa [decimals] [E | A | Z | B | D | P]
[{[printdigits].printdecimals | printdigits}]
[ROUND | ROUNDED] [(column heading)]
```

Figure 30 DYLaaaaaaa Command Syntax

Examples

```
DYLCOUNT1 5 A 6.2 ROUND
```

EDIT

```
EDIT dataname [index] INTO dataname [index]
[USING {hexliteral [E | A | Z | B | P | D | S] |
[NB | Q] | [W | NP] | [U | NZ] | [Y | NE] |
[X | NA] | [F | DE] | [G | NDE] | [K | DNE] |
[H | DA] | [J | NDA] | [L | DNA] | [D4 |V] }]
```

Figure 31 EDIT Command Syntax

Examples

```
EDIT AVG1 INTO JBALANCE
EDIT JBALANCE INTO ABALANCE USING E
EDIT CODE INTO TCODE (INX) USING X'402021206B202060'
```

EXEC SQL

VISION:Results Interface to DB2 or SQL/DS is required.

```
EXEC SQL
  sql statements
ENDEXEC
```

Figure 32	EXEC SQL Command Syntax

EXITCASE

```
EXITCASE
```

Figure 33	EXITCASE Command Syntax

EXITDO

```
EXITDO
```

Figure 34	EXITDO Command Syntax

FILE (BDAM Files)

```
FILE filename BDAM
  {RELTRK dataname2 | RELBLK dataname2} STATUS
  dataname2 {F | V | U} recordsize
  {INPUT | OUTPUT FROM filename | IO}  {RANDOM
  dataname2  {CH | NU | BI | PD}
  [decimals] | SEQUENTIAL [dataname2 {CH | NU | BI |
  PD} [decimals]]}
  [KEYLEN  nnn]  [COUNT dataname2]  [LENGTH
  dataname2] [DYNAM] [SYSnnn]
  {DISK {2311 | 2314 | 3330 | 3340 | 3350}} {EXTENTS
  nn}
```

Figure 35	FILE (BDAM Files) Command Syntax

Notes

Command Syntax Item	Item Information
KEYLEN nnn	From 1 to 255.
SYSnnn	Specify logical unit from 0 to 240 (VSE only).
EXTENTS nn	Default is 1 extent.

Examples

```
FILE DAFILE  BDAM  OUTPUT  FROM DAFILE  F 35
  RANDOM  RELBLK BLKNUM  STATUS BDAMSTAT
```

FILE (EXIT Files)

```
FILE filename EXIT modulename [nK]
 [SEQUENTIAL] [{INPUT | OUTPUT FROM filename |IO}]
 [{FB | VB | S | F | V | U | SB}
 [recordsize [blocksize]]]  [SIZE nnnn]  [PARM
 ({WORK | dataname3 ...})]
 [DYNAM] {STATUS dataname2]  [TAPE]  [SYSnnn]
 {DISK {2311 | 2314 | 3330 | 3340 |
 3350 | 3375 | 3380 | FBA}}
```

Figure 36   FILE (EXIT Files) Command Syntax

Notes

Command Syntax Item	Item Information
SIZE nnnn	From 1 to 32767.
SYSnnn	Specify logical unit from 1 to 240 (VSE only).

Examples

```
FILE TEMPFL VB 75 1508 OUTPUT FROM TEMPFL EXIT DYLIOU
```

FILE (ISAM Files)

```
FILE filename ISAM
 {INPUT | OUTPUT FROM filename | IO}  [{FB | VB | F
 | V}  [recordsize [blocksize]]]
 {RANDOM dataname2 {CH | NU | BI | PD}  [decimals]
 |SKIP dataname2 {CH | NU | BI | PD}
 [decimals] | SEQUENTIAL} [KEYLEN nnn [KEYLOC
nnnn]]  [POSITION dataname2]
 [DYNAM] [STATUS dataname2] [COUNT dataname2]
 [LENGTH dataname2]  [BYPASS nn]  [DROPERR]
 [DUMMY]  [NULL]  [RETAIN | NORETAIN] [{MODIFY
modulename  [nK] }  [SIZE nnnnn]
 [PARM ({IOAREA | WORK | dataname3...})]]
 {DISK {2311 | 2314 | 3330 | 3340}} [EXTENTS nn]
 [MASTER] [CYLOFL nn] [SYSnnn] [ONEBUFF]
```

Figure 37 FILE (ISAM Files) Command Syntax

Note: IBM withdrew support for ISAM beginning with z/OS r1.7. Therefore, VISION:Results cannot support access to ISAM data sets under operating systems after z/OS r1.6.

Notes

Command Syntax Item	Item Information
KEYLEN nnn	From 1 to 255.
KEYLOC nnnn	Required if file is being defined for VSE; not required for z/OS ISAM. 1 to 4095.
BYPASS nn	Bypass 1 to 99 blocks of data at the start of the file before processing the first record.
SIZE nnnnn	From 1 to 32767.
WORK	WORK is valid only for MODIFY and when SIZE is specified.
EXTENTS nn	Default is 3 extents.
CYLOFL nn	Required if number of cylinder overflow tracks on each cylinder is not zero.
SYSnnn	Specify logical unit from 0 to 240.

Examples

```
FILE OUTFILE ISAM OUTPUT FROM INFILE FB 120 1200 KEYLEN 3 KEYLOC 1
FILE ISAMFL ISAM FB 70 700 RANDOM SKEY KEYLEN 3 KEYLOC 1 STATUS ISTAT
FILE ISAMFIL ISAM FB 70 700 DISK 3340 RANDOM SKEY KEYLEN 3 KEYLOC 1
```

FILE (PDS Files)

```
FILE filename PDS dataname2
 MEMBER dataname2  [NEWNAME dataname2]  STATUS
 dataname2
 [INPUT|OUTPUT FROM filename]
 {FB | VB | F | V | U}  recordsize  [blocksize]
 [COUNT dataname2]  [LENGTH dataname2] [DYNAM]
```

Figure 38 FILE (PDS Files) Command Syntax

Examples

```
FILE PDSOURCE PDS DSNAREA FB 80 320 MEMBER MEMNAME
 COUNT MEMCOUNT STATUS PDSTATUS
FILE SYSOURCE PDS DSNAREA U MEMBER SYSNAME LENGTH
 SYSLNGTH STATUS SYSTATUS
FILE LIBFILE PDS DSNAREA FB 80 MEMBER LIBNAME OUTPUT
 FROM LIBFILE STATUS LIBSTAT
```

FILE (Sequential Files)

```
FILE filename
 [SEQUENTIAL]  {INPUT | OUTPUT FROM filename}  [{FB
| VB | S | F | V | U | SB}
 [recordsize [blocksize]]]  [{CARD | CARDS}
[{INTERPRET|INSTEL|SELECT|JCL}]]
 [STATUS dataname2]  [COUNT dataname2]  [LENGTH
dataname2}  [BYPASS nn]
 [DROPERR]  [DUMMY]  [NULL]  [RETAIN | NORETAIN]
[{MODIFY modulename [nK]}
 [SIZE nnnnn]  [PARM ({IOAREA
| WORK | dataname3...})]]  [DYNAM]
 [DISK [2311 | 2314 | 3330 | 3340 | 3350 | 3375 | 3380 | FBA]]
 [DISKETTE | {DATAVER | NOFEED | PROTECT | SECURE | VERIFY | VOLVER}]
 [NUMFILES dataname2]  [SYSnnn]  [ONEBUFF]  [TAPE
[{NL | SL | T}]
 [REWIND | NORWD | UNLOAD}]  [n]  [NUMFILES dataname2]]
```

Figure 39 FILE (Sequential Files) Command Syntax

Notes

Command Syntax Item	Item Information
BYPASS nn	Bypass 1 to 99 blocks of data at the start of the file before processing the first record.
SIZE nnnnn	From 1 to 32767.
WORK	WORK is valid only for MODIFY and when SIZE is specified.
SYSnnn	Required if file is on tape (VSE only). Specify logical unit from 0 to 240 (VSE only).

Examples

```
FILE ARFILE
FILE ARFILE  FB 352 5280
FILE OUTFILE  FROM OUTFILE  OUTPUT  FB 25 2500
FILE ARFILE  FB  352 5280  DISK  3330  SYS027
```

FILE (SSL Files)

```
 FILE filename SSL
     BOOK dataname2 STATUS dataname2 {COUNT dataname}
```

Figure 40 FILE (SSL Files) Command Syntax

Examples

```
FILE ARTEXT SSL BOOK BOOKNAME STATUS TEXTSTAT
FILE RTEXT SSL BOOK SOURCEPGM STATUS SOURCESTAT
 COUNT SOURCECNT
```

FILE (VSAM Files)

```
FILE filename [VSAM | {KSDS | RRDS | ESDS}]
[PASSWORD {'password' | dataname2}]  {INPUT |
OUTPUT FROM filename | IO}  [{F | V | S}
[recordsize]] {{RANDOM | RANDOMX} dataname2 {CH | NU | BI |
PD}  [decimals]  | {SKIP | SKIPX} dataname2
{CH | NU | BI | PD} [decimals] | SEQUENTIAL}
[KEYLEN nnn [KEYLOC nnnn]]
```

Figure 41 FILE (VSAM Files) Command Syntax (Page 1 of 2)

```
[{RELBYTE | RELBYTEX} dataname2] [RELREC dataname2]
[POSITION dataname2] [REUSE] [ERASE dataname2]
[PARTKEY nnn] [STATUS dataname2]
[COUNT dataname2]  [LENGTH dataname2] [DROPERR]
[DUMMY] [RETAIN | NORETAIN] [DYNAM]
[{MODIFY modulename [nK] }  [SIZE nnnnn] [PARM
({IOAREA | WORK | dataname3...})]]
[SYSnnn]
```

Figure 41 FILE (VSAM Files) Command Syntax (Page 2 of 2)

Notes

Command Syntax Item	Item Information
KEYLEN nnn	From 1 to 255. Required for VSAM KSDS RLS files.
KEYLOC nnnn	From 1 to 4095; not required when taken from IDCAMS definition. Required for VSAM KSDS RLS files if the key position is not position 1 in the file.
SIZE nnnnn	From 1 to 32767.
WORK	WORK is valid only for MODIFY and when SIZE is specified.
SYSnnn	Specify logical unit from 0 to 240.

Examples

```
FILE FKSDSI KSDS F
FILE FKSDS VSAM OUTPUT FROM SEQIN
FILE VSAMIN ESDS V COUNT KCOUNT LENGTH KLENGTH
FILE TRANS VSAM
FILE OUTFILE VSAM LENGTH OUTLEN OUTPUT FROM OUTFILE
FILE VRRDS1 VSAM OUTPUT FROM FILEIN RELREC RELRECNUM
FILE EXAMPLE VSAM IO RANDOM RELRCD STATUS STATFLG
```

FILLER

```
FILLER length [start] [CH | NU | PD | BI] [decimals] [VALUE literal [REINIT]]
```

Figure 42 FILLER Command Syntax

Examples

```
FILLER 1
FILLER 2 VALUE 'GN'
FILLER 4 81 VALUE 0
FILLER 5 PD 2  VALUE 89.95 REINIT
```

FIN

```
FIN
```

Figure 43 FIN Command Syntax

GOTO

```
GOTO {tagname | ACCEPT | REJECT | NEXT | STOP}
```

Figure 44 GOTO Command Syntax

Examples

```
GOTO BEGIN
```

GETCOM

VSE only.

```
GETCOM dataname
```

Figure 45 GETCOM Command Syntax

HEX

```
HEX ! {filename | dataname | (index) | 'literal!'}
 [LENGTH {dataname | nnnn}] !...
```

Figure 46 HEX Command Syntax

Examples

```
HEX ARFILE
HEX ACCOUNT
HEX ARFILE LENGTH ARLEN
HEX NAME LENGTH 10
HEX DATA4(INX)
HEX DATA1 DATA2
```

HEXPRINT

```
HEXPRINT! {filename! | dataname! | index |
’literal’}
 [LENGTH {dataname | nnnn}]!...
```

Figure 47 HEXPRINT Command Syntax

Examples

```
HEXPRINT ’ABCDZ’
HEXPRINT ARFILE LENGTH 25
HEXPRINT DATA1 DATA2
```

IF

```
IF [NOT] condition [!{AND|OR}] [NOT]
 condition! ...] imperative
 [{ELSE | OTHERWISE} imperative] ENDIF
```

Figure 48 IF Command Syntax, Statement Example 1

Examples

```
IF ACCOUNT EQ ’BO’ THRU ’IO’ AND TRANS EQ 3217801
 GOTO GHIJ ENDIF
IF ACCOUNT EQ ’KO’ AND TRANS EQ 7314213 GOTO ABC
 ENDIF
IF DIV EQ 2 AND HOURS LT 40 GOTO REPTLINE ENDIF
IF DIV EQ 3 AND DEPT EQ 106 NEXT ELSE REJECT ENDIF
IF DIV EQ 2 AND HOURS LT 40 OR DIV EQ 3 AND DEPT EQ
 106
  LIST DIV DEPT EMPNO NAME ELSE REJECT ENDIF
```

```
IF [!(!...] [NOT] condition !{AND | OR} [!(!...]
[NOT] condition !...!{AND | OR}
   [NOT] condition !...[!)!...] [!)!...]
imperative [{ELSE | OTHERWISE} imperative] ENDIF
```

Figure 49 IF Command Syntax, Statement Example 2

Examples

```
IF (AVG1 EQ 0 OR AVG2 GT 100) AND BAL LT 20 MOVE '—-' TO COMMENT ENDIF
IF (AVG1 GT 10 OR AVG3 EQ 20 THRU 40) AND ((BAL GT
 100 OR AMOUNT LT 500) AND
   STATE NE 'CA') NEXT ELSE REJECT ENDIF
```

```
IF [NOT] [MATCHED [file1 file2] | condition] [
!{AND |OR} [NOT]
          [ADVANCED file1]
          [DUPLICATE file1]
          [FIRSTDUP file1]
          [LASTDUP file1]
          [REPLICA file1]

  [MATCHED filex filey | condition!...] imperative {ELSE|OTHERWISE} imperative ENDIF
  [ADVANCED filen]
  [DUPLICATE filen]
  [FIRSTDUP filen]
  [LASTDUP filen]
  [REPLICA filen]
```

Figure 50   IF Command Syntax, Statement Example 3

Examples

```
IF MATCHED AND KEY1A EQ 123 LIST RECA ENDIF
IF DUPLICATE FILEA LIST RECB ENDIF
```

```
IF [NOT] SAMPLING nn imperative [{OTHERWISE |
ELSE} imperative] ENDIF
```

Figure 51   IF Command Syntax, Statement Example 4

Examples

```
IF SAMPLING 4 NEXT ELSE REJECT ENDIF
IF SAMPLING 2 ACCEPT ELSE GOTO SAMP3 ENDIF
```

SUM, SUMn, and index are mutually exclusive.

```
IF condition imperative [{ELSE | OTHERWISE} imperative] ENDIF

Where condition is:
                                       {EQ}
                                       {NE}
[{SUM  | SUMn}] dataname[index]  [NOT] {LT} [{SUM
|SUMn}] dataname[index]
                                       {LE}
                                       {GT}
                                       {GE}
or

[{SUM  |SUMn}] dataname[index]  [NOT] {EQ | NE} ![{SUM  | SUMn}] dataname[index]

    [[THRU | THROUGH] [{SUM  | SUMn}] dataname[index]]!...

or
                                       {EQ}
                                       {NE}
[{SUM | SUMn}]1 dataname[index]  [NOT] {LT} literal
                                       {LE}
                                       {GT}
                                       {GE}
or

[{SUM  | SUMn}] dataname[index]  [NOT] {EQ | NE}
!literal [[THRU | THROUGH] literal]!...

or

[{SUM  | SUMn}] dataname[index] [NOT]  {EQ | NE}
![{SUM  | SUMn}] dataname[index]

[[THRU | THROUGH] [{SUM | SUMn}] dataname[index]]
!... !literal [[THRU | THROUGH] literal]!...
```

Figure 52 IF Command Syntax, Statement Example 5

Examples

```
IF ACCOUNT EQ 'NA' REJECT ENDIF
IF ACCOUNT EQ 'NA' 'FO' 'IO' REJECT ENDIF
IF ACCOUNT EQ 'NA' 'FO' THRU 'IO' NEXT ELSE REJECT
   ENDIF
IF BALANCE GE 300 ACCEPT ENDIF
IF TRANS EQ ITRANS MOVE 0 TO BAL ENDIF
IF SUM BAL EQ 0 MOVE '***' TO COMMENT ENDIF
IF BAL EQ SAVE1 SAVE2 THRU SAVE3 REJECT ENDIF
```

SUM, SUMn, and index are mutually exclusive.

```
IF [{SUM | SUMn}] dataname [index] [NOT] {MIXED |
ONES} literal imperative
   [{OTHERWISE | ELSE} imperative] ENDIF
```

Figure 53 IF Command Syntax, Statement Example 6

Examples

```
IF SWITCH ONES X'03' NEXT ELSE REJECT ENDIF
IF FLAG ONES X'80' MOVE 0 TO COUNT ENDIF
IF SWITCH MIXED X'0F' NEXT ELSE REJECT ENDIF
```

SUM, SUMn, and index are mutually exclusive.

```
IF [{SUM   | SUMn}] dataname[index]  [NOT]
{POSITIVE | NEGATIVE | NUMERIC} imperative
  [{OTHERWISE | ELSE} imperative] ENDIF
```

Figure 54 IF Command Syntax, Statement Example 7

Examples

```
IF TRANS NUMERIC GOTO OK ELSE GOTO BAD ENDIF
IF BAL NEGATIVE MOVE '***' TO COMMENT ENDIF
IF BALANCE POSITIVE GOTO ABC ELSE GOTO NEGTEST ENDIF
```

IQFILE

```
  IQFILE filename [RETAIN] [VB reclen blklen] [COUNT dataname]
     [LENGTH dataname] [STATUS dataname]
     DECLARE_QUERY

     (VISION:Inquiry statements)

     END_QUERY
```

Figure 55 IQFILE Command Syntax

IQREAD

```
IQREAD filename
```

Figure 56 IQREAD Command Syntax

ITERATE

```
ITERATE
```

Figure 57 ITERATE Command Syntax

LCPRINT

```
LCPRINT! {filename | dataname!... | (index) | 'literal'}
        ![LENGTH {dataname | nnnn}]!...
```

Figure 58 LCPRINT Command Syntax

Examples

```
LCPRINT TEXTFILE
LCPRINT TEXTLINE LENGTH 20
LCPRINT 'lower case'
LCPRINT DATA(INX)
LCPRINT DATA1 DATA2
```

LETTER

```
LETTER nn  [FROM nn TO nn | FROM nn}]
```

Figure 59 LETTER Command Syntax

Notes

Command Syntax Item	Item Information
`FROM nn TO nn`	Invokes line ranges (1-99) appended to LTD command.

Examples

```
LETTER 01
LETTER 02 FROM 5 TO 8
```

LINEAR

```
 LINEAR nn  XNAME dataname YNAME dataname [TITLE dataname]
  [XSCATTER [scaling factor] FROM starting value BY interval size]
        [FOR number of intervals]
  [YSCATTER [scaling factor] FROM starting value BY interval size]
        [FOR number of intervals]
```

Figure 60 LINEAR Command Syntax

Notes

Command Syntax Item	Item Information
`TITLE dataname`	The title cannot exceed 70 characters.
`scaling factor`	Defaults to unit of 1; otherwise code values from 1 to 4, where: 1=thousands 2=millions 3=billions 4=trillions
`FROM starting value`	A numeric value with a maximum of 14 characters; can be negative. Unit value determined by scaling factor specified for this variable.
`BY interval size`	A positive numeric value with a maximum of 14 characters. Unit value determined by scaling factor specified for this variable.
`XSCATTER number of intervals`	An integer between 1 and 30; default is 30.
`YSCATTER number of intervals`	An integer between 1 and 20; default is 20.

Examples

```
LINEAR 01  XNAME SALES  YNAME PROFIT
     XSCATTER 2  FROM 0  BY .5  FOR 24
     YSCATTER 2  FROM 0  BY .2  FOR 18
LINEAR 02  XNAME SALES  YNAME PROFIT
```

LIST

```
LIST ![SUPPRESS | PRIOR | SUM | SUMn] [(]
!{dataname | (index) | literal | TALLY }
  [[E | A | Z | B | P | D | S] | [NB | Q]
| [W | NP] | [U | NZ] | [Y | NE] | [X | NA]
| [F | DE] | [G | NDE] | [K | DNE] | [H | DA] | [J
| NDA] | [L | DNA] | [D4 |V]]
[AT {n | dataname [+n]} ]
  [(column heading)] !...[)] !...
  [WITH [{EJECT | n} BEFORE] [[AND] [{EJECT | n}] AFTER]]
```

Figure 61 LIST Command Syntax

Notes

Command Syntax Item	Item Information
n BEFORE n AFTER	The default is 0 before and 1 after.

Examples

```
LIST NAME
LIST SUM BALANCE WITH 2 AFTER
LIST 'NUMBER OF RECORDS PRINTED' AT ACCOUNT TALLY AT
 BALANCE
LIST SUPPRESS ACCOUNT TRANS NAME BALANCE E JBALANCE
 E COMMENT
LIST BALANCE (BALANCE'AMOUNT)
LIST NAME AT 5 ADDRESS AT 35 WITH 2 AFTER
LIST PRIOR (ACCT NAME)
LIST FIELD1 (INX) (HEADER1) FIELD2
```

LTD

LTH

```
LTH nn [nnn WIDE] [nn LONG] [NOEJECT]
[LASER[nn]] [INDENT nnn] [DYLEXTxx nnnK
    [WRITEALL]] [ASA] [IDLENGTH nn] [DYLTRxx [BY yyy]
    [DISK {2311 | 2314 | 3330 | 3340 | 3350 | 3375
| 3380 | FBA} | TAPE} SYSnnn]
    [TWOUP [nnn BETWEEN] [SWITCH] [LEFTSAVE
nnnK]]  [OPTIMIZE | NOOPTIMIZE]

LTD[nn] [@(NEWPARA) [Lnn | Inn] | @(ASIS)]
[@(SPn) | @(CHnn) | @(EJECT) ] [@nnn]
    [@dataname[(edit code[,printsize])]] [@(CSnn)]

ENDLTD
```

Figure 62   LTD and LTH Command Syntax

Notes

Command Syntax Item	Item Information
LTH nn	Unique application number given to each letter and body definition.
nnn WIDE	Characters per line. An integer value from 1 to 260; default is 80.*

Command Syntax Item	Item Information
nn LONG	Lines per page. The default is 56.
LASER[nn]	Maximum number of character sets per letter. Maximum is 4 and default is 4.
INDENT nnn	Shifts every line of the letter nnn spaces to the right.*
DYLEXTxx nnnK	The size of the subroutine, DYLEXTxx, in K.
WRITEALL	Exit subroutine handles the writing of all records; no records are written to AUDPRINT/SYS009.
IDLENGTH nn	Contains the length of the identification field.*
DYLTRxx BY yyy	Output letter file and blocking factor. xx = AA-ZZ, 00-99; default blocking factor is 10.
TAPE} SYSnnn	Required by VSE if DYLTRxx is used.
LEFTSAVE nnnK	Used with TWOUP option (two letters per page). nnn BETWEEN = number of bytes between left- and right-side letters; default is 5. SWITCH determines when and where the next letter command executes. LEFTSAVE nnnK is the amount of memory in K for holding the letter on the left hand side; default is 10.
LTD[nn]	LTD appended with numeric value used in LETTER command FROM-TO option.
@(ASIS)	@ sign precedes text edit option that, with only one exception (@nnn), is enclosed within parentheses, as shown: @(SPn) Line spacing n = 1-9. @(ASIS) Override automatic text editing. @(NEWPARA) Begin text that follows on new line. @(EJECT) Page eject at this line of the letter. @(CHnn) Channel control spacing; nn=1-12. @nnn nnn equals numeric value for text print positioning (nnn value plus text cannot exceed characters per line WIDE).

Command Syntax Item	Item Information	
printsize	Variable information must be prefixed with an @ followed by the data name @dataname. The data name has a minimum length of 2 and must start with an alphabetic character A-Z.	
	@DYLDATE	DYLDATE is a reserved word containing current date in the form MM/DD/YY.
	$@dataname(E)	Data names containing variable information for insertion into the letter body can have an edit code suffix. For edit code definitions, see Chapter 5: Programming Tips.
	@AMOUNT(E,10)	Additionally, a print size can follow the edit code to force column data to line up in each detail line specified. The print size must immediately follow the edit code and be separated by a comma, as shown in the example on page 52.
ENDLTD	Required to end the body of the letter.	

LTH, LTD, and ENDLTD must always be the first commands coded on their respective statements.

Examples

```
LTH 05  80 WIDE  45 LONG  NOEJECT  INDENT 4
       IDLENGTH 4
LTD1   Text of letter (LTD statements) here,
       including insertion of variable data.
       LTD  @(SPn)
       LTD  @(ASIS)
       LTD212
       LTD  @(NEWPARA)
       LTD  @dataname       @DYLDATE
       LTD
LTD
       LTD312
       ENDLTD15
```

MATCH

```
MATCH filename1 KEY1 dataname [KEY2 dataname KEY3
dataname...KEY9 dataname]
      filename2 KEY1 dataname [KEY2 dataname KEY3
dataname...KEY9 dataname]
      [filenamen KEY1 dataname [KEY2 dataname KEY3
dataname...KEY9 dataname]
      SETREAD {MASTERUP | COMPAREN | ORIGINAL}
[USING dyl282mat subroutine]
Special IF statements
(IF  [NOT] [ADVANCED | FIRSTDUP | LASTDUP |
MATCHED | DUPLICATE])
```

Figure 63 MATCH Command Syntax

Examples

```
MATCH FILEA KEY1 KEY1A KEY2 KEY2A
      FILEB KEY1 KEY1B KEY2 KEY2B
```

MERGE

```
MERGE filename1 KEY1 dataname [KEY2 dataname KEY3
dataname...KEY9 dataname]
      filename2 KEY1 dataname [KEY2 dataname KEY3
dataname...KEY9 dataname]
      [filenamen KEY1 dataname [KEY2 dataname KEY3
dataname...KEY9 dataname]
   AREA  dataname [FROM dataname] [LENGTH
dataname] [USING {DY282MER | subroutine}]
```

```
Special IF statement here.
```

(For complete format, see IF on page 44.)

```
(IF [NOT] REPLICA)
```

Figure 64 MERGE Command Syntax

Examples

```
MERGE TRAN1 KEY1 T2KEY1
    TRAN2 KEY1 T2KEY1
    TRAN3 KEY1 T3KEY1
   AREA OUTREC FROM SOURCEFL LENGTH OUTLENX
```

MOVE

SUM, SUMn, and index are mutually exclusive.

```
MOVE [SUM  | SUMn] dataname[index] [LENGTH {n |
dataname[index]}] TO [SUM  | SUMn]1
     dataname[index]
```

Figure 65 MOVE Command Syntax, Statement Example 1

Examples

```
MOVE SUM BALANCE TO OUTBALANCE
MOVE NAME LENGTH 10 TO SAVENAME (INW)
MOVE ADDNAME LENGTH 20 TO BUFFER1
```

```
MOVE [SUM  | SUMn] dataname[index] TO [SUM | SUMn] dataname[index]
     [LENGTH  {n | dataname[index]}]1
```

Figure 66 MOVE Command Syntax, Statement Example 2

Examples

```
MOVE SPACES TO RECORDAREA LENGTH 100
MOVE SUM BALANCE TO BALANCEOUT (INW)
MOVE FIELDA(INW) TO FIELDB(INX) LENGTH FLDLENGTH(INZ)
```

```
MOVE literal TO [SUM | SUMn] dataname[index]
```

Figure 67 MOVE Command Syntax, Statement Example 3

Examples

```
MOVE '22AB' TO TABLE(INY)
MOVE  0  TO AVG1
MOVE  3  TO INW
MOVE  'COMMENT' TO REMARKS
```

```
MOVE {CORR |CORRESPONDING} qualifier TO qualifier [SUPPRESS]
```

Figure 68 MOVE Command Syntax, Statement Example 4

Examples

```
MOVE CORRESPONDING ARFILE TO ARTBL
MOVE CORR AFILE TO BFILE
MOVE CORR ACCTFILE TO ACCTBL SUPPRESS
MOVE {NUMERIC | ZONE | OFFSET} {SUM dataname |
SUMn dataname | literal | dataname[index]} TO
[{SUM}] {literal         }    [{SUMn}]1
```

Figure 69 MOVE Command Syntax, Statement Example 5

Examples

```
MOVE NUMERIC ACCOUNT TO NUMACCT
```

```
MOVE [UNSIGNED] dataname to [UNSIGNED | ABSOLUTE | NORMALIZED] dataname
```

Figure 70 MOVE Command Syntax, Statement Example 6

Examples

```
MOVE UNSIGNED BALANCE TO OUTBALANCE
```

MQBACK

```
MQBACK  hconn compcode reason [errormsg]
```

Figure 71 MQBACK Command Syntax

MQBEGIN

```
MQBEGIN  hconn mqbo compcode reason [errormsg]
```

Figure 72 MQBEGIN Command Syntax

MQCLOSE

```
MQCLOSE hconn hobj hoptions compcode reason [errormsg]
```

Figure 73 MQCLOSE Command Syntax

MQCMIT

```
MQCMIT  hconn compcode reason [errormsg]
```

Figure 74 MQCMIT Command Syntax

MQCONN

```
MQCONN qmgr hconn compcode reason [errormsg]
```

Figure 75 MQCONN Command Syntax

MQCONNX

```
MQCONNX  qmgr  mqcno  hconn  compcode  reason [errormsg]
```

Figure 76 MQCONNX Command Syntax

MQDISC

```
MQDISC hconn compcode reason [errormsg]
```

Figure 77 MQDISC Command Syntax

MQGET

```
MQGET  hconn hobj mqmd mqgmo bufferlen buffer datalen compcode reason [errormsg]
```

Figure 78 MQGET Command Syntax

MQINQ

```
MQINQ  hconn hobj  selectorcount selectorstable intattrcount intattrstable charattrlength
charattrs compcode reason [errormsg]
```

Figure 79 MQINQ Command Syntax

MQOPEN

```
MQOPEN  hconn mqod hoptions hobj compcode reason [errormsg]
```

Figure 80 MQOPEN Command Syntax

MQPUT

```
MQPUT hconn hobj mqmd mqpmo bufferlen buffer compcode reason [errormsg]
```

Figure 81 MQPUT Command Syntax

MQPUT1

```
MQPUT1  hconn mqod mqmd mqpmo bufferlen buffer compcode reason  [errormsg]
```

Figure 82 MQPUT1 Command Syntax

MQSET

```
MQSET  hconn hobj selectorcount selectorstable intattrcount intattrstable charattrlength
charattrs compcode reason [errormsg]
```

Figure 83 MQSET Command Syntax

NEWPAGE

```
NEWPAGE
```

Figure 84 NEWPAGE Command Syntax

NEXT

```
NEXT
```

Figure 85 NEXT Command Syntax

ON

```
ON {{END OF {SORTING | INPUT}} | FINAL | CHANGE IN
dataname | {ONE imperative ENDONE}}
```

Figure 86   ON Command Syntax

Examples

```
ON END OF SORTING
ON END OF INPUT
ON FINAL
ON CHANGE IN ACCOUNT
ON ONE MOVE ACCOUNT TO SAVACCT ENDONE
```

OPTION

```
OPTION [$$DUMP]
       [$$KEYLEVEL release#]
       [$$PCB]
       [!USERINXnn dataname!]
       [CATPLANID planname]
       [CATSYSID ssid]
       [CDLOAD | NOCDLOAD]
       [COBAPOS | NOCOBAPOS]
       [COBXSIGN | NOCOBXSIGN]
       [COB2NR | NOCOB2NR]
       [COLUMNS nn nn]
       [CURRENCY {C’x’ |X’nn’}]
       [DATA  numeric literal | alphanumeric
            literal]
       [DB2SYSID  {dataname | ’literal’}]]
       [DISK type]
       [DMAP]
       [DYL4YEAR | NODYL4YEAR]
       [DYNAMDB2 [DB2PLANID {dataname | ’literal’}]
       [EXCEL | NOEXCEL]
       [EXCELPATS | NOEXCELPATS]
       [EXPRTERR | NOEXPRTERR]
       [FREEZDD ddname]
       [FREEZE modulename]
       [IQBATCH name]
       [LE |NOLE]
       [MSGCSECT]
       [NDVRCOM | NONDVRCOM]
       [NDVRENV]
       [NDVRSYS]
       [NDVRTYP]
       [nn LONG]
       [NOSORTAB]
       [NULLON | NULLOFF]
       [NUMCHAR | NONUMCHAR]
       [NUMPD | NONUMPD]
       [OPTLIST | NOOPTLIST]
       [PDSREPA | PDSREPN]
       [{PRINT | HEX | HEXPRINT | LCPRINT } filename]
       [PRINTCTRS]
       [PRINTDIGITS | NOPRINTDIGITS]
       [PRINTEP | NOPRINTEP]
       [{PRINTERR | PRINTERROR | NOERROR}]
       [QLF|NOQLF| QUALIFIERS|NOQUALIFIERS]
       [QUIKJOB]
```

Figure 87   OPTION Command Syntax (Page 1 of 2)

```
[REPORTFILE filename]
[RESTORE modulename [n] [RESTART nnn]DELIM[C’c’|X’xx’]]
[SPOOL | NOSPOOL]
[STATSQL modname PLANID planname SYSTEMID ssid}]
[STRUCTURED | STRUCTURED2 | CONVENTIONAL
     | USERDEFAULT]
[SYSBLOCK | NOSYSBLOCK]
[SYSnnn [filename]]
[TABAREA {ABOVE | BELOW }]
[TIMECOLON | TIMEDOT]
{TOTAL | NOTOTAL}
[TSIZE nnnK]
[VERIFY]
{VSAMCAT | NOVSAMCAT}
[VSEALL | NOVSEALL]
[VSEDISK | NOVSEDISK]
[VSETAPE | NOVSETAPE]
[XREF | XREFA | NOXREF]
[ZDIVORG | ZDIVAB | ZDIVRC]
[260]
```

Figure 87 OPTION Command Syntax (Page 2 of 2)

Examples

```
OPTION HEXPRINT ARFILE
OPTION FREEZE MYPROG
OPTION RESTORE MYPROG DATA ’01/02/96’
OPTION VERIFY COLUMNS  7  71
OPTION SYS016 WORK280 DISK 3350
OPTION $$DUMP
OPTION NOTOTAL
```

PCFILE

VISION:Journey for Windows or Advantage™ VISION:Intraccess™ only.

```
PCFILE fileid
FIELDS !dataname!...
[MAXBLKSIZE nnnnn]
[STATUS dataname]
```

Figure 88 PCFILE Command Syntax

PCWRITE

VISION:Journey for Windows or VISION:Intraccess only.

```
PCWRITE fileid
```

Figure 89 PCWRITE Command Syntax

PERFORM

```
PERFORM  tagname [TO tagname] [VARYING dataname
FROM {dataname | numeric literal}
        TO {dataname | numeric literal} [BY
{dataname | numeric literal}]]
```

Figure 90 PERFORM Command Syntax

Notes

Command Syntax Item	Item Information
numeric literal	The default BY value is 1.

Examples

```
PERFORM BEGIN TO ENDIT VARYING CNT1 FROM 5 TO 10 BY CNT2
```

PICNSAVE

```
PICNSAVE idname USING dataname ... [BY nnnnn]
[SYSnnn] [DISK [2311 | 2314 | 3330 | 3340 | 3350 |
   3375 | 3380 | FBA]] [TAPE]
```

Figure 91 PICNSAVE Command Syntax

Notes

Command Syntax Item	Item Information
BY nnnnn	The default blocking factor is 1.

PRINT

```
 PRINT! {filename | dataname!... | (index) |
'literal!'}
      ![LENGTH {dataname | nnnn}]!...
```

Figure 92 PRINT Command Syntax

Examples

```
PRINT ARFILE
PRINT 'ARFILE DUMP'
PRINT NAME LENGTH 10
PRINT DATA(INX)
PRINT DATA1 DATA2
```

PRINTGEN

```
PRINTGEN
```

Figure 93 PRINTGEN Command Syntax

PRINTNOGEN

```
PRINTNOGEN
```

Figure 94 PRINTNOGEN Command Syntax

PUTCOM

VSE only.

```
PUTCOM dataname
```

Figure 95 PUTCOM Command Syntax

Examples

```
PUTCOM ADDRR
```

QUIT

```
QUIT
```

Figure 96 QUIT Command Syntax

QUITALL

```
QUITALL
```

Figure 97 QUITALL Command Syntax

READ

```
READ filename
```

Figure 98 READ Command Syntax

Examples

```
READ ARFILE
```

READBOOK

VSE only

```
READBOOK filename
```

Figure 99 READBOOK Command Syntax

Examples

```
READBOOK ARTEXT
```

READDIR

```
READDIR filename  [directory area]
```

Figure 100 READDIR Command Syntax

Examples

```
READDIR MASTIN
READDIR MASTIN DIRAREA
```

READMEM

```
READMEM filename
```

Figure 101 READMEM Command Syntax

Examples

```
READMEM MASTIN
```

REDEFINE

```
{REDEFINE | REDEF} [AT] dataname
```

Figure 102 REDEFINE Command Syntax

Examples

```
REDEFINE DATE
REDEF AT SOCSECNO
```

REGRESSION

```
REGRESSION [LINEAR | TREND | SCATTER] nn
```

Figure 103 REGRESSION Command Syntax

Examples

```
REGRESSION 22
REGRESSION LINEAR 01
REGRESSION TREND 02

REGRESSION SCATTER 03
```

REJECT

```
REJECT
```

Figure 104 REJECT Command Syntax

RELSORT

```
RELSORT
```

Figure 105 RELSORT Command Syntax

REPORT

The default is 132 WIDE, 55 LONG, and 5 BETWEEN.

z/OS:

```
REPORT[nnn] [nnn WIDE][DELIM[C'c'|X'nn']] [[AND] nn LONG]AND] nn
BETWEEN]AND] SYS280Rx]HTML ddname [COUNT nnnn][STYLEnn] [HTMLTITL dataname] ] | [{[AND]
[[PLUnnn]DISK [nn] TAPE [nn]]NL | SL | T]
[REWIND | NORWD | UNLOAD]]AND]ASA]
|{DYL280Ra | DYLPRCa}}]
```

Figure 106 REPORT Command Syntax (z/OS)

VSE:

```
REPORT [nnn][nnn WIDE][DELIM[C’c’|X’xx’]][nnn LONG][ nn BETWEEN]
[DYL280Ra | [[ASA][PLUnnn] [DISK [nn] |
[TAPE [nn] [NL|SL|T] [REWIND|NORWD|UNLOAD]]]]
```

Figure 107 REPORT Command Syntax (VSE)

Examples

```
REPORT 10 BETWEEN
REPORT 2 BETWEEN AND 30 LONG
REPORT ASA PLU027 DISK 50
REPORT 80 WIDE DYL280RB              z/OS only
REPORT DYLPRCD
REPORT12
REPORT 100 WIDE DELIM C'$'

WORKAREA
  WEBTITLE 40 CH VALUE "Account Receivables Sample Report"
  REPORT 80 WIDE HTML HTMLOUT STYLE 00 HTMLTITL WEBTITLE
```

REPORTFILE

VSE only.

```
 REPORTFILE {filename! | dataname!... | (index) |
     ’literal!’}
           [LENGTH {dataname | nnnn}]
```

Figure 108 REPORTFILE Command Syntax

Examples

```
REPORTFILE RPFILE
REPORTFILE RPFILE LENGTH 20
REPORTFILE DATA(INX)
REPORTFILE DATA1 DATA2
```

RETRIEVE

```
RETRIEVE {tablename [RANDOM] | arrayname (coordinates)}
```

Figure 109 RETRIEVE Command Syntax

Notes

Command Syntax Item	Item Information
coordinates	Two to four data names containing coordinate values.

Examples

```
RETRIEVE CUST RANDOM
RETRIEVE ACCTS (YEAR MONTH)
```

RETSORT

```
RETSORT
```

Figure 110 RETSORT Command Syntax

SAMPLE

```
SAMPLE nn INTERVAL interval size
starting number  [maximum sample size]
```

Figure 111 SAMPLE Command Syntax, Statement Example 1

Notes

Command Syntax Item	Item Information
interval size	An integer value from 1 to 999999999.
starting number	An integer value from 0 to 999999999.
maximum sample size	An integer value from 1 to 999999999; default is 999999999.

Examples

```
SAMPLE 02  INTERVAL  50 10 500
SAMPLE 34  INTERVAL 100 100,000
```

```
SAMPLE nn RANDOM {population size1 sample size
 desired1 | percent of pop %}
 [SEED original seed number]
```

Figure 112 SAMPLE Command Syntax, Statement Example 2

Notes

Command Syntax Item	Item Information
percent of pop %	Maximum 100%; minimum .00001%. It cannot exceed five digits to the right of the decimal point.
seed number	Maximum seed number is 9999999.

Examples

```
SAMPLE 01  RANDOM 50,000 450
SAMPLE 02  RANDOM 10%
SAMPLE 03  RANDOM .038%  SEED 8642
```

SCATTER

```
SCATTER nn XNAME dataname  YNAME dataname [TITLE
dataname]
    XSCATTER [scaling factor] FROM starting
value BY interval size [FOR number of intervals]
    YSCATTER [scaling factor]5  FROM starting value6  BY interval size7  [FOR number of
 intervals]
```

Figure 113 SCATTER Command Syntax

Notes

Command Syntax Item	Item Information
SCATTER nn	Unique application number from 1 to 99.
dataname	The length of the data name field cannot exceed 70 characters.
XSCATTER scaling factor YSCATTER scaling factor	Defaults to a unit of 1; otherwise code values from 1 to 4, where: 1=thousands 2=millions 3=billions 4=trillions
FROM starting value	A numeric value with a maximum 14 characters; can be negative. Unit value determined by scaling factor specified for this variable.
BY interval size	Positive numeric value with a maximum 14 characters; can be negative. Unit value determined by scaling factor specified for this variable.
FOR number of intervals (X)	An integer between 1 and 30; the default is 30.
FOR number of intervals (Y)	An integer between 1 and 20; the default is 20.

Examples

```
SCATTER 02  XNAME SALES  YNAME PROFIT
  XSCATTER 2  FROM 0  BY .5  FOR 24
  YSCATTER 2  FROM 0  BY .2  FOR 18
```

SEARCH

```
SEARCH tablename argument-name
```

Figure 114 SEARCH Command Syntax

Notes

Command Syntax Item	Item Information
argument-name	Data name of a field containing the search value.

Examples

```
SEARCH CUST CUSTNO
```

SORT

```
 {PD0|SPT|Y2C|Y2z|YR2|Y2S|Y2P|SYR|Y2D|YR1|CMp|PMp|FMp}
 SORT {filename | dataname [recordsize]} [nK] USING !dataname {A | D}
     !... [UNTIL dataname | filename] [WORK nnn]
```

Figure 115 SORT Command Syntax

Notes

Command Syntax Item	Item Information
recordsize nK	The default is 1.
Mp	Valid values are M0-M9, MA, or MB.

Examples

```
SORT ARFILE USING ACCOUNT
SORT SORTREC 40 USING STRANS D UNTIL AREND
SORT ARFILE USING ACCOUNT WORK 3
```

STATEOFF

```
STATEOFF
```

Figure 116 STATEOFF Command Syntax

STATEON

```
STATEON
```

Figure 117 STATEON Command Syntax

STOP

```
STOP
```

Figure 118 STOP Command Syntax

STOPALL

```
STOPALL
```

Figure 119 STOPALL Command Syntax

STORE

```
STORE {tablename [RANDOM] | arrayname
  (coordinates)}
```

Figure 120 STORE Command Syntax

Notes

Command Syntax Item	Item Information
arrayname coordinates	Two to four data names containing coordinate values.

Examples

```
STORE CUST
STORE PHONES (STATE CITY)
STORE HISTORY RANDOM
```

tagname

```
{tagname: | tagname :}
```

Figure 121 Tagname Command Syntax

Examples

```
BEGIN:
ENDIT :
8:
```

TABLE

```
 TABLE tablename  STATUS dataname2 {F entrylen | V
   maxlen LENGTH dataname2}
     {ENTRY | OFFSET} dataname2 [KEYLOC
   {dataname2 | n}4 KEYLEN
     {dataname2 | n}] [FROM filename] {RETAIN |
   NORETAIN}
```

Figure 122 TABLE Command Syntax

Notes

Command Syntax Item	Item Information
F entrylen	From 1 to 32767.
V maxlen	From 1 to 32765.

Command Syntax Item	Item Information
`LENGTH dataname2`	Contains length of entry before a STORE or after a RETRIEVE.
`ENTRY OFFSET dataname2`	For ENTRY, contains entry number; for OFFSET, contains offset.
`KEYLEN dataname2 n`	From 1 to 255.
`FROM filename`	Automatic table load from a previously defined file.

Examples

```
TABLE DEPTTB  F 33  ENTRY TBLENT  STATUS TBSTAT
TABLE VARTBL  V 209  LENGTH ENTLEN  OFFSET RBA  STATUS TBSTAT
```

TRANSLATE

```
TRANSLATE dataname [INTO dataname] {FROM  literal TO  literal | FROM ASCII TO EBCDIC
| FROM EBCDIC TO ASCII | FROM UPPER TO LOWER | FROM LOWER TO UPPER}
```

Figure 123 TRANSLATE Command Syntax

Examples

```
TRANSLATE AMOUNT FROM ' ' TO 0

TRANSLATE TYPECODE FROM SPACE TO 9
TRANSLATE AMOUNT INTO EAMOUNT FROM ',' TO '.'
TRANSLATE RECORD FROM ASCII TO EBCDIC
```

TREND

```
TREND nn XNAME dataname YNAME dataname
 [TITLE dataname]
 [XSCATTER [scaling factor] FROM starting value
 BY interval size]  [FOR number of intervals]
 [YSCATTER [scaling factor]3  FROM starting value4
 BY interval size5]  [FOR number of intervals]
```

Figure 124 TREND Command Syntax

Notes

Command Syntax Item	Item Information
`TREND nn`	Unique application number from 1 to 99.
`TITLE dataname`	The length of the data name field cannot exceed 70 characters.

`XSCATTER scaling factor`
`YSCATTER scaling factor`
	Defaults to a unit of 1; otherwise code values from 1 to 4, where:
	1=thousands
	2=millions
	3=billions
	4=trillions

`FROM starting value`	A numeric value with a maximum 14 characters; can be negative. Unit value determined by scaling factor specified for this variable.
`BY interval size`	Positive numeric value with a maximum 14 characters. Unit value determined by scaling factor specified for this variable.
`FOR number of intervals (X)`	An integer between 1 and 30; default is 30.
`FOR number of intervals (Y)`	An integer between 1 and 20; default is 20.

Examples

```
TREND 03  XNAME YEARS  YNAME SALES
     XSCATTER    FROM 1973 BY 1    FOR 12
     YSCATTER 1  FROM 0    BY 100  FOR 10
TREND 04  XNAME YEARS  YNAME SALES
```

TSORT

```
TSORT tablename {ASCENDING | A | DESCENDING | D}
```

Figure 125   TSORT Command Syntax

Examples

```
TSORT ACCOUNTS
TSORT ACCOUNTS D
```

Tn

```
T{1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9}
 [{! alphanumeric literal!... [FOOTING] [FIXED]
 [WITH n AFTER] | + nnn {dataname | literal}}]
```

Figure 126 Tn Command Syntax

Notes

Command Syntax Item	Item Information
WITH n	The default is 1.
+ nnn	n is relative to the starting position of the title line.

Examples

```
T1 'EMPLOYEE REPORT 7' WITH 2 AFTER
T1+5 DYLDATE
T2
T2+20 DIVISION
T3 'EMPLOYEE LIST' FOOTING
T4 'SALARY LIST' FIXED WITH 5 AFTER
T5
T5+20 'ACCOUNT'
```

USE

```
USE {filename | idname} [STATUS dataname]
```

Figure 127 USE Command Syntax, Statement Example 1

Notes

Command Syntax Item	Item Information
STATUS dataname	Required if filename or idname is not sorted.

```
USE {tablename | arrayname} [NORETAIN]
```

Figure 128 USE Command Syntax, Statement Example 2

Examples

```
USE DEPTTB
USE VARTBL NORETAIN
```

VCLOSE

VISION:Vlink only.

```
VCLOSE vfileid
```

Figure 129 VCLOSE Command Syntax

Examples

```
VCLOSE LIBRARY
```

VDELETE

VISION:Vlink only.

```
VDELETE vfileid
```

Figure 130 VDELETE Command Syntax

Examples

```
VDELETE VARFILE
```

VFILE

VISION:Vlink only.

```
VFILE vfileid [LIBNAME ’ddname’ | DDNAME
 dataname]  DISKETTE {nnn | dataname}
 [FILENAME {’filename’ | dataname}] [{EXTENSION |
 EXT} {’extension’ | dataname}]
 [INPUT | OUTPUT {NEW  | REPLACE}] [FIELDS
 !dataname!...]  [VSAMRC dataname]
 [VSAMEC dataname]  [STATUS dataname] [ERROR
 dataname]
 [PATHNAME {!’subdirectory’!... | !dataname!...}]
  or
 [PATHNAMEnn {’subdirectory’ | dataname}]
```

Figure 131 VFILE Command Syntax (Page 1 of 2)

```
[FORMAT {DLM  | TEXT}]  [LIBPASSWORD  {’password’
| dataname}]
[DISKPASSWORD {’password’ | dataname}] [RECFM {F
[nnnnn] | V [nnnnn] | dataname }]
[LENGTH dataname]  [XLATE {’modulename’ |
dataname}] [OPTIONS {ASA | HTAB | ASIS | CTRLZ
| S2R | DROPCRLF | PAD {’char’ | dataname}
| L2U }] [DIRECTORY dataname]  [NEWFILENAME
{’filename’ | dataname }] [{NEWEXTENSION | NEWEXT}
{’extension’ | dataname}]
[ATTRIBUTE {X’hh’ | dataname | SUBDIR | READONLY |
HIDDEN}] [COUNT dataname]
```

Figure 131 VFILE Command Syntax (Page 2 of 2)

VOPEN

VISION:Vlink only.

```
VOPEN vfileid
```

Figure 132 VOPEN Command Syntax

Examples

```
VOPEN VARFILE
```

VREAD

VISION:Vlink only.

```
VREAD vfileid
```

Figure 133 VREAD Command Syntax

Examples

```
VREAD VARFILE
```

VREADDIR

VISION:Vlink only.

```
{VREADDIR | VREADIR} vfileid
```

Figure 134 VREADDIR Command Syntax

Examples

```
VREADDIR VARFILE
```

VRENAME

VISION:Vlink only.

```
VRENAME vfileid
```

Figure 135 VRENAMEE Command Syntax

Examples

```
VRENAME VARFILE
```

VWRITE

VISION:Vlink only.

```
VWRITE vfileid
```

Figure 136 VWRITE Command Syntax

Examples

```
VWRITE VARFILE
```

WORKAREA

```
WORKAREA [NAMED dataname]
```

Figure 137 WORKAREA Command Syntax

Examples

```
WORKAREA
WORKAREA NAMED HOLD
```

WRITE

```
WRITE filename
```

Figure 138 WRITE Command Syntax

Examples

```
WRITE OUTFILE
WRITE ISAMMSTR
WRITE ARFILE
```

WRITEDIR

```
WRITEDIR filename [directory area] [ADD | ALIAS |
CHANGE | REPLACE | SCRATCH]
```

Figure 139 WRITEDIR Command Syntax

Examples

```
WRITEDIR PDSOUT
WRITEDIR PDSOUT DIRAREA
WRITEDIR PDSOUT CHANGE
WRITEDIR PDSOUT DIRAREA ALIAS
```

WRITEMEM

```
WRITEMEM filename
```

Figure 140 WRITEMEM Command Syntax

Examples

```
WRITEMEM MASTOUT
```

XMLFILE

```
XMLFILE ddname [STATUS dataname-s] FIELDS !dataname-f!
```

Figure 141 XMLFILE Command Syntax

Examples

```
XMLFILE XMLOUT FIELDS ACCOUNT NAME ADD1 BALANCE STATUS XMSTAT
```

XMLWRITE

```
XMLFILE ddname
```

Figure 142 XMLWRITE Command Syntax

Examples

```
XMLFILE XMLOUT
```

Chapter 3: DYLINSTL Macro

The DYLINSTL macro establishes the operating environment for the installation of VISION:Results. It contains both mandatory and optional parameters.

Mandatory Parameters

The mandatory parameters establish the operating environment and product being installed.

Default values are provided for all DYLINSTL parameters that will satisfy most installation requirements. However, if any optional parameters are changed, or the installation process will fail, you must specify mandatory parameters, PRODUCT and ENVIRON. If this happens, an Assembler MNOTE error of level 12 is generated with a descriptive message. The return code for the assembly is set to 12, and the macro generation is aborted. If you receive error messages, make the corrections indicated and run the DYLINSTL macro assembly again. The mandatory parameters are:

Parameter	Options	Description
PRODUCT	II	Product being installed is VISION:Results.
	60	Product being installed is VISION:Sixty. For example, PRODUCT=II. **Note:** 60 only should be specified when the VISION:60 loadlib is specified in the JCL. This is not needed when the VISION:Results loadlib is used.
ENVIRON	MVS	Operating system is z/OS or CMS. If the execution-time environment is different from what is specified in this parameter, unpredictable results occur.
	VSE DOSVS	Operating system is any VSE system prior to DOS/VSE SP 2.1.
	VSE2.1 DOS2.1	Operating system is any VSE system that is DOS/VSE SP 2.1 or later. For example, ENVIRON=VSE2.1.

Optional Parameters

The optional parameters change the default values for selected VISION:Results compiler options. Some of the options apply only to specific products while others are valid only on certain operating systems.

Some optional parameters may be overridden at execution time using the OPTION command. For more information about the OPTION command, see the *Advantage VISION:Results for z/OS Reference Guide.*

The optional parameters are validated and applied to the requested operating system environment. If the values are not accepted, the installation may not automatically terminate. An Assembler MNOTE error level of 4 or 8 is generated with a descriptive message:

- Level 4 indicates that the specified parameter, in the message text, is not available for the product being installed; the parameter is ignored.
- Level 8 indicates that the specified parameter has an invalid or null value, or is invalid for the specified operating system. The message text indicates what action is taken (ignore, truncate, assign a default value, and so on).

If you receive error messages, correct the errors and repeat the macro assembly.

Following are descriptions of the optional parameters. Column heading Prod/Env indicates the combination of products and operating systems in which the parameter is valid.

Prod	Description
A	Valid for all values of the PRODUCT keyword.
II	Valid for VISION:Results.
60	Valid for VISION:Sixty.

Env	Description
A	Valid for all values of the ENVIRON keyword.
O	Valid for z/OS and CMS only.
D	Valid for all releases of DOS or DOS/VSE, prior to DOS/VSE SP 2.1.
M	Valid for VSE/ESA at DOS/VSE SP r2.1 and later only.

Parameter	Prod / Env	Options	Description
ASALINE	II/ O	Y	Omits the extra line that is generated at the beginning of a REPORT when the ASA keyword is coded on the REPORT statement. The default is N. For example, ASALINE=Y.
BATCHIQ	II/ O	name	Specifies an alternate load module name (maximum of eight characters) for the IQ Batch – Results bridge. The default is IIBATCH. For example, BATCHIQ=MYNAME.
CATPLAN	II/ O	name	For COPYDB2 only. Provides a replacement name (maximum of eight characters) for the DB2 plan name in the DB2 catalog. The default is DYLDB2. For example, CATPLAN=MYPLAN. This parameter was originally called D2PLNID and is still valid under that name. However, whichever name is used, CATPLAN will print on the DEFAULT OPTIONS SELECTED page of the VISION:Results output.
CATSYS	II/ O	name	For COPYDB2 only. Provides a replacement name (maximum of eight characters) for the DB2 subsystem ID in the DB2 catalog. The default is DB2A. For example, CATSYS=DB2T. This parameter was originally called D2SYSID and is still valid under that name. However, whichever name is used, CATSYS will print on the DEFAULT OPTIONS SELECTED page of the VISION:Results output.

Parameter	Prod / Env	Options	Description
CBXSIGN	II/ O	Y	Allows the compiler to process COBOL data definitions containing the SIGN IS SEPARATE clause by increasing the size from the PIC clause by 1 to accommodate the sign, and storing the definition as a CH (character) field. No arithmetic operations are allowed. The default is N, which continues to allocate the size defined in the PIC clause and not account for the additional byte for the external size. For example, CBXSIGN=Y.
CDLOAD	A/ M	Y	Instructs the compiler to use the CDLOAD facility, rather than LOAD, for user programs referenced in a CALL statement. This allows the SIZE information on the CALL to be omitted. OPTION NOCDLOAD, or specifying NOCDLOAD in the CALL statement, overrides this parameter. The default is N. For example, CDLOAD=Y. ***WARNING!*** *If a called subroutine uses storage beyond the end of the CSECT and relies on the compiler to reserve space using the size (nK) on the CALL, do not load that subroutine with CDLOAD. The VISXLOAD table and the macros for updating it, VISXLOAD and VISXEND, are provided to contain the names of any subroutines that should not use CDLOAD.* For more information about the CALL Command, see the *Advantage VISION:Results for z/OS Reference Guide*.

Parameter	Prod / Env	Options	Description
CENTNEW	A/ A	nn	For Advantage™ VISION:Excel™subroutines DYLBETDT, DYLDATE, DYLFMTJG, and DYLSELDT only. Determines the century prefix for a 2-digit date field. It is also used by VSE systems (prior to 2.1) for determining the run-date century. If the 2-digit date field is less than CENTNEW, a value of 20 is used for the century prefix in 4-digit year calculations. Otherwise, 19 is used. For VSE systems prior to 2.1, this parameter sets the run-date century in DYLDATE4, DYLGREG4, DYLJULIAN4, and DYLDATEPG4. The default is 75. For example, CENTNEW=70 (2-digit date fields between 00 and 69 are prefixed with 20; for example, 2069).
CENTRY1	A/ A	nn	For VISION:Excel subroutines DYLBETDT, DYLDATE, DYLFMTJG, and DYLSELDT that deal with one or two dates as input. Provides a century prefix for the first (or only) input date. Values can be any numeric value from 00 to 99. The default is blank (use CENTNEW). Override at execution time by moving a value to the DYLCENTRY1 reserved word. For example, CENTRY1=19.
CENTRY2	A/ A	nn	For VISION:Excel subroutines DYLBETDT, DYLDATE, DYLFMTJG, and DYLSELDT that deal with two dates as input. Provides a century prefix to be used for the second input date. Values can be any numeric value from 00 to 99. The default is blank (use CENTNEW). Override at execution time by moving a value to the DYLCENTRY2 reserved word. For example, CENTRY2=20.
COBAPOS	II/ A	N	Specifies how two consecutive apostrophes coded on a COBOL VALUE clause are stored. Y stores them as a single apostrophe and N stores them as two apostrophes. This can be overridden using OPTION COBAPOS or NOCOBAPOS. The default is Y. For example, COBAPOS=N.

Parameter	Prod / Env	Options	Description
COBEDIT	II/ A	-, B, E, F, G, H, J, K, L, Q, U, W, X, Y, Z	Changes the default edit code for numeric fields defined in COBOL data definitions from A to some other value. The valid options are listed. To request a COBOL edit code of (blank), enter a minus sign (-) as the COBEDIT option. The default is A. For example, COBEDIT=E.
COBENV	A/ O	Y	Establishes the COBOL II or LE/370 run-time environment for each VISION:Results request, treating all called COBOL programs as subroutines. This parameter preserves the environment whether or not the SORT command is used in the request. The default is N. For example, COBENV=Y. If COBENV=Y, then the DYLINSTL parameter LE must be set to N. The parameters must be coded in the order: LE=N, COBENV=Y. If they are not in this order, COBENV=Y will be ignored by the DYLINSTL macro assembly job (CUSTMJCL).
COB2NR	A/ O	Y	Allows proper execution when calling COBOL II subroutines compiled with the NORES option. Because the COBOL run-time library is not used in a NORES application, specifying COB2NR overrides the COBENV automatic environment option. Specifying COB2NR on the OPTION statement will override a DYLINSTL-specified COBENV for a given program. The default is N. For example, COB2NR=Y.
COMPERR	A/ O,M	Y	If there is a compiler error, overrides the USER 4 ABEND (AB=0004) with a RETURN CODE 4 (RC=0004) instead. The default is N. For example, COMPERR=Y.

Parameter	Prod / Env	Options	Description
COMPWRK	A/ O	nnnn	Changes the default compiler work area size. The value specified is a decimal unit of 1024 bytes (1 KB) and cannot be less than 16 nor more than 9999. Code only the decimal number; do not code the K. The default is 1500 KB. An insufficient size can result in a DYL-012E error message. (See *Advantage VISION:Results for z/OS Messages and Codes*.) For example, COMPWRK=2000.
CONDBUF	II/ D,M	nnnn	Increases the default of 60 KB of memory for the COPYC (CONDOR) interface and I/O buffers. The value specified is a decimal unit of 1024 bytes (1 KB) and cannot be more than 9999. Code only the decimal number; do not code the K. For example, CONDBUF=999.
CPYSBLB	II/ D,M	C	Changes the default member type used in the COPY command from D to C. For example, CPYSBLB=C.
CURNCY	A/ A	a, C’a’, X’nn’,	Specifies the symbol used with the floating currency edit codes (F, G, K, H, J, L). The value can be expressed as a valid alphanumeric character (a), as a character notation of a valid alphanumeric (C’a’), or as a hexadecimal expression (X’nn’). The default symbol is a dollar sign ($). Override at execution time using OPTION keyword CURRENCY. For example, CURNCY = ?.
DATATRN	A/ A	N	Disables or modifies the translation of data by the file print and error analysis functions. (VISION:Results and VISION:Sixty translate everything except letters, numbers, and some special characters, to spaces, which improves printer speed and readability of listings.) The main reason for disabling translation is to allow lowercase to print. The default is Y. For example, DATATRN=N.

Parameter	Prod / Env	Options	Description
DB2DEC9	II/ O	Y	For COPYDB2 only. Allows numeric fields produced by COPYDB2 statements to contain more than nine decimal digits, without generating error messages, by defining the data as character. Additional scaling or conversion must be done by the user for this data to be used in an arithmetic operation. The default is N. For example, DB2DEC9=Y.
DB2ERR	II/ O	N	For VISION:Results Interface to DB2 (r4.0 or higher) only. Tells VISION:Results Interface to DB2 to check for SQLCODE errors. If it finds an error, then VISION:Results Interface issues an error message, executes a rollback, and quits the program. The default is Y. To override, use the EXEC SQL WHENEVER statement. For example, DB2ERR=N.
DB2NULL	II/ O	Y	For COPYDB2 only. Allows null indicator fields to be generated when used with COPYDB2. If a field can have a null value, then a 2-byte binary field with the same name plus a suffix of _IND follows. If the field's name is so large that the suffix cannot be added, then the null indicator field is not generated. The default is N. For example, DB2NULL=Y.
DB2PLAN	II/ O	name	For VISION:Results Interface to DB2—Dynamic (r3.5 or higher) only. Provides a replacement name (maximum of eight characters) for the DB2 plan name to be used for accessing the DB2 database using dynamic SQL. The default is DYLDB2. For example, DB2PLAN=TSTDB2. This replaces obsolete parameter D2PLNID.

Parameter	Prod / Env	Options	Description
DB2SNGL	II/ O	Y	For VISION:Results Interface to DB2—Dynamic (r4.0 or higher) only. Tells VISION:Results Interface to DB2 to check whether the embedded SELECT SQL statement retrieved more than one row. If it did, then VISION:Results quits with the following error message: DYLD021I (DYLSQLCODE = -9021) - MULTIPLE ROWS FOR EMBEDDED SELECT The default is N. For example, DB2SNGL=Y.
DB2SYS	II/ O	name	For VISION:Results Interface to DB2—Dynamic (r3.5 or higher) only. Provides a replacement name (maximum of eight characters) for the DB2 subsystem ID to be accessed using dynamic SQL. The default is DB2A. For example, DB2SYS=DB2T.
DECIML9	II/ A	W	Forces nine decimal places and replaces the error message with a warning when there are more than nine digits to the right of the decimal point in a COBOL copybook field definition or in a field following the $COBOL keyword.
		S	Same processing result. Suppresses the resultant warning message. The default is E. For example, DECIML9=S.

Parameter	Prod / Env	Options	Description
DELIM	II/ A	X'nn' C'c'	Formats the print lines (with no carriage control, titles, or column headings, and with the DELIM character separating the columns) and generates the output to a data set rather than a printer. The data set can then be downloaded to a PC. Any character except blank (C'' or X'40') is valid. There is no default. For example, DELIM=C'$'. Override at execution time using REPORT keyword DELIM. For more information about using report statements, see the *Advantage VISION:Results for z/OS Reference Guide*.
DLMFRST	A/ A	Y	Allows the leading delimit character in the first printable position of a report file when using REPORT DELIM. The default is N, which suppresses the leading character in the first printable position of a report file when using REPORT DELIM. For example, DLMFRST=Y.
DUPCBNM	II/ A	Y	Allows duplicate data names for COBOL data definitions. Duplicate data names are allowed, but they cannot be referenced. The default is N. For example, DUPCBNM=Y.
DYL4YR	A/ A	Y	Prints a 4-digit year date on SYSPRINT or SYSLST for the program listing, and any PRINT, HEXPRINT, and so on, statements. It also sets the 4-digit year run date (DYLDATE4) in the report headings for OPTION USERDEFAULT. The default is N. For example, DYL4YR=Y. Override at execution time using OPTION keywords DYL4YEAR and NODYL4YEAR.

Parameter	Prod / Env	Options	Description
DYLVARP	II/ A	Y	Allows the use of data names prefixed with DYL. Required for VISION:Results Interface to DB2 and VISION:Results Interface to SQL/DS only. The default is N. For example, DYLVARP=Y.
EDALIGN	A/A	Y	Determines alignment of numeric fields formatted on a standard report print line. If EDALIGN is set to Y, then multiple list statements that have the same numeric field, but with different edit codes, will use the same right-justified alignment. The alignment is based on the first LIST statement. When the BETWEEN parameter on a REPORT statement either is not coded or coded with the value of 5, the alignment of numeric fields is right-justified and adjusted to fit between the previous and subsequent columns. When the BETWEEN parameter is coded with a value other than 5, the edit code alignments for numeric fields use approximately the same alignment that the default edit code (P edit code format) uses when EDALIGN=N. The value of the BETWEEN parameter should be adjusted according to the numeric field size to prevent overlapping or truncation between other report line columns. The EDALIGN parameter will not be in effect when Fixed Print Positioning (dataname AT ...) is used. When EDALIGN=N, the alignment for the edit codes are the same as they were on prior releases. The default of EDALIGN is N. For example, EDALIGN=Y.
EDP1ZERO	A/ O	Y	Modifies the edited output of edit code P for fields containing the value 0. The default is N, which produces a string of 0s. The setting EDP1ZERO=Y produces a single 0 for zero fields and also suppresses leading zeroes on non-zero fields. For example, EDP1ZERO=Y.

Parameter	Prod / Env	Options	Description
EDSUPR	A/ O	Y	Suppresses the blank space appearing immediately before the decimal point for fields that contain a value less than 1 and are edited or listed using edit codes Y, F, G, K, X, H, J, L, U, or Q. The default is N. For example, EDSUPR=Y.
EURODAT	A/ A	Y	Changes the date default in all listings to European format (DD/MM/YY). The default is N. For example, EURODAT=Y.
EURONUM	A/ A	Y	Changes the editing for numeric fields to print in European format (1.000.436,33) rather than U.S. (1,000,436.33). The default is N. For example, EURONUM=Y.
EXCEL	II/ A	Y	Changes the default to OPTION EXCEL for VISION:Excel users. The default is N. For example, EXCEL=Y.
EXCLPAT	II/ A	Y	For VISION:Excel product. Instructs the compiler to print the VISION:Excel patch report for each request in the VISION:Excel jobs in the AUDPRINT dataset. The default is N, which suppresses the report. Override this parameter using the OPTION keywords EXCELPAT | NOEXCELPAT. For example, EXCLPAT=Y.
EXPRERR	A/ A	Y	Prints the address of the first byte of each line of a FILE or REFORMAT area in hexadecimal in the left margin of the error analysis output. The offset (the last byte of each line) prints in decimal and hexadecimal in the right margin. The default is N. Override at execution time using OPTION keywords EXPRTERR or NOEXPRTERR. For example, EXPRERR=Y.

Parameter	Prod / Env	Options	Description
EXTEND	A/ D, M	N	Determines if reserved words DYLSYSDATE and DYLSYSPARM will be populated from the VSE communications region. Default is Y, which populates these fields. EXTEND should be set to N if called subroutines are used to access DL/I databases. For example, EXTEND=N (fields are blank).

Parameter	Prod / Env	Options	Description
FREEMEM	II/ O	nnnn	Changes the amount of memory returned to the system during the compile phase. VISION:Results does a variable GETMAIN to obtain storage, and then returns the FREEMEM amount to the operating system for its use. The FREEMEM default value of 1000 KB will meet the needs of 99 percent of programs. Only in cases of extremely large programs is the default amount not enough. The value specified is a decimal unit of 1024 bytes (1 KB) of memory to be returned to the operating system. Code only the decimal number, with a maximum of 4 digits. The minimum value for FREEMEN is 6 and its maximum value is 9993 KB (equivalent to the maximum of GETMAX value minus 6). The default value is 1000 KB. **Note:** The default for GETMAIN storage can range from a minimum of 340 KB to a maximum of 9999 KB. The DYLINSTL parameters, COMPWRK and GETMAX, determine the GETMAIN storage that is obtained. The FREEMEM default value of 1000 KB is based on, and synchronized with, the default values of COMPWRK and GETMAX. If you decrease the value of COMPWRK and GETMAX from their defaults, you may have to decrease the value of FREEMEM accordingly. However, an increase of COMPWRK and GETMAX does not require an increase to FREEMEM unless the increase to COMPWRK and GETMAX is 4 times or higher than their defaults. This would be required for extremely large programs. VISION:Results does not ensure that it receives this amount of memory before issuing the FREEMEM. If you are running in too small a region, SA0A or SA78 freemain abends may result. For example, FREEMEM=8.

Parameter	Prod / Env	Options	Description
FREEZDD	A/ O	name	Replaces the default ddname of SYS280FZ (VISION:Results) and SYS260FZ (VISION:Sixty) with a maximum of eight alphanumeric characters. For example, FREEZDD=MYFREEZE. The FREEZE ddname is always overridden by the FREEZDD parameter of the OPTION statement when you freeze a program.
GETMAX	II/ O	nnnn	Changes the default maximum GETMAIN value for the compile phase. The value requested is a decimal unit of 1024 bytes (1 KB), replacing the 2500 KB default. Code only the decimal number, a value between 340 and 9999. An insufficient size can result in a DYL-997E error message. (See *Advantage VISION:Results for z/OS Messages and Codes*.) For example, GETMAX=4000.
KWDLT	II/ O	kw1... kwnn	Specifies up to 20 (kw1 to kw20) reserved words that should not be considered keywords by VISION:Results. Allows use of existing programs containing data names that are not the same as the newly introduced keywords and may cause violation errors. The exempted words are itemized in the OPTIONS summary report. If OPTION $$KEYPLEVEL is coded in your program, DYL-12171 informational messages listing the disabled keywords are printed when parsing the first FILE statement. Omitting the KWDLT parameter keeps all distributed keywords in effect. The default is N. For example, KWDLT=(VOPEN,VCLOSE,VFILE)

Parameter	Prod / Env	Options	Description
LE	A/ A	N	You can call subroutines written in any LE-compliant language, including COBOL, C, PL/I, FORTRAN, and Assembler. This includes IBM Language Environment (LE) Service Routines such as CEEDATE and CEEDAYS. This overrides the DYLINSTL parameter COBENV=Y. A warning message is produced if both are specified. The default is Y. For example, LE=N. Override at execution time using OPTION keywords LE | NOLE. Programs frozen under releases prior to VISION:Results r5.0 must be refrozen using r5.0 or later for this parameter to take effect.
LIBDLBL	II/ D,M	ddddddd	Allows you to specify a value for the DLBL name of the AllFusion® CA-Librarian® library. The default is MASTER. For example, LIBDLBL=MASTER2.
LIBSYS	II/ D,M	nnn	Allows you to specify a value for the system logical unit number for the CA-Librarian library. The default is 026. For example, LIBSYS=100.
LIBRBUF	II/ A	nnnn	Increases the default of 60 KB of memory for the COPYL (CA-Librarian) interface and I/O buffers. Specify the value as a decimal unit of 1024 bytes (1 KB) between 61 and 9999. Code only the decimal number; do not code the K. For example, LIBRBUF=999.
LPPUNLMT	A/ O	Y	Creates a report with unlimited lines per page. The specification REPORT 999 LONG in combination with LPPUNLMT=Y, eliminates report page breaks. The parameter produces a report with titles and column headings only on the first page of the report. The default is N (allows report page breaks). For example, LPPUNLMT=Y.

Parameter	Prod / Env	Options	Description
LSTSTMX	II/ A	Y	Overrides the maximum of 26 fields in a LIST statement, allowing a maximum of 40 fields. This may relieve DYL-300E error situations. The default is N. For example, LSTSTMX=Y.
LTRFROM	II/ A	Y	Causes the LETTER command to use the same line number for the TO clause as the FROM clause if the TO clause is not specified. For example, LETTER 1 FROM 2 executes the same as LETTER 1 FROM 2 TO 2. The default is N. For example, LTRFROM=Y.
LTRZERO	II/ A	Y	For letter writing only. Changes A and B edit codes to print zero when the value is zero and defined with no digits to the right of the assumed decimal point. The default is N. For example, LTRZERO=Y.
MACHCOR	II/ D,M	nn	Increases the memory size for a user-written MATCH/MERGE routine. This is necessary if the routine is greater than 4 KB (default). Specifies the value decimal format, between 4 and 99, and represents a request for nnK. For example, MACHCOR=10.
MACHORG	II/ A	Y	Sets MATCH processing to perform as in releases prior to r2.5 of VISION:Results. The default is N. For example, MACHORG=Y.
MAXDNLN	II/ A	nn	Changes the maximum data name length from the system default of 50 characters. Specify any decimal value between 10 and 50, inclusive. If data names containing more than 42 characters are included in programs, the area and location sections are not printed on the Cross Reference listing when the program requests XREFA or XREF. For example, MAXDNLN=40.

Parameter	Prod / Env	Options	Description
MAXDYLF	II/ A	nnnn	This parameter is not used for applications run with VISION:Results r4.0 and later. In releases prior to 4.0, this specified the maximum number of DYLIOU and PICNSAVE files in an application. The compiler reserved enough storage for this number of files when restoring a frozen application. If this parameter was specified in prior releases, you may need to retain the same specification to ensure compatibility with frozen applications. For example, MAXDYLF=40.
MNAMENU	II/ A	Y	Allows called subroutine names to start with a numeric instead of an alphabetic character. The default is N. For example, MNAMENU=Y.
MRWRK6	A/ A	xxxxxxxx	For VISION:Sixty or OPTION 260 only. Changes the default SYSWRK DDNAME. The SYSWRK file is used for temporary report processing for a VISION:Sixty multiple report. The DDNAME can not be more than eight alphanumeric characters. This can be overridden on the M parameter statement in column 8. For example, MRWRK6=WORKFILE.
NAMEHDR	A/ A	’text’	Text enclosed within single quotation marks will print at the top of pages produced on SYSPRINT or SYSLST. Maximum of 60 characters. The default is null. For example, NAMEHDR=’COMPANY NAME’.

Parameter	Prod / Env	Options	Description
NDVRCOM	II/ O	Y	Prints the attributes of a copied AllFusion® Endevor® Change Manager element as comments immediately following the COPYN statement in VISION:Results program listing. The default is N (the attributes will not be printed). For example, NDVRCOM=Y. Override at execution time using OPTION keywords NDVRCOM | NONDVRCOM.
NDVRENV	II/ O	cccccccc	An 8-byte character string identifying the Endevor Change Manager environment name to use as a default with the COPYN command. If the environment name is specified on the OPTION NDVRENV statement, it will override the DYLINSTL value. The default is blanks (no value has been supplied). For example, NDVRENV=yourname.
NODLETE	II/ O	Y	Prevents program aborts with an SC03 ABEND at end of job. These program aborts occur when a user calls a subroutine that opens a file but fails to close the file before VISION:Results terminates. During job termination, VISION:Results deletes all of the called modules that it has loaded. Then the operating system, as part of its job termination, attempts to close all unclosed files but is unable to find the DCB in the deleted module, causing an SC03 ABEND. The default is N. If you specify this parameter, VISION:Results will not delete loaded modules at end of job. This may be a problem for TSO/ISPF users using VISION:Results interactively (such as with Advantage™ VISION:Online® for TSO). They may quickly fill their TSO region with undeleted load modules and consequently run out of memory space. For example, NODLETE=Y.
NOPOWRT	II/ O	Y	Disallows writing to a PDS file. The default is N. For example, NOPOWRT=Y.

Parameter	Prod / Env	Options	Description
NOSRTAB	A/ O	Y	Sets the default for the system to generate a return code of 8 rather than 16 to terminate SORT before all the data has been processed. This can happen if your program logic issues a STOP or QUIT before SORT has processed all records (UNTIL data name not E). In some installations, returning a value of 16 to SORT causes an ABEND. The default is N. For example, NOSRTAB=Y. Override at execution time using OPTION keyword NOSORTAB.
NOTOTAL	II/ A	Y	Suppresses the file statistics and control total information on the control totals page at the completion of program execution. The default is N. For example, NOTOTAL=Y. Override at execution time using OPTION keyword TOTAL.
NOVSIO	II/ A	Y	Disallows I/O (update) to VSAM KSDS, ESDS, and RRDS files. The default is N. For example, NOVSIO=Y.
NOVSOIO	II/ A	Y	Disallows output and I/O on VSAM KSDS, ESDS, and RRDS files. The default is N. For example, NOVSOIO=Y.
NUMCHAR	A/ A	Y	Changes the IF NUMERIC test to check for only F0 through F9 in all bytes (that is, do not allow a sign in the low order digit) for NU or CH fields. For PD fields, checks the sign field for x’C’. The default is N. Override at execution time using OPTION keywords NUMCHAR | NONUMCHAR. For example, NUMCHAR=Y.
NUMPD	A/ A	Y	Allows you to use the IF NUMERIC test with PD fields. The default is N. For example, NUMPD=Y. Override at execution time using OPTION keywords NUMPD | NONUMPD.

Parameter	Prod / Env	Options	Description
OPTLIST	A/ A	N	Changes the default to suppress printing of the DEFAULT OPTIONS SPECIFIED page (formerly OPTIONS ACTIVE) at the end of the VISION:Results listing. The default is Y. For example, OPTLIST=N. The DEFAULT OPTIONS SPECIFIED page should be available when consulting CA Customer Support and must accompany any documentation requested by them. Override at execution time using OPTION keywords OPTLIST | NOOPTLIST.
OPTPRDG	II/ A	N	Changes the default from OPTION PRINTDIGITS to OPTION NOPRINTDIGITS. This option affects only LIST statements. N will override the default which allows printing of very large numbers (21 digits and 9 decimal places). Setting OPTPRDG to N forces the output to format numbers within the limitations of VISION:Results r2.2 and earlier (10 digits and 5 decimal places). The default is Y. For example, OPTPRDG=N. Override at execution time using OPTION keywords PRINTDIGITS | NOPRINTDIGITS.
OPTPRER	II/ A	Y	Defaults to extended error analysis. This has the same effect as using OPTION PRINTERR | PRINTERROR. The default is N. For example, OPTPRER=Y.
OUTFILE	II/ A	Y	Produces a null file when a file defined as an output file does not write any records. This allows subsequent steps referring to this file to proceed normally. The default is N. For example, OUTFILE=Y.
PANDEV	II/ D,M	dddd	Allows you to specify a value for the DASD type of the AllFusion® CA-Panvalet® library. The default is 3330. For example, PANDEV=1000.

Parameter	Prod / Env	Options	Description
PANSYS	II/ D,M	nnn	Allows you to specify a value for the System Logical Unit number for the CA-Panvalet library. The default is 026. For example, PANSYS=100.
PANVBUF	II/ A	nnnn	Increases the default of 60 KB of memory for the COPYP (CA-Panvalet) interface and I/O buffers. Specify the value as a decimal unit of 1024 bytes (1 KB) between 60 and 9999. Code only the decimal number; do not code the K. For example, PANVBUF=999.
PDSREPL	II/ O	Y	Causes the PDS WRITEDIR with REPLACE function to place a value of A in the status byte if the REPLACE results in an ADD because the member did not pre-exist. The default is to return a value of N. For example, PDSREPL=Y. Override at execution time using the OPTION keywords PDSREPA | PDSREPN.
PGLINER	A/ A	nn	Alters lines per page on the report file: ■ VSE file name SYSLST ■ z/OS ddname SYS280R Default value is 55; specify a decimal value less than 100. For VSE only, if this value is greater than the operating system default for the system printer, make sure that the FCB is greater or equal to the PGLINER value. For example, PGLINER=52.
PGLINES	A/ A	nn	Alters lines per page on the listing file: ■ VSE file name SYSLST ■ z/OS ddname SYSPRINT Default value is 55; specify a decimal value from 10-100. For example, PGLINES=52.

Parameter	Prod / Env	Options	Description
PROGMOD	II/ A		Changes programming mode and cross-reference option, and disallows exponentiation. Specify multiple options by coding a list separated by commas, enclosed in parentheses. You can specify the items in any order, but they must be unique. If you choose only one item, do not include parentheses. The default is XREFREF,CONVENTIONAL,EXP. Choices for each item are described by group.
			Programming Mode
		STRUCT	This option group defines the default programming mode. See the *Advantage VISION:Results for z/OS Reference Guide*. CONVENTIONAL is the default and disallows the use of structured commands. STRUCTURED allows the use of structured commands and nested IF statements. Disallows GOTO, ACCEPT, and REJECT commands.
		STRUCT2	STRUCTURED2 allows the use of structured commands and nested IF statements. Also allows the use of GOTO, ACCEPT, and REJECT commands.
		USERDEF	USERDEFAULT disallows the use of structured commands and some conventional report facilities.

Parameter	Prod / Env	Options	Description
			Cross-Reference Option
		XREFREF	System-released default value. Produce cross-reference for referenced data names. This value is retained if this item is not specified.
		XREFA	Request cross-reference for all data names.
		NOXREF	Do not produce cross-reference.
			Exponentiation Exponentiation is the default.
		NOEXP	Do not allow exponentiation. For example, PROGMOD=NOEXP. For example, PROGMOD=(STRUCT,XREFA). For example, PROGMOD=(STRUCT,NOEXP).
PRTCTRS	II/O	Y	Instructs the compiler to print the values and labels for all 126 counters at the end of the execution for each request even if the value is zero. The default is N, which does not print the counter values and labels. For example, PRTCTRS=Y. Override at execution time using OPTION keywords PRINTCTRS | NOPRINTCTRS.
PRTERR6	A/A	1 2 3	For VISION:Sixty or OPTION 260 only. Changes the default for print error analysis (C parameter, column 29) to: 1 = extended error analysis 2 = data name error analysis 3 = extended and data name error analysis If omitted, the default STANDARD (blank) is used. For example, PRTERR6=3.
PRTZERO	A/A	Y	Changes the A and B edit codes to print 0 for zero values. The default is N. For example, PRTZERO=Y.

Parameter	Prod / Env	Options	Description
QLF	II/ A	Y	Sets the system to have the data name qualification option to allow non-unique data names. Use this parameter with caution. The default is N, which forces unique names. For example, QLF=Y. Override at execution time using OPTION keywords NOQLF | QLF.
RANDMPCT	A/ A	Y	Determines which random number generator algorithm is used. RANDMPCT=N/Y. If set to Y, will use for random sampling a random number generator algorithm that generates uniform random numbers according to the method of Fishman and Moore (1982), using a prime modulus multiplicative generator with modulus 2**31 and multiplier 397204094. The default is N. For example, RANDMPCT=Y.
RDYONLY	II/ A	Y	Creates a read-only version of the product that does not allow references to output data sets. LIST is allowed, but commands such as FILE or WRITE are inhibited. The default is N. For example, RDYONLY=Y.
RESRWRD	II/ A	Y	Allows VISION:Results reserved words (commands, keywords, or both) as data names. The default is N. For example, RESRWRD=Y. **Note:** Setting this parameter to Y disables many VISION:Results features such as Letter Writer, PDS Update, and Table Handling. To override individual keywords, use optional parameter KWDLT.

Parameter	Prod / Env	Options	Description
RETCODE	A/ A	N	Determines if the return code from a called subroutine is available to the programmer in reserved word DYLCOMRG. The default is Y. For example, RETCODE=N. When this parameter is set to Y, the default, include the following program logic to capture the return code: WORKAREA XX 2 REDEFINE XX YY 2 BI CALL subroutine USING parameters MOVE DYLCOMRG TO XX IF YY NE 0 (to handle non-zero return-code) ENDIF
RPTASA	A/ O	Y	Instructs the report writer feature to produce all reports with ASA carriage control characters, instead of machine carriage control. The default is N. For example, RPTASA=Y. REPORT ASA keyword overrides this parameter.
RPTDDNM	A/ O	ccccccc	Supplies an alternative to SYS280R for the JCL DDname of the report file produced by LIST statements. If specified, the value must be a maximum of 7 alphanumeric characters. It cannot be a specified reserved word. The default is SYS280R. For example, RPTDDNM=OVTDATA. When the Report SYS280Rx syntax is used, the DDname supplied in the JCL must be cccccccx (the same name as supplied in this parameter).
RPTXPAG	A/ O	Y	Allows the report writer feature to print a blank page at the beginning of a report. The default is N, which suppresses the blank page. For example, RPTXPAG=Y.

Parameter	Prod / Env	Options	Description
R15RC	A/ M	Y	Loads the return code into register 15 at end of job so the return code can be tested in conditional JCL statements. The default is N. For example, R15RC=Y.
SORTDEV	II/ O	name	Changes the dynamic allocation feature of SORT to specify a unit name other than SYSDA. Specify any valid unit name known to the system, using up to eight characters. The default is SYSDA. For example, SORTDEV=3380. This parameter is valid only when parameter SORTDYN is also specified.
SORTDYN	II/ O	Y	Appends DYNALLOC=SYSDA to the SORT control statement passed to SORT during execution, dynamically allocating disk work areas in some SORT packages. Verify that the SORT installed on your system supports this feature before choosing this option. The default is N. For example, SORTDYN=Y. Override at execution time by supplying JCL for SORT work areas.

Parameter	Prod / Env	Options	Description
SORTMEM	A/ O	(nnnn, nnnn, nnnn, nnnn)	Alters DYLSORT sort memory minimum, maximum, core, and maximum reset size defaults. Currently, these values are: 36 KB MINIMUM 100 KB MAXIMUM 700 KB CORE=MAX 100 KB USED TO RESET MAXIMUM Alter these default values by creating a list of positional values of decimal units of 1024 bytes (1 KB) enclosed in parentheses and separated by commas. Code only the decimal number (no K). The positional values are in the order: MINIMUM, MAXIMUM, CORE=MAX, RESET MAXIMUM. To keep any default unchanged, omit the corresponding value in the list, while keeping commas as placeholders to indicate omitted values. If the only value you change is the MINIMUM, enter it without parentheses. You can omit commas in drop off order. Examples: SORTMEM=(,156) Change MAXIMUM to 156K SORTMEM=(40,,800) Change MINIMUM to 40K, CORE=MAX to 800K SORTMEM=44 Change MINIMUM to 44K
SORTNAM	A/ A	name	Changes the program name called by DYLSORT. The name is any valid program name up to eight characters long that is known to the system. The default is SORT. For example, SORTNAM=IGHCRO00.
SQLIFIF	II/ A	Y	Allows the use of nested IF statements without STRUCTURED or STRUCTURED2 in effect. If STRUCTURED or STRUCTURED2 is not being used, the VISION:Results Interface to DB2 requires SQLIFIF. The default is N. For example, SQLIFIF=Y.

Parameter	Prod / Env	Options	Description
SSMASK	A/ A	C	Changes the Social Security default print edit mask to Canadian format for social insurance numbers. The default is N. For example, SSMASK=C.
STATPLN	II/ O	name	For VISION:Results Interface to DB2—Static (r4.0 or higher) only. Provides a replacement name (maximum of eight characters) for the DB2 plan name to be accessed using Static SQL. The default is STATDB2. For example, STATPLN=DB2TEST.
STATSYS	II/ O	nnnn	For VISION:Results Interface to DB2—Static (r4.0 or higher) only. Provides a replacement name (maximum of eight characters) for the DB2 subsystem ID to be accessed using Static SQL. The default is DB2A. For example, STATSYS=DB2T.
STRUCGO	II/ A	Y	Provides GOTO capability in structured mode. Use of this parameter is not recommended because the use of GOTO in a structured program can lead to confusion. The default is N. For example, STRUCGO=Y. Setting this parameter to Y allows structured programming to be used with VISION:Results Interface to DB2 and VISION:Results Interface to SQL/DS (VSE and CMS).
SUBRADD	A/ M,O	N	Specifies whether calls to user-written modules may reside above the 16-MB line. The default is Y to support AMODE(31) subroutines. CMS systems must specify N. For example, SUBRADD=N.

Parameter	Prod / Env	Options	Description
SUPCOBW	II O	N	Allows the compiler to print the DYL-188W and DYL-876W messages when processing COBOL data definitions. The default is Y to prevent the messages from printing. For example, SUPCOBW=N.
SUPRESQ	II/ A	Y	For VISION:Results Interface to DB2 or VISION:Results Interface to SQL/DS only. Suppresses the Q%% variables from the XREF listing. The default is N. For example, SUPRESQ=Y.
SUP182W	II/ A	Y	Suppresses the DYL-182W message issued mainly when a level 88 item is encountered by the COBOL copy facility. The default is N. For example, SUP182W=Y.
SUP452E	II/ D,M	nn	Eliminates the error message DYL-452E: COPY MEMBER NOT FOUND (CONDOR LIBRARY). The value entered in decimal units of 1024 bytes (1 KB) represents the buffer size for CONDOR library. The buffer size supplied to CONDOR must be at least 7 KB greater than the largest block size of any CONDOR library being accessed. The default is 8 KB. nn is the largest block size + 7 KB. Specify values between 8 KB and 99 KB. For example, SUP452E=27 for a 20 KB buffer.
SYSBLOK	A/ O	Y	Allows the operating system to calculate the appropriate block size of a non-VSAM output file created by VISION:Results. This parameter is ignored if the block size is specified on the FILE statement or the DD JCL statement associated with the file, or the BY parameter of the PICNSAVE statement. The default is N. For example, SYSBLOK=Y. Override at execution time using OPTION keywords SYSBLOCK | NOSYSBLOCK.

Parameter	Prod / Env	Options	Description
TAPENO#	II/ D,M	Y	Allows the SYS number to be omitted on tape FILE statements. The default is N. For example, TAPENO#=Y.
TIMESEP	A/ A	C	For VISION:Results—Changes the default symbol used to separate HH, MM, and SS when listing DYLETIME. The default is D (period) and can be changed to C (colon). For example, TIMESEP=C (DYLETIME lists as HH:MM:SS). For VISION:Sixty or OPTION 260—Modifies the USTE field accordingly. For example, TIMESEP=C (USTE field has HH:MM:SS format). Override at execution time using OPTION keywords TIMECOLON | TIMEDOT.
VDUPABND	II/ A	Y	Causes VISION:Results to terminate processing with a u300 abend when a VSAM duplicate key condition occurs. The default is N (duplicate key condition causes a value of 1 to be placed in the file status byte and processing continues). For example, VDUPABND=Y.
VSAMCAT	A/ O,M	N	Retrieves the file attributes for a FILE statement for a VSAM file (KSDS, ESDS, RRDS) from the currently accessed VSAM catalog when the program is compiled. It also allows the user to code VSAM as a file type on the FILE statement. The default is Y. VSAM files using record-level sharing (RLS) require VSAMCAT. For example, VSAMCAT=N. Override at execution time using OPTION keywords VSAMCAT | NOVSAMCAT.

Parameter	Prod / Env	Options	Description
VSAMMSG	II/ O	Y	Allows the compiler to print warning message DYL-1161W when processing a VSAM FILE statement that contains attributes (record format, length) that differ from the VSAM catalog entries for the file, or that are omitted from the FILE statement. The default is N (suppress the message). For example, VSAMMSG=Y.
VSEATTR	A/ D,M	A D T	Establishes the default for VSE I/O processing for Disk (D), Tape (T), or both (A) types of devices, to use VSE control blocks (DTFs) that represent the file attributes and record format specified on the FILE statement. If you do not code this parameter, the compiler continues to use UNDEFINED record format for all non-EXIT DYLIOU or non-VSAM files. The default is N. For example, VSEATTR=D (establishes a default to use the FILE statement attributes and record format in the DTFs for disk files). Override at execution time using OPTION keywords NOVSEALL | NOVSEDISK | NOVSETAPE | VSEALL | VSEDISK | VSETAPE.
WRKFDEV	II/ D,M	2314 3340 3350	Changes the default device type for the VISION:Results work file. No entry is required for 3310, 3330, 3370, 3375, or 3380. The default is 3350. For example, WRKFDEV=2314.
WRKFNAM	II/ D,M	name	Changes the default file name of the VISION:Results work file (the default name is IJSYS04). You can specify a name up to seven characters long. For example, WRKFNAM=SYSWORK.
WRKFSYS	II/ D,M	nnn	Changes the default SYS number for the VISION:Results work file (currently SYS004). Specify a DECIMAL value less than 256. For example, WRKFSYS=19.

Parameter	Prod / Env	Options	Description
XREF$	II/ O	Y	Allows the compiler to include data names that include the character $ on the cross-reference listing. The default is N. For example, XREF$=Y.
ZDIVAB	A/ A	Y R	An entry of Y forces the system to ABEND if a divide by zero is encountered. The default is O which utilizes automatic recovery and sets the results of zero divide operations to 0. An entry of R corrects a divide by zero by changing the divisor to 1 and sets the return code to 4. For example, ZDIVAB=Y. For VISION:Results—Override at execution time using OPTION keywords ZDIVAB | ZDIVORG | ZDIVRC. For OPTION 260 and VISION:Sixty—Override at execution time by setting UDDI to 3 to invoke standard error recovery processing. The arithmetic result is zero.

Note: The following parameters are no longer being used by VISION:Results—PRODCDE, REFNO, TABLEHI, and TOMSG.

Note: The D2PLNID and D2SYSID parameters have been replaced by CATPLAN and CATSYS.

Considerations for Large Programs

This section provides some considerations for running very large programs and describes additional information for the four optional parameters: COMPWRK, GETMAX, FREEMEM, SORTMEM. The DYLINSTL macro defaults of COMPWRK=1500 KB, GETMAX=2500 KB, and FREEMEM=1000 KB are sufficient to run 99 percent of all VISION:Results programs. This topic applies to extremely large programs and programs that have large sort requirements.

COMPWRK

The DYLINSTL macro parameter, COMPWRK, provides an override of the VISION:Results execution-time work area and the VISION:Sixty compiler work area. Of the default 1500 KB value, 25 percent is reserved for internal tables. (For VISION:Results, this consists of the execution and report logic tables. For VISION:Sixty, this consists of D, T, and P parameter data.) The remaining 75 percent of the COMPWRK area is used for work areas, file tables, literals, labels, and so on.

Note: An insufficient allocation results in a DYL-012E error message. When the allocation error does occur, you can increase the value of COMPWRK and rerun the CUSTMJCL job to increase the value for the site, or change the value within the application.

To change the COMPWRK value in a specific program at execution time, use the PARM WA=nnn parameter specified on the JCL EXEC statement. The value specified by nnn is added to (or subtracted from) the value in COMPWORK. (PARM='WA=100' indicates that 100 KB will be added to COMPWRK.) The 25/75 percent split discussed for COMPWRK applies to the value as modified by WA=100.

The JCL EXEC PARM WA= is limited to 3 digits, so you can increase COMPWRK by 999 KB only. Using the JCL WA PARM and changing COMPWRK, and running CUSTMJCL are both required to increase the COMPWRK work area by more than 999 KB.

GETMAX, FREEMEM

GETMAX and FREEMEM control the amount of storage available to VISION:Results at program compilation time. The compiler allocates as much contiguous storage as possible up to a value of GETMAX and returns the amount of storage indicated by FREEMEM back to the operating system. FREEMEM should be increased if the operating system generates a storage allocation error that is not resolved by increasing the region size. However, using REGION=0 MB should prevent this problem.

Note: Very large programs can result in a DYL-997E error message if the GETMAX value is insufficient. If this error occurs, increase the value of GETMAX and rerun the CUSTMJCL job. The FREEMEM value does not need to be increased.

SORTMEM Considerations

Use the following values to reset DYLSORT sort memory size defaults.

Value	Description
MIN,MAX:	SORT is provided with the largest contiguous amount of storage between these values. The default values are 36 KB for MIN and 100 KB for MAX. If nnnK is coded on the SORT statement (VISION:Results) or nnn is placed in columns 10 through 12 of the SORT control parameter (VISION:Sixty), nnn replaces the value of MAX. If this nnn value is less than MIN, it is ignored and MAX remains unchanged. If the MIN amount of storage is unavailable, a U500 ABEND is issued for VISION:Results. In VISION:Sixty, no sort is performed and ER is returned to the SORT control parameter in columns 1 and 2.
COREMAX:	VISION:Sixty, OPTION 260 only. If L is coded in column 10 of the SORT control parameter to exclude storage from SORT, SORT is provided with the largest amount of contiguous storage available between the values of MIN and COREMAX-nnn where nnn is the value from columns 11 through 12 of the SORT control parameter. If the available storage is less than MIN, or less than 24 KB if MIN is less than 24 KB, ER is returned in the SORT parameter list and no SORT is performed. This parameter keeps a minimum of 24 KB of the region unavailable to SORT. The default is 700 KB.
RESETMX:	This value re-establishes the value of MAX after the termination of SORT. The default is 100 KB.

Additional Considerations for Extremely Large Programs

After adjusting COMPWRK/GETMAX to eliminate DYL-012E or DYL-997E errors, the program may abend with program checks using the LIST statement. If this occurs, take the following steps:

1. Separate the program into two requests.
2. Use PICNSAVE to capture those fields on which you want to report.
3. Move the LIST statements to the second request, where PICNSAVE data can be used.

Calling COBOL Programs

VISION:Results allows COBOL programs to be processed with the CALL statement. Before the COBOL program is invoked, VISION:Results will automatically establish the COBOL run-time environment. There are different COBOL run-time environments for VS COBOL II (includes VS COBOL) programs and LE COBOL programs, so the DYLINSTL macro parameter determines the COBOL run-time environment that VISION:Results will use for initialization.

The DYLINSTL macro parameter COBENV is used to initialize the VS COBOL II run-time environment. The DYLINSTL macro parameter LE is used to initialize the LE COBOL environment. These two DYLINSTL parameters are mutually exclusive. The DYLINSTL default is to have the LE parameter turned on and the COBENV parameter turned off. If you have programs that are exceptions to the general rule of your COBOL processing, use the OPTION statement in your program to override the DYLINSTL parameter settings.

Use of the automatic COBOL environment initialization requires that either the COBOL run-time library (if COBENV is being used) or the LE run-time library (if LE is being used) be available at program execution time either on a LINKLIST, STEPLIB, or JOBLIB statement.

DYLINSTL Macro Example and Options List

When using the DYLINSTL MACRO, follow all standard IBM Assembler coding conventions:

- Do not start the macro name (DYLINSTL) in position 1 of the input line.
- Separate the macro name by at least one blank space from the first parameter (ENVIRON in the example shown below).
- Specify the parameters in any order, separated by commas with no intervening blanks.
- Do not extend the input line past position 71.
- If the line must be continued, leave blank spaces after the last comma prior to position 71 and enter any non-blank character is position 72 (x in the following example). The last comma prior to position 71 of a continued statement may be the one separating parameters or items within an option list (such as the SORTMEM parameter in the example below).
- Begin the next parameter (or option list item) in position 16 of the following input line.
- Follow the last parameter (SORTMEM in the example) with blanks.
- Code the final line with the keyword END between positions 2 and 71.

After coding the parameters to be used with the DYLINSTL macro, submit the CUSTMJCL job. The CUSTMJCL job assembles and link-edits the DYLINSTL macro, creates the DYLPCPS module, and places it into the VISION:Results operational library.

The CUSTMJCL example below shows a sample DYLINSTL macro where COBOL II programs are used instead of LE COBOL programs. (The COBENV and LE parameters are mutually exclusive.) Other changes specify the COBOL edit code be changed to E, no writing to PDS files, and change the default from OPTION PRINTDIGITS to OPTION NOPRINTDIGITS.

```
10    16                                                        72
 |     |                                                         |
 DYLINSTL ENVIRON=MVS,LE=N,COBENV=Y,                             x
       PRODUCT=II,COBEDIT=E,NOPOWRT=Y,                           x
       OPTPRDG=N,SORTMEM=(100,500,1000,                          x
       500)
 END
```

The following shows how the DYLINSTL parameter settings will be displayed.

Note: The changed parameters are shown in **red** and **bold**.

```
DEFAULT OPTIONS SPECIFIED ARE:


         ENVIRONMENT IS MVS
         CPUID  2084       10851


         ASALINE(N/), BATCHIQ(IIBATCH/), CBXSIGN(N/), CENTNEW(75/), CENTRY1(/), CENTRY2(/), COBAPOS(Y/),
         COBEDIT(A/E), COBENV(N/Y), COMPERR(N/), COB2NR(N/), COMPWRK(1500K/), CURNCY($/), DATATRN(Y/),
         DECIML9(E/), DELIM(/), DLMFRST(N/), DUPCBNM(N/), DYLVARP(N/), DYL4YR(N/), EDALIGN(N/),
         DP1ZERO(N/), EDSUPR(N/), EURODAT(N/), EURONUM(N/), EXCEL(N/), EXCLPAT(N/), EXPRERR(N/),
         FREEMEM(1000K/), FREEZDD(SYS280FZ/), GETMAX(2500K/), KWDLT(N/), (LE(Y/N), LIBRBUF(60K/),
         LPPUNLMT(N/), LSTSTMX(N/), LTRFROM(N/), LTRZERO(N/), MACHORG(N/), MAXDNLN(50/), MAXDYLF(175/),
         MNAMENU(N/), NAMEHDR(/), NDVRCOM(N/), NDVRENV(/), NODLETE(N/), NOVSOIO(N/), NOPOWRT(N/Y),
         NOSRTAB(N/), NOTOTAL(N/), NOVSIO(N/), NUMCHAR(N/), NUMPD(N/), OPTPRDG(Y/N), OPTPRER(N/Y),
         OUTFILE(N/), PANVBUF(60K/), PDSREPL(N/), PGLINER(55/), PGLINES(55/),
         PROGMOD(XREFREF,CONVENTIONAL,EXP/XREFA), PRTCTRS(N/), PRTZERO(N/), QLF(N/), RANDMPCT(N/),
         RDYONLY(N/), RESRWRD(N/), RETCODE(Y/), RPTASA(N/), RPTDDNM(SYS280R/), RPTXPAG(N/),
         SORTDEV(SYSDA/), SORTDYN(N/), SORTMEM(36K,100K,700K,100K/100K,500K,1000K,500K),
         SORTNAM(SORT/), SSMASK(N/), STRUCGO(N/), SUBRADD(Y/), SUPCOBW(Y/), SUPRESQ(N/), SUP182W(N/), SYSBLOK(N/),
         TIMESEP(D/), TABLEHI(N/), VSAMCAT(Y/), VDUPABND(N/), VSAMMSG(N/), XREF$(N/), ZDIVAB(O/)
```

Run CUSTMJCL to Change the DYLINSTL Macro Parameters

The CUSTMJCL member in YOUR.RESULTS.SOURCE.LIBRARY contains a job to assemble and link-edit the DYLINSTL macro.

This job creates the DYLPCPS module, which has all DYLINSTL parameter changes. The operational library is link-edited with the new DYLPCPS module. The CUSTMJCL member for assembling and link editing the DYLINSTL macro is shown in Figure 143.

When CUSTMJCL is run to customize any parameter, the PRODUCT and ENVIRON parameters must also be specified.

Note: Every time you submit the CUSTMJCL job you must have all the parameters you plan to override coded or the default will be used for the uncoded parameter values.

```
//* YOUR JOB CARD GOES HERE
//ASMLINK PROC DISK=SYSDA,
//             SOUT='*',
//             SOURCLB='YOUR.RESULTS.SOURCE.LIBRARY',        OLD
//             LOADLIB='YOUR.RESULTS.OPERATIONAL.LIBRARY'    OLD
//*
//*********************************************************************
//*
//HLASM   EXEC PGM=ASMA90,REGION=0M,TIME=(,40),                       X
//        PARM=(OBJECT,NODECK,'LINECOUNT(48)','USING(MAP,WARN(3))',   X
//             TERM,'XREF(FULL)')
//SYSLIB       DD  DSN=&SOURCLB,DISP=SHR
//             DD  DSN=SYS1.MACLIB,DISP=SHR
//SYSLIN   DD  DSNAME=&&OBJ,UNIT=&DISK,SPACE=(3040,(80,80),,,ROUND),
//             DISP=(MOD,PASS),
//             DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)
//SYSUT1   DD  DSNAME=&SYSUT1,UNIT=&DISK,SPACE=(1700,(400,50)),
//             SEP=(SYSLIN)
//SYSUT2   DD  DSNAME=&SYSUT2,UNIT=&DISK,SPACE=(1700,(400,50)),
//             SEP=(SYSLIN,SYSUT1)
//SYSUT3   DD  DSNAME=&SYSUT3,UNIT=&DISK,SPACE=(1700,(400,50)),
//             SEP=(SYSLIN,SYSUT1,SYSUT2)
//SYSPRINT DD  SYSOUT=&SOUT
//SYSPUNCH DD  DUMMY
//SYSTERM  DD  SYSOUT=&SOUT
//*
//STEP02  EXEC PGM=IEWL,REGION=0M,PARM='XREF,LIST,NCAL',
//             COND=(0,NE)
//* LINK EDIT THE CUSTOMIZING INFORMATION.
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUT1   DD  UNIT=&DISK,SPACE=(1024,(250,20))
//SYSLMOD  DD  DSN=&LOADLIB.(DYLPCPS),DISP=SHR
//SYSLIN   DD  DSN=&&OBJ,DISP=(OLD,DELETE)
//*
//         PEND
/*
//STEP1   EXEC ASMLINK
//SYSIN  DD  *
         DYLINSTL ENVIRON=MVS,LE=N,COBENV=Y,                           X
               PRODUCT=II,COBEDIT=E,NOPOWRT=Y,                         X
               OPTPRDG=N,SORTMEM=(100,500,1000,                        X
               500)
         END
/*
```

Figure 143 CUSTMJCL Job

Make the following changes to the default symbolic parameters in the JCL procedure before submitting it for execution:

1. Replace the first line of this file with the job statement necessary to run this job at your installation.
2. Change SYSDA in DISK=SYSDA to the device class type or SMS defined unit that is used for the VISION:Results operational library.
3. Change YOUR.RESULTS.SOURCE.LIBRARY to the name of the VISION:Results installation source library.
4. Change YOUR.RESULTS.OPERATIONAL.LIBRARY to the name of the VISION:Results operational library you created.

   Note: Every time you submit the CUSTMJCL job you must have all the parameters you plan to override coded or the default will be used for the uncoded parameter values.

Creating Different Functional Versions of VISION:Results

To create and run different functional VISION:Results systems, create separate versions of DYLPCPS and place them in libraries other than your standard VISION:Results library. By creating different DYLPCPS modules and linking them into separate libraries, you can execute VISION:Results using the macro options specific to that version of DYLPCPS. You can use the sample JCL presented in DYLINSTL Macro Example and Options List on page 114 to assemble DYLINSTL and create DYLPCPS.

You need to change the DSN in the SYSLMOD DD statement to the name of the alternate library containing DYLPCPS. When executing a VISION:Results program, you need to place the alternate DYLPCPS library before the standard VISION:Results library in the STEPLIB DD concatenation in your JCL.

Chapter 4: Reserved Words

VISION:Results uses certain words for its own purposes. You should avoid choosing them for your own data names. For example, the following is invalid:

```
FILE ARFILE INPUT FB 100 3000
  COUNT 5 PD
```

COUNT is a VISION:Results reserved word and cannot be used as a data name in a field definition.

All words starting with DYL are reserved, as well as those in the list beginning on the next page.

Note: DYLVARP is a DYLINSTL macro parameter that allows the use of data names prefixed with DYL. It is required only for VISION:Results Interface to DB2 and VISION:Results Interface to SQL/DS. For a definition of DYLVARP, see the *Advantage VISION:Results Interface to DB2 Reference Guide*.

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

Symbols

$$DUMP
$$KEYLEVEL
$$PCB
$COBOL
$DEFAULT
$DEND
$ECOBOL
$ELSE ♣
$ENDGDEF ♣
$ENDGSET ♣
$ENDSET ♣
$GDEF ♣
$GSET ♣
$IF
$IFE
$IFVALUE
$PRINT ♣
$SET ♣
$
% ♣
(
)
*
**
+
-
/
@ ♣
@nnn ♣
;
:
,
=
% macroname ♣
#anything

A

ABOVE
ABSOLUTE
ACCEPT
ACCEPTABLE ♥
ACCEPTANCE ♥
ACCTITLE ♥
ADD
ADVANCED ♣
AFTER
AGE ♥
AGEDATE ♥
AGEING ♣
AGING ♣
ALIAS
ALLOCATE
ALPHANUMERIC ♥
AMOUNTFIELD ♥
AND
AREA ♣
ARRAY ♣
ASA
ASCENDING ♣
ASCII
ASIS ♣
AT
ATTRIBUTE ♥
AUDIT ♣

B

BDAM
BEFORE
BELOW
BETWEEN
BI
BINSEARCH ♣
BITS
BLANK
BLANKS
BOOK
BOTH
BY
BYPASS

C

CALL
CALL_ATTACH
CARD
CARDS
CASE
CATPLANID
CATSYSID
CDLOAD
CELL ♥
CH
CHANGE
CHnn ♣
CLOSE
CMX
COBAPOS
COBOL
COBXSIGN
COB2NR
COLUMNS
COMBINE
COMPAREN ♣
CON ♣
CONFIDENCE ♥
CONTROL
CONVENTIONAL
COPY
COPYC
COPYDB2
COPYE
COPYL
COPYN
COPYP
COPY260
CORR
CORRELATION ♥
CORRESPONDING
COUNT
CS
CSnn ♣
CTLBRK ♥
CTRLZ
CURRENCY
CYLOFL

D

DA
DATA
DATAVER
DATEFIELD ♥
DATEFORMAT ♥
DB2PLANID
DB2SYSID
DD/MM/YY ♥
DD/MM/YYYY ♥
DDMMYY ♥
DDMMYYYY ♥
DDNAME
DE
DECLARE
DELETE ♣
DELIM
DESCENDING ♣

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

DETAIL ♥
DIM ♣
DIMENSION ♣
DIRECTORY
DISCOVERY ♥
DISK
DISKETTE
DISKPASSWORD
DISTFIELD ♥
DLM
DMAP
DNA
DNE
DOLLAR ♥
DOS
DOUNTIL
DOWHILE
DROPCRLF
DROPERR
DUMMY
DUPLICATE ♣
DYACCEPT
DYL
DYL-SP
DYLCNAME0A
DYLCNAME0B
DYLCNAME0C
DYLCNAME0D
DYLCNAME0E
DYLCNAME0F
DYLCNAME0G
DYLCNAME0H
DYLCNAME0I
DYLCNAME0J
DYLCNAME0K
DYLCNAME0L
DYLCNAME0M
DYLCNAME0N
DYLCNAME0O
DYLCNAME0P
DYLCNAME0Q
DYLCNAME0R
DYLCNAME0S
DYLCNAME0T
DYLCNAME0U
DYLCNAME0V
DYLCNAME0W
DYLCNAME0X
DYLCNAME0Y
DYLCNAME0Z
DYLCNAME00
DYLCNAME1
DYLCNAME10
DYLCNAME11
DYLCNAME12
DYLCNAME13
DYLCNAME14
DYLCNAME15
DYLCNAME16
DYLCNAME17
DYLCNAME18
DYLCNAME19
DYLCNAME2
DYLCNAME20
DYLCNAME21
DYLCNAME22
DYLCNAME23
DYLCNAME24
DYLCNAME25
DYLCNAME26
DYLCNAME27
DYLCNAME28
DYLCNAME29
DYLCNAME3
DYLCNAME30
DYLCNAME31
DYLCNAME32
DYLCNAME33
DYLCNAME34
DYLCNAME35
DYLCNAME36
DYLCNAME37
DYLCNAME38
DYLCNAME39
DYLCNAME4
DYLCNAME40
DYLCNAME41
DYLCNAME42
DYLCNAME43
DYLCNAME44
DYLCNAME45
DYLCNAME46
DYLCNAME47
DYLCNAME48
DYLCNAME49
DYLCNAME5
DYLCNAME50
DYLCNAME51
DYLCNAME52
DYLCNAME53
DYLCNAME54
DYLCNAME55
DYLCNAME56
DYLCNAME57
DYLCNAME58
DYLCNAME59
DYLCNAME6
DYLCNAME60
DYLCNAME61
DYLCNAME62
DYLCNAME63
DYLCNAME64
DYLCNAME65
DYLCNAME66
DYLCNAME67
DYLCNAME68
DYLCNAME69
DYLCNAME7
DYLCNAME70
DYLCNAME71
DYLCNAME72
DYLCNAME73
DYLCNAME74
DYLCNAME75
DYLCNAME76
DYLCNAME77
DYLCNAME78
DYLCNAME79
DYLCNAME8
DYLCNAME80
DYLCNAME81
DYLCNAME82
DYLCNAME83
DYLCNAME84
DYLCNAME85
DYLCNAME86
DYLCNAME87
DYLCNAME88
DYLCNAME89
DYLCNAME9
DYLCNAME90
DYLCNAME91
DYLCNAME92
DYLCNAME93
DYLCNAME94
DYLCNAME95
DYLCNAME96
DYLCNAME97
DYLCNAME98
DYLCNAME99
DYLCOMMON
DYLCOMRG
DYLCOUNT0A
DYLCOUNT0B
DYLCOUNT0C
DYLCOUNT0D
DYLCOUNT0E
DYLCOUNT0F
DYLCOUNT0G
DYLCOUNT0H
DYLCOUNT0I
DYLCOUNT0J

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

DYLCOUNT0K
DYLCOUNT0L
DYLCOUNT0M
DYLCOUNT0N
DYLCOUNT0O
DYLCOUNT0P
DYLCOUNT0Q
DYLCOUNT0R
DYLCOUNT0S
DYLCOUNT0T
DYLCOUNT0U
DYLCOUNT0V
DYLCOUNT0W
DYLCOUNT0X
DYLCOUNT0Y
DYLCOUNT0Z
DYLCOUNT00
DYLCOUNT1
DYLCOUNT10
DYLCOUNT11
DYLCOUNT12
DYLCOUNT13
DYLCOUNT14
DYLCOUNT15
DYLCOUNT16
DYLCOUNT17
DYLCOUNT18
DYLCOUNT19
DYLCOUNT2
DYLCOUNT20
DYLCOUNT21
DYLCOUNT22
DYLCOUNT23
DYLCOUNT24
DYLCOUNT25
DYLCOUNT26
DYLCOUNT27
DYLCOUNT28
DYLCOUNT29
DYLCOUNT3
DYLCOUNT30
DYLCOUNT31
DYLCOUNT32
DYLCOUNT33
DYLCOUNT34
DYLCOUNT35
DYLCOUNT36
DYLCOUNT37
DYLCOUNT38
DYLCOUNT39
DYLCOUNT4
DYLCOUNT40
DYLCOUNT41
DYLCOUNT42
DYLCOUNT43
DYLCOUNT44
DYLCOUNT45
DYLCOUNT46
DYLCOUNT47
DYLCOUNT48
DYLCOUNT49
DYLCOUNT5
DYLCOUNT50
DYLCOUNT51
DYLCOUNT52
DYLCOUNT53
DYLCOUNT54
DYLCOUNT55
DYLCOUNT56
DYLCOUNT57
DYLCOUNT58
DYLCOUNT59
DYLCOUNT6
DYLCOUNT60
DYLCOUNT61
DYLCOUNT62
DYLCOUNT63
DYLCOUNT64
DYLCOUNT65
DYLCOUNT66
DYLCOUNT67
DYLCOUNT68
DYLCOUNT69
DYLCOUNT7
DYLCOUNT70
DYLCOUNT71
DYLCOUNT72
DYLCOUNT73
DYLCOUNT74
DYLCOUNT75
DYLCOUNT76
DYLCOUNT77
DYLCOUNT78
DYLCOUNT79
DYLCOUNT8
DYLCOUNT80
DYLCOUNT81
DYLCOUNT82
DYLCOUNT83
DYLCOUNT84
DYLCOUNT85
DYLCOUNT86
DYLCOUNT87
DYLCOUNT88
DYLCOUNT89
DYLCOUNT9
DYLCOUNT90
DYLCOUNT91
DYLCOUNT92
DYLCOUNT93
DYLCOUNT94
DYLCOUNT95
DYLCOUNT96
DYLCOUNT97
DYLCOUNT98
DYLCOUNT99
DYLDATE
DYLDATE4
DYLDATEPAG
DYLDATEPG4
DYLDLA
DYLETIME
DYLEXTA
DYLEXTAA
DYLFQX
DYLFQxx
DYLGREG
DYLGREG4
DYLILDATE
DYLJULIAN
DYLJULIAN4
DYLLINE
DYLNOTOT
DYLNRP
DYLONE
DYLPAGE
DYLPAGE1
DYLPAGE2
DYLPAGE3
DYLPAGE4
DYLPAGE5
DYLPAGE6
DYLPAGE7
DYLPARM
DYLPARMLEN
DYLPPAGE
DYLPRCA
DYLPROGRAM
DYLPRTCOMM
DYLPRTMAXL
DYLPRTNUMB
DYLPRTPAGE
DYLREMAIN
DYLREPMAXL
DYLRESET
DYLRETURN
DYLSYSDATE
DYLSYSPARM
DYLTIME
DYLTRA
DYLTRAA
DYLTRAN

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

DYLUPSI
DYL280RA
DYL4YEAR
DYNAM
DYNAMDB2
DYNEXT
DYSTORE
D4

E

EBCDIC
EDIT
EDOS
EJECT ♣
ELSE
END
ENDCASE
ENDDO
ENDIF
ENDLTD ♣
ENDONE
ENTRY ♣
EOS
EQ
EQUAL ♥
ERASE
ERROR
ESDS
ESTIMATION ♥
EXCEL
EXCELPAT
EXCLUSIVE
EXEC
EXIT
EXITCASE
EXITDO
EXOR
EXPRTERR
EXT
EXTENSION
EXTENTS

F

F
FB
FBA
FFFF
FFFFEE
FFFFFF
FIELDNAME
FIELDS
FILE
FILENAME
FILES
FILL ♣
FILLER
FINAL
FIRSTDUP ♣
FIXED
FMX
FOOTING
FOR ♣
FORMAT
FREE
FREEZDD
FREEZE
FREQ ♥
FREQUENCY
FROM
FUTURE ♥

G

GE
GETCOM
GOTO
GROUP1 ♥
GROUP10 ♥
GROUP2 ♥
GROUP3 ♥
GROUP4 ♥
GROUP5 ♥
GROUP6 ♥
GROUP7 ♥
GROUP8 ♥
GROUP9 ♥
GT

H

HEX
HEXPRINT
HIDDEN
HIGHAMT ♥
HIGHVALUES
HISTOGRAM ♥
HOLD
HTAB
HTML
HTMLTITL

I

IDLEN ♣
IDLENGTH ♣
IDMSMOVE
IF
INDENT ♣
INPUT
INTERPRET
INTERVAL ♥
INTERVALNO ♥
INTERVALSIZE ♥
INTO
INTSEL
INW
INX
INY
INZ
IO
IQBATCH
IQFILE
IQREAD
ISAM
ITEMFIELD ♥
ITEMNO ♥
ITERATE

J

JCL

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

K

KEY ♥
KEYLEN
KEYLOC
KEYn ♣
KSDS

L

LASERnn ♣
LASTDUP ♣
LCPRINT
LE
LEFT
LEFTSAVE ♣
LENGTH
LETTER ♣
LIB ♣
LIBNAME
LIBPASSWORD
LINEAR ♥
LIST
LISTAG
LISTAUD
LISTINS
LIST26
LOGARITHMIC ♥
LOGIC
LONG
LOWAMT ♥
LOWER
LOWVALUES
LT
LTD ♣
LTDnn ♣
LTH ♣
L2U

M

MASTER
MASTERUP ♣
MATCH ♣
MATCHED ♣
MAXBLKSIZE
MAXIMUM ♥
MEAN ♥
MEMBER
MERGE ♣
MIXED
MM/DD/YY ♥
MM/DD/YYYY ♥
MMDDYY ♥
MMDDYYYY ♥
MODIFY
MONETARY ♥
MOVE
MQBACK
MQBEGIN
MQCLOSE
MQCMIT
MQCONN
MQCONNX
MQDISC
MQGET
MQINQ
MQOPEN
MQPUT
MQPUT1
MQSET
MSGCSECT

N

NA
NAMED
NB
NDA
NDE
NDV
NDVRCOM
NDVRENV
NDVRSTG
NDVRSUB
NDVRSYS
NDVRTYP
NE
NEGATIVE
NEW
NEWEXT
NEWEXTENSION
NEWFILENAME
NEWNAME
NEWPAGE
NEWPARA ♣
NEXT
NL
nn
NO ♥
NOCDLOAD
NOCOB2NR
NOCOBAPOS
NOCOBXSIGN
NODYL4YEAR
NOEDIT ♣
NOEJECT ♣
NOERROR
NOEXCEL
NOEXCELPAT
NOEXPRTERR
NOFEED
NOLE
NONDVRCOM
NONUMCHAR
NONUMPD
NOOPTIMIZE ♣
NOOPTLIST
NOPRINTCTRS
NOPRINTD
NOPRINTDIGITS
NOPRINTEP
NOSYSBLOCK
NO_PRINT_ENTRY_POINTS
NOQLF
NOQUALIFIERS
NORETAIN
NORMALIZED
NORWD
NOSORTAB
NOSPOOL
NOSQL2DBC
NOSUM
NOT
NOTOTAL
NOVSAMCAT
NOVSEALL
NOVSEDISK
NOVSETAPE
NOXREF
NP
NU
NULL
NULLOFF
NULLON
NUMCHAR
NUMERIC
NUMFILES
NUMPD
NZ

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

O

OFFSET
ON
ONE
ONEBUFF
ONES
OPTIMIZE ♣
OPTION
OPTIONS
OPTLIST
OR
ORIGINAL
OS
OTHERWISE
OUTPUT
OVERAUDITING ♥

P

PAD
PAN ♣
PARM
PARTKEY
PASSWORD
PAST ♥
PATCHES
PATHNAME
PATHNAME01
PATHNAME02
PATHNAME03
PATHNAME04
PATHNAME05
PATHNAME06
PATHNAME07
PATHNAME08
PATHNAME09
PATHNAME1
PATHNAME10
PATHNAME11
PATHNAME2
PATHNAME3
PATHNAME4
PATHNAME5
PATHNAME6
PATHNAME7
PATHNAME8
PATHNAME9
PCFILE
PCWRITE
PD
PDS
PDSREPA
PDSREPN
PD0
PERCENTAGE ♥
PERFORM
PICNSAVE
PLANID
PLUnnn
PMX
POPULATION ♥
POSITION
POSITIVE
PRECISION
PRINT
PRINTCTRS
PRINTD
PRINTDIGITS
PRINTEP
PRINT_ENTRY_
  POINTS
PRINTERR
PRINTERROR
PRINTGEN ♣
PRINTNOGEN ♣
PRIOR
PROPORTIONAL ♥
PROTECT
PUTCOM

Q

QLF
QUALIFIERS
QUIKJOB
QUIT
QUITALL

R

RANDOM
RANDOMX
RANDPROG ♥
RANDPROGSIZE ♥
READ
READBOOK
READDIR
READIR
READMEM
READONLY
RECFM
REDEF
REDEFINE
REGRESSION ♣
REINIT
REJECT
RELBLK
RELBYTE
RELBYTEX
RELIABILITY ♥
RELREC
RELSORT
RELTRK
REPLACE
REPLACING
REPLICA ♣
REPORT
REPORTFILE
REPORTn
REPORTnn
REPORTnnn
REPTITLE
RESTART
RESTORE
RETAIN
RETRIEVE ♣
RETSORT
REUSE
REWIND
RIGHT
ROUND
ROUNDED
RRDS

S

SAMPLE ♥
SAMPLING
SB
SCATTER ♣
SCHEMADD
SCHEMADS
SCHEMAHF
SCHEMAUL
SCRATCH
SEARCH ♣
SECURE
SEED ♥

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

SELECT
SEQUENTIAL
SETREAD ♣
SIZE
SKIP
SKIPX
SL
SMALL ♥
SORT
SORTING
SPACE
SPACES
SPn ♣
SPT
SPOOL
SQL2DBC
SSL
STANDARD ♥
START ♥
STATEOFF
STATEON
STATSQL
STATUS
STATUSFLAG
STDREPT ♥
STOP
STOPALL
STORE ♣
STRATIFIED ♥
STRATUM1 ♥
STRATUM10 ♥
STRATUM2 ♥
STRATUM3 ♥
STRATUM4 ♥
STRATUM5 ♥
STRATUM6 ♥
STRATUM7 ♥
STRATUM8 ♥
STRATUM9 ♥
STRUCTURED
STRUCTURED2
STYLE
SUBDIR
SUBTOTAL
SUM
SUMMARY ♥
SUM1
SUM2
SUM3
SUM4
SUM5
SUM6
SUM7
SUPPRESS

SWITCH ♣
SYR
SYSBLOCK
SYSnnn
SYSTEMID
SYS280RA
S2R

T

TABAREA
TALLYn
TABLE ♣
TAPE
TEST
TEXT
THEN
THROUGH
THRU
TIMECOLON
TIMEDOT
TITLE ♣
Tn
TO
TOTAL
TPRINT
TRAN
TRANSLATE
TREND ♣
TSIZE ♣
TSO_ATTACH
TSORT ♣
TWOUP ♣

U

UDTL
UNIT ♥
UNL
UNLOAD
UNSIGNED
UNTIL
UNWARRANTED ♥
UPPER
USE
USERDEFAULT
USERINX
USERINXnn
USING

V

V
VAL
VALUE
VARYING
VB
VERIFY
VFILEON
VOLVER
VSAM
VSAMCAT
VSAMEC
VSAMRC
VSEALL
VSEDISK
VSETAPE

W

WHEN
WIDE
WITH
WORK
WORKAREA
WORKSHEET ♥
WRITE
WRITEALL ♣
WRITEDIR
WRITEMEM

Note: You cannot use the blue reserved words (marked with a ♥) as data names if you ever intend to use OPTION EXCEL. You can only use the red reserved words (marked with a ♣) and blue reserved words (marked with a ♥) as data names if the DYLINSTL parameter RESRWRD is used, but this disables certain VISION:Results capabilities and CA does not recommend doing this.

X

XLATE
XMLFILE
XMLWRITE
XNAME ♣
XREF
XREFA
XSCATTER ♣

Y

YES ♥
YNAME ♣
YR1
YR2
YSCATTER ♣
YY/MM/DD ♥
YYDDD ♥
YYMMDD ♥
YYYY/MM/DD ♥
YYYYMMDD ♥
YYYYDDD ♥
Y2C
Y2D
Y2P
Y2S
Y2B
Y2Z

Z

ZDIVAB
ZDIVORG
ZDIVRC
ZERO ♥
ZONE

Numbers

2311
2314
260
3330
3340
3350
3375
3380

Definitions

Name Type	Length	Data Type	Edit	Decimals
BLANK or BLANKS	1	CH		
One space. Can be used to space fill any size character field because a character field is space-filled to the right after a value is moved into it.				
DYLCENTRY1 DYLCENTRY2	2	CH		
Use to assign a century prefix for 2-digit years to produce a 4-digit output. If DYLCENTRY1 or DYLCENTRY2 parameters are not set, VISION:Results defaults to the values set in the DYLINSTL parameters CENTRY1 and CENTRY2, respectively. If no values are set in CENTRY1 and CENTRY2, the system defaults to the DYLINSTL parameter CENTNEW.				
DYLCNAMEnn	20	CH		
Contains 126 twenty-byte label fields that you can set to identify the corresponding DYLCOUNTnn counters when they are printed. nn may be 1-99, 00, and 0A-0Z. If you do not move your own labels into these fields, the standard identifiers are to the left of the counters as TOTAL NUMBER 01, TOTAL NUMBER 02, and so on.				

Name Type	Length	Data Type	Edit	Decimals
DYLCOMMON	4	BI		
This field contains the address of the VISION:Results common area. It is used as a parameter that is passed using CALL to a user module. You might need this to gain access to the VISION:Results DTF/DCBs.				
DYLCOMRG	11	CH		
The value is used for multiple purposes. It is a copy of bytes 12-22 of the VSE partition communications region. It is used to communicate the XML API completion code and reason code. It contains the value of register 15 for the CSV routines, TOCSV and FROMCSV. Changing the value in DYLCOMRG has no effect on the actual value in the communications region.				
DYLCOUNTnn	10	PD	A	0
Contains 126 ten-byte packed counters that you can use to maintain your own totals. nn may be 1-99, 00, and 0A-0Z. If you reference any of these counters, the series is printed as part of the end-of-run file statistics.				
DYLDATE	8	CH		
The initial value is the current date in MM/DD/YY format.				
DYLDATE4	10	CH		
The initial value is the current date in MM/DD/YYYY format.				
DYLDATEPAG	30	CH		
The current date and page number edited and ready for insertion in report titles. The format is: DATE MM/DD/YY PAGE NNNNNNN				
DYLDATEPG4	32	CH		
The current date and page number edited and ready for insertion in report titles. The format is: DATE MM/DD/YYYY PAGE NNNNNNN				
DYLDLA	4	BI		
The first parameter in a CALL statement when invoking DYLTDLI (IMS/DLI database interface option). DYLDLA should never be modified in your program.				
DYLETIME	8	CH		
Value is the execution start time of VISION:Results in HH.MM.SS format.				
DYLGREG	4	PD	D	
The current Gregorian date in packed format. Packed format is 0MMDDYYs.				

Name Type	Length	Data Type	Edit	Decimals
DYLGREG4	5	PD	D	
The current Gregorian date in packed format. Packed format is 0MMDDYYYYs.				
DYLILDATE	4	BI		
The current Lilian date in binary format.				
DYLJULIAN	3	PD		
The current Julian date in packed format. The format is YYDDD.				
DYLJULIAN4	4	PD		
The current Julian date in packed format. The format is YYYYDDD.				
DYLLINE	2	PD	A	
The line count of the current report page in packed format.				
DYLNOTOT	1	CH		
Control total inhibit. Can be set by the user. b̸ = Not used (initial setting). 1 = Do not increment control break totals on this entrance to the report writer.				
DYLNRP	1	CH		
A detail line inhibit switch that you can set (enter a 2) before the first detail line LIST statement is issued.				
DYLONE	1	PD		
Value 1. Useful for counting.				
DYLPAGE	13	CH		
The literal PAGE followed by the edited page number. The format is: PAGE NNNNNNN				
DYLPAGE1	1	CH		
Edited 1-character page number.				
DYLPAGE2	2	CH		
Edited 2-character page number.				
DYLPAGE3	3	CH		
Edited 3-character page number.				
DYLPAGE4	4	CH		
Edited 4-character page number.				

Name Type	Length	Data Type	Edit	Decimals
DYLPAGE5	5	CH		
Edited 5-character page number.				
DYLPAGE6	6	CH		
Edited 6-character page number.				
DYLPAGE7	7	CH		
Edited 7-character page number.				
DYLPARM	60	CH		
Variable data provided by either the PARM parameter in the EXEC statement (z/OS only) or the DATA xxxx phrase in the OPTION statement.				
DYLPARMLEN	2	PD		
The length of the data stored in DYLPARM.				
DYLPPAGE	6	PD	A	
The current report page number. Can be reset.				
DYLPROGRAM	8	CH		
z/OS: A user-specified program identification name. The default is the library member name assigned to the SYSIN DD statement.				
DYLPRTCOMM	10	CH		
This field can be set to hold an identifier to print to the left of a record or field being printed using the print (PRINT, HEXPRINT, and so on) commands. If you do not move anything in before printing, the file name or data name appears.				
DYLPRTMAXL	2	PD		
The maximum lines per page to be allowed for file printing. It is set either by default or by using the LONG keyword of the OPTION statement.				
DYLPRTNUMB	4	PD		
The count of records file printed by VISION:Results.				
DYLPRTPAGE	3	PD		
The page number for the file print listing.				
DYLREPMAXL	2	PD		
The maximum lines per page allowed for the report.				
DYLRESET	1	BI		
Reset report page number after break indicator. 0 = Not used (initial setting). 1-6 = Reset report page number to 1 after break 1-6.				

Name Type	Length	Data Type	Edit	Decimals
DYLRETURN	2	BI		
Set this before a run terminates to cause a condition code to be passed to the next step or to cause an ABEND to occur. Setting DYLRETURN does not cause the run to terminate. Processing continues until all files are at end, or a STOP or a QUIT command is executed. 0 = Normal completion. 1-99 = Condition code value to be passed. 100-4095 = ABEND with system dump when processing terminates.				
DYLSYSDATE	9	NU		
VSE only. The system date from the partition communication region (offset X`4F'). The first six bytes contain the Gregorian date (MMDDYY or DDMMYY), and the last three bytes contain the Julian day number (DDD). This date is not affected by the // DATE statement.				
DYLSYSPARM				
z/OS: DYLSYSPARM contains the PCB address list reflected in the PSB. For details, see Database Access in the *Advantage VISION:Results for z/OS Reference Guide.* VSE: Size 8, type CH. DYLSYSPARM contains 0 to 8 bytes of data from the SYSPARM=statement on your EXEC statement, padded on the right with blanks. This is release-dependent.				
DYLTIME	4	PD	A	
Value is the execution start time of VISION:Results. Packed format is 0HHMMSS.				
DYLTRAN	1	CH		
Refers to the first byte of a 256-byte translate table. If you do not want to use the FROM and TO option of the TRANSLATE command, you can specify your own translate table by moving values to the appropriate offset from DYLTRAN.				
DYLUPSI	1	BI		
Value is a copy of the VSE UPSI byte. Changing this value has no effect on the actual UPSI byte.				
INW	2	BI	A	
An index used to manipulate a table or fields and loop through records.				
INX	2	BI	A	
An index used to manipulate a table or fields and loop through records.				
INY	2	BI	A	
An index used to manipulate a table or fields and loop through records.				
INZ	2	BI	A	
An index used to manipulate a table or fields and loop through records.				

Name Type	Length	Data Type	Edit	Decimals
SPACE or SPACES	1	CH		
One space. Can be used to space fill any size character field because a character field is space-filled to the right after a value is moved into it. Same as BLANK or BLANKS.				
TALLY	16	PD	Z	9
The total of records printed or accepted in the report since the last control break.				

Chapter 5: Programming Tips

This chapter contains programming tips for you to consider when using VISION:Results.

Punctuation Limitations

Keep these punctuation limitations in mind when programming VISION:Results.

Colons and Periods

Punctuation such as colons (:) or periods (.) should never be used to delimit command statements, parameter lists, or a list string in an IF statement.

The exception to the above is in expressing numeric literals that can have commas, a decimal point, and a negative amount indication and reference point tag names that must be followed by a colon (:).

```
591,324.08
-.01
LISTDET:
```

Asterisks and Semicolons

Asterisks (*) and semicolons (;) are reserved for comments. The asterisk (*) can be used only if placed in position 1 (byte 1) of the input record, regardless of whether the OPTION COLUMNS override is used. (The asterisk is also used as the arithmetic symbol for multiplication.) The semicolon can appear anywhere (1-80) in the free-form input record.

```
* COMMENTS REMARK
; COMMENTS
MOVE FIELDA TO FIELDB   ; DATE TRANS
```

Continuation Rules

Data names, field names, keywords, and commands cannot be phonetically or otherwise placed on separate lines.

```
MOVE FIELDA TO
 FIELDB

LIST BALANCE ACCOUNT TRANS
 WITH 2 AFTER

WBALANCE = (BALANCE*2.15)/3
 IF WBALANCE GE 100 GOTO GOOD ENDIF
```

Literals can be continued on another line only in Tn statements (maximum is 204) and the VALUE keyword (maximum 255) in field definitions. Literals in a MOVE statement can be 70 characters long. Literals in other procedure logic statements (maximum 20) and literals in a LIST statement (maximum is 30 characters each) cannot be continued on separate lines.

```
MOVE ’ABCDEFGHIJKL’
 TO FIELDC

MOVE FIELDA TO FIELDB MOVE
 32157 TO COUNTER

DATANAME1 20 VALUE ’CHECK’
 ’ING OUT TIME’

DATANAME2 30 VALUE ’TOTA’
 ’L’’S THE SUM OF’

DATANAME2 30 VALUE ’TOTA’
 “L’S THE SUM OF”

LIST SUM BALANCE AT TOTAL
 ’FINAL TOTAL’ AT TOTAL+20
```

Naming Conventions

The prefix DYL is reserved for VISION:Results use only.

Data Names

Data names (field names) can be from 2 to 50 characters (installation default is 50 characters) long and must start with an alphabetic character. Data names cannot be the same as reserved words. Data names can be alphanumeric but cannot include any special symbols (for example, - + * / , ; : . = @ ( ) or blanks) except the underscore.

```
MM_DD_YY    DATE
TRANSDATE   DATES
ACCT_CODE   AMT_FLD
```

File Names

File names can be from 1-8 (1-7 for VSE) characters long and must start with an alphabetic character. File names can be alphanumeric but should never contain any punctuation or arithmetic symbols.

```
PAYROLL    ARFILE
PAYMSTER   ARFILE
```

Tag Names

Tag names can be from 1-10 characters long and can start with an alphabetic character or a number. Tag names cannot be the same as reserved words. Tag names can be alphanumeric but should never contain any punctuation or arithmetic symbols.

The exception for using # is when it is a symbolic variable in a COPY member or book where substitution occurs prior to validation.

The exception for using the colon (:) is when the tag name is shown as the reference point.

```
GOTO         LISTDET
GOTO         WRT_RECORD
```

```
LISTDET:
WRT_RECORD:
```

Note: The length can be 1-10 plus a colon.

Number of Characters per Input Line

The free-form input to VISION:Results is 80-byte (position) records; the first 72 bytes (VISION:Results default) contain the free-form statements. This leaves bytes (positions) 73 to 80 for an identification or sequence field.

The default can be changed, if desired, by the COLUMNS keyword in the OPTION statement. To change the start and end (the minimum is 30 positions) of the VISION:Results free-form in the free-form input record, use:

```
OPTION COLUMNS 10 60
```

In the above, all valid entries will begin in byte (position) 10 and end in byte (position) 60 of the free-form input record. All other information will be considered as extraneous and will not be validated. COLUMNS nn nn must be entered on a statement by itself and must be within the bounds of both the old and new free-form columns.

However, a statement used for comments, and denoted as such by an asterisk (*), must have the asterisk coded in position 1 regardless of the COLUMNS option chosen.

Literals

There are instances when you need to express a literal in your program.

To move a value to a field:

```
MOVE 'CA' TO STATE
MOVE 100 TO AMOUNT
```

To compare to a field in a conditional statement:

```
IF AMOUNT GE 500.75 GOTO TO PROC3
IF STATE EQ 'CA' GOTO PROC4
```

To assign an initial value to a field during its definition:

```
FILE FILEOUT OUTPUT FROM FILEOUT
  FIELDA  5 30 PD   VALUE 0
  FIELDC 15 50   VALUE 'LOS ANGELES'
WORKAREA
  BASEAMOUNT 5 PD 2   VALUE 250.50
```

To cause a constant to be printed on a line of a report:

```
ON CHANGE IN COSTCTR
  LIST '*COST CENTER TOTAL' AT NAME
```

To cause a constant to be printed during file print:

```
PRINT 'ERROR 201'
```

To use a numerical value in an arithmetic expression:

```
AMOUNT=AMOUNT*1.20
COUNT=COUNT+1
NEGVALUE=AMOUNT*-1
```

Character or Alphanumeric Literals

Enclose a character or alphanumeric literal, such as 'JONES' or '201-22,' in single or double quotation marks to distinguish it from a data name.

```
MOVE 'JONES' TO NAME
IF STATE EQ 'CA'
  MOVE 'CALIFORNIA' TO STATENAME
PRINT 'ERROR 555'
```

Repeat the quotation mark to create a literal that contains a quotation mark.

```
MOVE 'IT''S' TO FIELDA
```

FIELDA will contain IT'S.

Moving Character Literals

You can move a character literal to a character field. The literal’s maximum length is 70 and the field’s maximum is 32767. You must code the literal on one line; it cannot be continued. If the literal is smaller than the field, the literal is left-aligned and padded with blanks.

```
MOVE ’JONES’ TO NAME
```

If NAME is a 10-byte field, the receiving field will contain JONESb/b/b/b/b/.

You cannot move literals to variable-length fields.

When padding occurs, the length of the receiving field is used to determine how many blanks are moved. No checking is done to ensure that an indexed MOVE does not exceed the boundaries of the receiving field. Use the LENGTH keyword to limit the amount of padding.

Comparing Character Literals

You can compare a character literal to a character field. The literal’s maximum length is 20, and the field’s maximum is 256. The literal must appear to the right of the relational operator (EQ, GT, and so on).

If the literal is smaller than the field, the literal will be padded with blanks before the compare is done.

For example, if NAME is a 10-byte field, the compare will work like this:

```
IF NAME EQ ’JONES’
```

Assigning Character Literals to Fields

If you are defining a character type field in an output FILE or WORKAREA statement, you can assign an initial value to that field using a character literal. If the literal is smaller than the field, the literal is left-justified and padded with blanks.

```
WORKAREA
  FIELD1 10  VALUE ’CODE G’
```

Because the literal ’CODE G’ is only 6 bytes and FIELD1 is a 10-byte field, 4 blanks are appended to the value.

A literal can be up to 255 bytes long. You can continue this type of literal onto several lines by breaking it into multiple literals surrounded by quotation marks.

```
WORKAREA
  ALPHANUM 36 VALUE ’ABCDEFGHIJ’
    ’KLMNOPQRSTUVWXYZ0123456789’
```

Listing Literals in a Report

You can code a character type literal in your LIST statement in place of a field name. This causes the value to print on the report.

```
ON CHANGE IN DEPARTMENT
  LIST 'DEPARTMENT TOTAL' AT NAME
SUM SALARY
```

When a control break occurs caused by a change in the value of DEPARTMENT, a line is printed that contains the constant 'DEPARTMENT TOTAL' along with the other fields specified.

The maximum size of a literal appearing in a LIST statement is 30. It must be coded on one line; it cannot be continued.

Printing a Literal in File Print

You can use the file print feature (PRINT, HEXPRINT, HEX, and so on) to print a message or identifier to the SYSPRINT (z/OS) or SYSLST (VSE) file.

```
PRINT 'ERROR 321'
PRINT 'BEGIN FILEA PRINT'
```

The literal can be up to 70 bytes long.

Numeric Literals

A numeric literal such as 0, 595, 300.50, or -1 can be expressed without quotation marks. It can include a leading minus sign, commas, and a decimal point. It can contain a maximum of 16 decimal digits. The total length of the numeric literal must not exceed 20 bytes. It can be used in operations with numeric, packed, or binary type fields.

```
MOVE 0 TO AMOUNT
IF AMOUNT GE 500.50 GOTO ...
COUNT=COUNT+1
CALC=FIELDA*37.5/100
```

Decimal alignment is automatic.

```
MOVE 400 TO AMOUNT
```

If AMOUNT is a 2-decimal position field, 400 will actually be moved as though it were 400.00.

```
WORKAREA
  FIELDA  5  2  PD  VALUE 0
```

The 2-byte field 'FIELDA' is initialized to zero.

Moving Numeric Literals

You can move a numeric literal to a numeric, packed, or binary field.

```
MOVE 9,999.99 TO AMOUNT
MOVE -700 TO AMOUNT2
```

AMOUNT or AMOUNT2 could have been defined as numeric (NU), packed (PD), or binary (BI) fields.

Comparing Numeric Literals

You can compare a numeric, packed, or binary field to a numeric literal. The literal must appear to the right of the relational operator (EQ, GE, and so on).

```
IF AMOUNT GE 0
IF AMOUNT2 GT 9,999.99
```

Using Numeric Literals in Arithmetic Expressions

Numeric literals can be used in arithmetic expressions with other numeric literals, numeric packed, or binary fields.

```
AMOUNT=AMOUNT*1.10
COUNT1=COUNT1+1
COUNT2=(AMOUNT-1.75)/4
NEGAMT=AMOUNT*-1
AMOUNT2=FIELDA+1,245,373.25
```

Assigning Numeric Literals to Numeric, Packed, or Binary Fields

If you are defining a numeric, packed, or binary field in an output FILE or WORKAREA statement, you can assign a value (up to nine decimal positions) to that field using a numeric literal.

```
WORKAREA
  FIELDA 5 PD 2  VALUE 0
  FIELDB 3 NU 1  VALUE 3.1
  FIELDC 1 BI    VALUE 127
  FIELDD 1 PD    VALUE -1
  FIELDE 6 PD 9  VALUE 1.123456789
```

Hexadecimal Literals

VISION:Results allows you to express values using hexadecimal notation. The literal must be coded on one line; it cannot be continued to another line.

You cannot use this type of literal in arithmetic operations.

Moving Hexadecimal Literals

You can move up to a 34-byte hexadecimal literal to a field. The receiving field must have a character or binary data type and must be the same size as represented by the literal. For example, FIELDG must be a 2-byte character field:

```
MOVE X'0102' TO FIELDG
```

Comparing Hexadecimal Literals

You can compare up to a 9-byte character field to a hexadecimal literal. The literal must appear to the right of the relational operator (EQ, NE, and so on). The field must be the same size as represented by the literal. For example, FIELDA must be a 3-byte character field:

```
IF FIELDA EQ X'FFFFFF' GOTO . . .
```

Assigning Values to Fields Using Hexadecimal Notation

If you are defining a character type field in an output FILE or WORKAREA statement, you can assign an initial value to that field using a hexadecimal literal. The field to be initialized can be up to 9 bytes in size. The field must be the same size as represented by the literal.

```
WORKAREA
 TABLE1 9 CH VALUE X'010203040506070809'
 SWITCH 1 CH VALUE X'FF'
```

Rules for Comparing Fields

The following compare (IF) combinations are possible.

Character/Character

Both fields must be less than 256 characters. When the two fields are not the same length, the shorter field is padded with trailing blanks prior to the compare. A logical compare is done on the fields.

Character/Numeric or Packed

The character field's length must be 16 bytes or less. If the character field is longer than 16 bytes, a validation error results. An algebraic compare is done; if the character field contains non-numeric data, a program check occurs.

Character/Binary

Both fields must be the same length. A logical compare is done.

Numeric/Numeric

An algebraic compare is done.

Numeric/Binary

An algebraic compare is done.

Numeric/Packed

An algebraic compare is done.

Packed/Packed

An algebraic compare is done.

Packed/Binary

An algebraic compare is done.

Binary/Binary

Both fields must be the same length. An algebraic compare is done.

Rules for Moving Fields

The following MOVE combinations are possible.

Character/Character

A 1- to 32767-byte character field can be moved to a 1- to 32767-byte character field. If you move a shorter field to a longer field, the shorter field is left-justified and padded with blanks. If you move a longer field to a shorter field, the longer field is left-justified and truncation occurs.

When padding occurs, the length of the receiving field is used to determine how many blanks are moved. No checking is done to ensure that an indexed MOVE does not exceed the boundaries of the receiving field. Use the LENGTH keyword to limit the amount of padding.

Character/Numeric

Treated as numeric to numeric.

Character/Packed

Treated as numeric to packed.

Character/Binary

Treated as numeric to binary.

Numeric/Character

A numeric field can be moved to a 1- to 16-byte character field. If the sign of the numeric field is C, the sign is changed to an F in the receiving field. If you move a shorter numeric field to the character field, it is right-aligned in the receiving field and high order zero filling occurs. If the receiving field is shorter, high order truncation occurs. If the numeric field is defined with decimals, they are omitted.

Numeric/Numeric

A numeric field can be moved to a numeric field. Decimal alignment occurs automatically. The sign of the receiving field is the same as that of the sending field. If the receiving field is not large enough to hold the sending field, high or low order truncation occurs.

Numeric/Packed

A numeric field can be moved to a packed field. Data conversion and decimal alignment occur automatically. If the packed field is not large enough to hold the value of the numeric field, high or low order truncation occurs.

Numeric/Binary

A numeric field can be moved to a binary field. Data conversion and decimal alignment occur automatically. If the sending field is unsigned, the receiving field is signed positive. If the binary field is not large enough to hold the value of the numeric field, high or low order truncation occurs and the sign can change.

Packed/Character

A packed field can be moved to a 1- to 16-byte character field. Data conversion occurs automatically. If the sign of the packed field is C, the sign is changed to F in the receiving field. If the character field is not large enough to hold the value of the packed field, high order truncation occurs. If the packed field is defined with decimals, they are truncated. If the character field is larger, high order zero filling occurs.

Packed/Numeric

A packed field can be moved to a numeric field. Data conversion and decimal alignment occur automatically. If the sending field is unsigned, the receiving field is signed positive (C). If the numeric field is not large enough to hold the value of the packed field, high or low order truncation occurs.

Packed/Packed

A packed field can be moved to a packed field. Decimal alignment occurs automatically. If the sending field is unsigned, the receiving field is signed positive (C). If the receiving field is not large enough to hold the value of the sending field, high or low order truncation occurs.

Packed/Binary

A packed field can be moved to a binary field. Data conversion and decimal alignment occur automatically. If the binary field is not large enough to hold the value of the packed field, high or low order truncation occurs and the sign can change.

Binary/Character

A binary field can be moved to a 1- to 16-byte character field. Data is converted to external decimal automatically. If the value is positive, the sign of the character field is set to F to ensure a printable field. If the character field is not large enough to hold the value of the binary field, high order truncation occurs. If the character field is larger, high order zero filling occurs. If the binary field is defined with decimals, they are omitted.

Binary/Numeric

A binary field can be moved to a numeric field. Data conversion and decimal alignment occur automatically. If the sending field is positive, the receiving field is signed positive (C). If the numeric field is not large enough to hold the value of the binary field, high or low order truncation occurs.

Binary/Packed

A binary field can be moved to a packed field. Data conversion and decimal alignment occur automatically. If the sending field is positive, the receiving field is signed positive (C). If the packed field is not large enough to hold the value of the binary field, high or low order truncation occurs.

Binary/Binary

A binary field can be moved to a binary field. Decimal alignment occurs automatically. If the binary receiving field is not large enough to hold the value of the sending field, high or low order truncation occurs and sign changes can take place.

Numeric and Packed Key Considerations

A positive sign for a numeric or packed field can be 'C' or 'F.' This can be a problem if you are searching for a particular record on an ISAM or VSAM file and the key you are using to search has a different sign than the key in the record. For example, you have set up a packed key in the ISAM or VSAM key area and the key has been converted from numeric to packed and a 'C' sign resulted. If the key in the record contains an 'F' sign, a status code of blank (no record found) is returned to the program after a random retrieval read. The same situation could occur with numeric fields.

To prevent or correct for unlike signs, the following approaches can be used.

Whenever possible, treat the keys as character data. The keys actually have a data type of packed, but they are defined as character so no sign change can occur. If the key has to be treated as packed for other purposes, it can be redefined.

```
FILE VSAMIN VSAM F 120 RANDOM SEARCHKEY
  STATUS STATUSFLAG KEYLEN 3
FILE TRANIN STATUS TRANEOF
  KEYIN 3 CH   ;  (key is actually
                   packed decimal)
PROCEDURE:
  IF TRANEOF EQ 'E' STOP ENDIF
  MOVE KEYIN TO SEARCHKEY   ;  (sign
  READ VSAMIN                  remains
                               unchanged)
IF STATUSFLAG NE 'Y'
  GOTO NOTFOUND ENDIF
```

Convert the packed key's sign from C to F using OR logic. For example, the sign of a 2-byte packed key is set to F.

```
COMBINE BITS X'000F' OR SRCHKEY
```

Convert the packed key's sign from F to C using AND logic. For example, the sign of a 2-byte packed key with an F sign is set to C.

```
COMBINE BITS X'FFFC' AND SRCHKEY
```

Edit Codes

The edit codes you can supply in your field definition to override the default edit format (P edit code) are as follows:

Edit Code	Description	Examples
E	Edit with zero suppression, commas, decimal insertion, and negative sign to the right of the field. Blank if zero.	001234.56 prints as 1,234.56 000000.00 prints as (blank)
Y	Leading '-'. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the number. Blank if zero.	-001234.56 prints as -1,234.56 000000.00 prints as (blank)
NE	Leading '-'. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the number. Blank if zero.	-001234.56 prints as -1,234.56 000000.00 prints as (blank)
DE	Leading '$'. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the right of the field. Float a currency sign (the default is $) to the left of the most significant digit. Blank if zero.	-001234.56 prints as $1,234.56- 000000.00 prints as (blank)
F	Leading '$'. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the right of the field. Float a currency sign to the left of the field. Blank if zero.	-001234.56 prints as $1,234.56- 000000.00 prints as (blank)
NDE	Leading '-$'. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the field. Float a currency sign to the direct left of the field. Blank if zero.	-001234.56 prints as -$1,234.56 000000.00 prints as (blank)
G	Leading '-$'. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the field. Float a currency sign to the direct left of the field. Blank if zero.	-001234.56 prints as -$1,234.56 000000.00 prints as (blank)
DNE	Leading '$-'. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the direct left of the field. Float a currency sign to the left of the negative sign. Blank if zero.	-001234.56 prints as $-1,234.56 00000000 prints as (blank)
K	Leading '$-'. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the direct left of the field. Float a currency sign to the left of the negative sign. Blank if zero.	-001234.56 prints as $-1,234.56 00000000 prints as (blank)

Edit Code	Description	Examples
A	Edit with zero suppression, commas, decimal insertion, and negative sign to the right of the field. Print decimal point and zeros to the right of decimal. If field has no decimal positions and has a zero value, nothing prints.	001234.56 prints as 1,234.56 0000000000 prints as (blank)
X	Leading ’-’. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the field. Print decimal point and zeros to the right of decimal. If field has no decimal positions and has a zero value, nothing prints.	001234.56 prints as 1,234.56 -001234.56 prints as -1,234.56 0000000.00 prints as .00 0000000000 prints as (blank)
NA	Leading ’-’. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the field. Print decimal point and zeros to the right of decimal. If field has no decimal positions and has a zero value, nothing prints.	001234.56 prints as 1,234.56 -001234.56 prints as -1,234.56 0000000.00 prints as .00 0000000000 prints as (blank)
DA	Leading ’$’. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the right of the field. Float a currency sign (the default is $) to the left of the most significant digit. Blank if zero.	001234.56 prints as $1,234.56 -001234.56 prints as $1,234.56- 0000000.00 prints as $.00 0000000000 prints as (blank)
H	Leading ’$’. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the right of the field. Float a currency sign (the default is $) to the left of the most significant digit. Blank if zero.	001234.56 prints as $1,234.56 -001234.56 prints as $1,234.56- 0000000.00 prints as $.00 0000000000 prints as (blank)
NDA	Leading ’-$’. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the field. Float a currency sign to the direct left of the field. Blank if zero.	001234.56 prints as $1,234.56 -001234.56 prints as -$1,234.56 0000000.00 prints as $.00 0000000000 prints as (blank)
J	Leading ’-$’. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the left of the field. Float a currency sign to the direct left of the field. Blank if zero.	001234.56 prints as $1,234.56 -001234.56 prints as -$1,234.56 0000000.00 prints as $.00 0000000000 prints as (blank)
DNA	Leading ’$-’. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, commas, decimal insertion, and negative sign to the direct left of the field. Float a currency sign to the left of the negative sign. Blank if zero.	001234.56 prints as $1,234.56 -001234.56 prints as $-1,234.56 0000.00 prints as $.00 0000000000 prints as (blank)

Edit Code	Description	Examples
L	Leading ’$-’. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, commas, decimal insertion, and negative sign to the direct left of the field. Float a currency sign to the left of the negative sign. Blank if zero.	001234.56 prints as $1,234.56 -001234.56 prints as $-1,234.56 0000.00 prints as $.00 0000000000 prints as (blank)
Z	Edit with zero suppression, decimal insertion, and negative sign to the right of the field, but no commas. Blank if zero.	001234.56 prints as 1234.56 00000000 prints as (blank)
U	Leading ’-’. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, decimal insertion, and negative sign to the left of the field, but no commas. Blank if zero.	00123456 prints as 1234.56 -00123456 prints as -1234.56 00000000 prints as (blank)
NZ	Leading ’-’. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, decimal insertion, and negative sign to the left of the field, but no commas. Blank if zero.	001234.56 prints as 1234.56 -001234.56 prints as -1234.56 00000000 prints as (blank)
B	Edit with zero suppression, decimal insertion, and negative sign to the right of the field, but no commas. Blank if zero. If zero value, print decimal point and zeros to the right of decimal. If field has no decimal positions and has a zero value, nothing prints.	001234.56 prints as 1234.56 000000.00 prints as .00 000000000 prints as (blank)
Q	Leading ’-’. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with zero suppression, decimal insertion, and negative sign to the left of the field, but no commas. Blank if zero. If zero value, print decimal point and zeros to the right of decimal. If field has no decimal positions and has a zero value, nothing prints.	001234.56 prints as 1234.56 -001234.56 prints as -1234.56 000000.00 prints as .00 000000000 prints as (blank)
NB	Leading ’-’. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with zero suppression, decimal insertion, and negative sign to the left of the field, but no commas. Blank if zero. If zero value, print decimal point and zeros to the right of decimal. If field has no decimal positions and has a zero value, nothing prints.	001234.56 prints as 1234.56 -001234.56 prints as -1234.56 000000.00 prints as .00 000000000 prints as (blank)
P	Default edit code. Edit with decimal insertion, and negative sign to the right of the field, but no commas. If the field has no decimal positions and has a zero value, zeros print. If it has a zero value, zeros print.	001234.56 prints as 001234.56 000000000 prints as 000000000
W	Leading ’-’. For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit with decimal insertion, and negative sign to the left of the field, but no commas. If the field has no decimal positions and has a zero value, zeros print. If it has a zero value, zeros print.	-001234.56 prints as -001234.56 0000000.00 prints as 0000000.00 000000000 prints as 000000000

Edit Code	Description	Examples
NP	Leading '-'. Applies only to data name definitions and LIST and EDIT statement overrides. Edit with decimal insertion and negative sign to the left of the field, but no commas. If the field has no decimal positions and has a zero value, zeros print. If it has a zero value, zeros print.	-001234.56 prints as -001234.56 0000000.00 prints as 0000000.00 000000000 prints as 000000000
D	Date edit of a 6-byte zoned decimal or 4-byte packed field.	120100 prints as 12/01/00
V	For use in data name definitions, LIST, EDIT, and LTD (Letterwriter) statements. Edit a date field containing a 4-digit year. The date field must be either an 8-byte zoned decimal or 5-byte packed field.	12012000 prints as 12/01/2000
D4	Applies only to data name definitions and LIST and EDIT statement overrides. Edit a date field containing a 4-digit year. The date field must be either an 8-byte zoned decimal or 5-byte packed field.	12012000 prints as 12/01/2000
S	Edit a social security number from a 9-byte numeric or 5-byte packed decimal field.	999999999 prints as 999-99-9999