Advantage™ VISION:Report®
Advantage™ VISION:Forms™
for z/OS

Getting Started Guide

16.1

Computer Associates®

RPGSZ161.PDF/D21-009-011

Contents

Chapter 4: Upgrading from Previous Releases 4-1

Chapter

1

Introduction

Advantage™ VISION:Report® for z/OS™ is an easy-to-use program development and report writing tool that you use to build and execute programs, minimizing the amount of programming time and effort required to fix a file, prepare a report, or generate test data.

- Advantage VISION:Report for z/OS will be referred to as VISION:Report.
- Advantage VISION:Forms™ for z/OS will be referred to as VISION:Forms. VISION:Forms is an ancillary product associated with VISION:Report.
- Advantage VISION:Report Interface to DB2 will be referred to as VISION:Report Interface to DB2. VISION:Report Interface to DB2 was previously known as VISION:Interface for DB2 with VISION:Report.
- The terms z/OS, OS/390®, and MVS® are used interchangeably throughout the Release 16.1 documentation.
- The terms DB2® and SQL/DS™ are used interchangeably throughout the Release 16.1 documentation. In general, the term DB2 is used in reference with z/OS and the term SQL/DS is used in reference with VSE.

There is virtually no limit to the types of reports VISION:Report can create. Preprinted forms, memos, letters, and labels are handled with ease. Arithmetic operations are provided with automatic decimal alignment and rounding, but you can override these features at any time. Normally, up to eight reports can be produced in one program. VISION:Report also supports European format for numeric printing.

VISION:Report uses a straightforward, self-documenting, COBOL-like language. First time users can be up and running after less than eight hours of training. Since VISION:Report requires you to define only the fields to be used, time-consuming COBOL file definition activity is eliminated.

VISION:Report also allows file definitions and program source statements to be saved in AllFusion™ CA-Librarian®, AllFusion™ CA-Panvalet®, PDS, or other source statement libraries to be shared by other programs, avoiding duplicate programming work. Character, zoned, decimal, packed, binary, and unconventional data types are supported. You can also validate programs prior to their execution.

One common language means that VISION:Report is easier to learn and manage than several individually sold utilities. Everything from file comparison and format conversions to testing, debugging, and prototyping can be accomplished using VISION:Report.

Using VISION:Report, you have the option to access, retrieve, update, and report data from the following file types and databases:

- VSAM
- Sequential
- Computer Associates® Advantage™ CA-IDMS®/DB
- IBM®
  - DB2® and SQL/DS™
  - IMS™ and DL/I

The CALL verb (also called command) is available to access callable databases. To enhance file access, VISION:Report also:

- Internally reads CA-Librarian, CA-Panvalet, and source statement libraries.
- Provides a CALL command for invoking subroutines written in Assembler or COBOL.
- Allows literals to be used as parameters in a CALL statement.

VISION:Report performs multiple functions with just one pass of the database. Conditional record selection, sorting, translating, matching, and merging can all occur with only one read of the master file. For further efficiency, VISION:Report also:

- Handles internal sorts in one statement.
- Releases and returns records from SORT in one easy step.
- Accepts compound conditional IF statements, allowing for complex data selection.
- Allows conditional branching.

VISION:Report operates under:

- z/OS, OS/390®, MVS®/SP/XA/ESA
- VSE/SP/ESA
- VM/CMS

Installing and Accessing the Documentation

Before you install Release 16.1 of VISION:Report, read this section.

CD-ROM Contents

The following is a list of documents by title and PDF file name:

- Advantage VISION:Report Advantage VISION:Forms Reference Guide
  RPREF161.PDF
- Advantage VISION:Report Advantage VISION:Forms for z/OS Installation Guide
  RPINZ161.PDF
- Advantage VISION:Report Advantage VISION:Forms for z/OS Getting Started Guide
  RPGSZ161.PDF
- Advantage VISION:Report Messages and Codes
  RPMSG160.PDF
- VISION:Forms Reference Guide
  FOREF307.PDF
- VISION:Forms Tutorial
  FOTUT307.PDF
- Advantage VISION:Report Interface to DB2 Reference Guide
  ILREF161.PDF
- Advantage VISION:Report Interface to DB2 for z/OS Installation Guide
  ILINZ161.PDF
- Adobe® Acrobat® Reader software and Help

About the Online Documentation

The CD-ROM contains the documentation for VISION:Report. The documents are in Adobe Acrobat Portable Document Format (PDF) and are designed for you to read online using the Acrobat Reader.

Each online document contains a table of contents and index.

Installing Online Documentation and the Acrobat Reader

You can install the online documentation on your local hard drive or on a network server. Alternatively, you can access the documentation directly from the CD-ROM.

If you do not have Acrobat Reader installed, you can install it from the CD-ROM.

To install the online documentation, the Acrobat Reader, or both

1. Close all application programs, including screen-saver software.
2. Insert the CD-ROM into the CD-ROM drive.
3. Click the Start menu and select Run.
4. In the Run dialog box, type: D:\Books\Setup.exe,
   where D: is the CD-ROM drive.

   To complete the Run dialog box and continue, click OK.
5. Follow the instructions. Computer Associates recommends that you install the online documentation in the default directory (C:\Program Files\CA\Advantage VISION_Report 16.1 zOS\Books\) or a directory of your choice (for example, C:\VISION_Report 16.1 zOS\Books\).

Viewing Online Documentation

Regardless of the location of the online documentation (on a local drive, a network server, or CD-ROM), you can view the online documentation using the following methods:

- Click the Start menu, point to Programs, point to Advantage VISION_Report 16.1 zOS, and click the document title.
- In Windows® Explorer, point to the directory on the hard drive where you installed the online documentation. Double-click the PDF file name.
- In Windows Explorer, point to the Books directory on the CD-ROM drive and double-click the PDF file name.

Using Adobe Acrobat Reader

Use Acrobat Reader to view the online documentation, adjust the size of the page, perform searches, and print a range of pages. For more information, use the Acrobat Help menu.

Contacting Computer Associates

For technical assistance with this product, contact Computer Associates Technical Support on the Internet at http://supportconnect.ca.com. Technical support is available 24 hours a day, 7 days a week.

Chapter

2

New Features and Functions

Enhancements for VISION:Report Release 16.1

The enhancements of VISION:Report for z/OS 16.1 are:

- 31-bit address mode
- Support for DB2 Versions 6 and 7
  - CALL command enhanced to access stored SQL procedures
  - IMS Attach is now available (in addition to the default CALL Attach and TSO Attach (batch and foreground))
- Language Environment (LE) support
- VSAM Enhancements
  - Two native VSAM commands: CLOSER and TCLOSE
  - Support VSAM Key Sequenced Data Set (KSDS) files with a Logical Record Length (LRECL) greater than 32K
  - 8-byte Relative Byte Address (RBA)
  - Support VSAM Linear Data Set (LDS) and Variable Relative Record Data Set (VRRDS) file types
  - Support up to a maximum of 256 concurrently open VSAM files
- Additional specification options on the SORT statement
- License Management Program (LMP)

31-bit Address Mode

VISION:Report supports either 24-bit or 31-bit addressing and can be link edited in either mode. This enhancement takes advantage of the increased storage provided when running applications in 31-bit mode as opposed to 24-bit mode.

Support for DB2 Versions 6 and 7

VISION:Report Interface to DB2 customers now have access to DB2 Version 6 and Version 7.

Enhanced CALL Command Accesses Stored Procedures

The VISION:Report Interface to DB2 component of the product has additional CALL command parameters to access stored (SQL) procedures. The new CALL command syntax is:

CALL {procname} {(parm1, parm2, …)} where:

- Parm1 is a stored procedure name or host_variable
- Parm2 is an optional stored procedure name or host_variable

Example:

```
CALL PROC1(:VAR,SALARY)
```

IMS Attach, TSO Attach, and CALL Attach

You do not need to change current programs that use CALL Attach. You can now use IMS Attach, TSO Attach (batch and foreground), or TSO Attach (default) by using JCL appropriate to the Attach facility.

Language Environment (LE) Support

Support has been added to establish a single common Language Environment (LE) runtime environment (process and enclave) for LE-conforming programs to execute multiple times when called from VISION:Report using the CALL statement.

VISION:Report examines all called program modules for LE signatures. If they are present in any program module, VISION:Report establishes a common LE runtime environment for all LE programs to execute in. This environment remains for the duration of the VISION:Report program execution.

VSAM Enhancements

Native VSAM Commands CLOSER and TCLOSE

Two new native VSAM commands have been added: CLOSER and TCLOSE, which allow native VSAM commands to be the same as calling QUIKVSAM:

- CLOSER calls independent VSAM services to close, then reopen a VSAM data set. VISION:Report does not release the data set.
- TCLOSE is a temporary close of the data set, but VSAM does not release control of it.

Support KSDS Files with LRECL Greater Than 32K

This release supports VSAM Key Sequenced Data Set (KSDS) files with a Logical Record Length (LRECL) greater than 32K. This requires the appropriate request for storage and the ability to reference beyond 32K in QUIKVSAM.

8-Byte Relative Byte Address (RBA)

The 4-byte Relative Byte Address (RBA) for a VSM feedback area is now 8-bytes to accommodate the extended format data sets.

Support LDS and VRRDS File Types

This release supports the VSAM file types, LDS (Linear Data Set) and VRRDS (Variable Relative Record Data Set).

- LDS supports Control Interval (CI) access.
- VRRDS supports numbered records with a maximum record size as part of the record.

Support 256 Concurrently Open VSAM Files

Increase table size to accommodate 256 VSAM files (that is, KSDS, RRDS, ESDS, and so on) being open at the same time, and being able to track them.

Additional Specification Options on the SORT Statement

There are new specification options on the SORT statement. They are in the format XXVVVVVVVV, where XX is the option and VVVVVVVV is the value. The format is similar to the record length specification RL00352. The additional specification options include:

- PG - Sort program name
- PR - Printing of message option and overrides SORTPRT option
- RT - Sort message routing
- ST - Storage value specified in bytes
- WF - Number of work files
- WN - Name of work files

License Management Program (LMP)

VISION:Report Release 16.1 uses the Computer Associates License Management Program (CA-LMP) licensing process instead of the previous VISION:Report licensing mechanism.

The License Management Program (CA-LMP) is an automated, standardized approach to tracking licensed Computer Associates products.

You will find the CA-LMP key on the CA-LMP key certificate that accompanies the product. For licensing information, contact Computer Associates Total License Care (TLC).

Chapter

3

Installation Overview

This chapter provides a brief overview of the installation procedure for VISION:Report in the z/OS environment. For detailed installation instructions, see the *Advantage VISION:Report Advantage VISION:Forms for z/OS Installation Guide*.

Installation Steps

If you have not done so already, please contact Computer Associates Total License Care (TLC) for proper CA-LMP keys. For additional information, see the *Advantage VISION:Report Advantage VISION:Forms for z/OS Installation Guide*.

To install VISION:Report for z/OS

For complete installation instructions, see the *Advantage VISION:Report Advantage VISION:Forms for z/OS Installation Guide.*

1 Copy the first file on the release tape to disk.

   This creates an installation control library that contains the members QJINSTL, QJTEST, QJTSTJCL, and so on.

2 Customize and run the member QJINSTL.

   The instructions in QJINSTL specify how to tailor the JCL for your installation. The QJINSTL JCL installs the base VISION:Report system.

3 Verify the installation by running a test program with the VISION:Report system.

4 Use the optional materials to implement additional VISION:Report functionality .

Chapter

4

Upgrading from Previous Releases

This chapter is designed to assist you in migrating from previous releases of VISION:Report. Carefully review the following sections for situations that could apply to your installation.

Command and Keyword Syntax

With the exception of features and options added to the VISION:Report language, the original syntax of this product has not changed.

If your VISION:Report source programs follow the syntax as described in the *Advantage VISION:Report Advantage VISION:Forms Reference Guide*, you should not experience any difficulty in migrating to the new release of VISION:Report, especially if you have been implementing previous releases in a timely manner.

If you have previously implemented Release 15.0, 15.1, or 16.0 of VISION:Report, most of the common syntax problems should have already been resolved. To verify your syntax, check your current production VISION:Report programs prior to the implementation of these releases. Simply use the EDIT=YES option. EDIT=YES performs syntax checking and creates a proof copy of report output without executing the program.

To easily test your most critical production VISION:Report programs in one execution, generate your QJOPTION block with all your normal options and EDIT=YES. Assuming your VISION:Report source programs are in a partitioned data set or library, simply set up a job stream to find out whether your VISION:Report programs compile correctly. Then, before using the new version of VISION:Report in a production environment, remove the EDIT=YES option and run your most critical production programs to ensure that they perform as intended.

If you receive diagnostic errors in programs that worked in previous releases, most likely the syntax was incorrect. The most common problems include commands or parameters being misspelled, optional parameters not in the proper order, and non-existent optional parameters at the end of an otherwise valid statement. All keyword validation has been strengthened and all keywords or parameters must be spelled correctly or completely in order to pass diagnostics.

Compatibility Mode

If you need to run VISION:Report in a compatibility mode with previous releases, there is a member called ZAPCOMP that applies all of the necessary Program Compatibility Patches (PCPs) to make this release compatible with the previous release. You should carefully review all the PCPs, to determine which applies to your company, before running the job.

Note: Evaluate ZAPCOMP members carefully.

We recommend that you do not modify the original modules, but run ZAPCOMP against a separate library, such as PCPLIB.

The member ZAPCOMP is in the SAMPLIB under MVS, and in the source sublibrary for VSE.

Common Syntax Problems

The following exhibits some of the more common syntax problems. The cause of the error is in **bold** print.

- The SORT command ON noise word was left out.

```
SORT flddefl RLnnnn etc              should be:

SORT flddefl RLnnnn ON etc.
```

- The MOVE command was misspelled.

```
MOV A TO B                           should be:

MOVE A TO B
```

- The mandatory keyword TO was left out.

```
MOVE INF1-2 PRT1                     should be:

MOVE INF1-2 TO PRT1
```

- Invalid keyword or parameters were entered, possibly in the wrong sequence. The MOVE command can be used with either an edit code or a variable length operand, but not both at the same time.

```
MOVE A TO B 0C X                     should be:

MOVE A TO B 0C                       (MOVE with Edit Code) or
MOVE A TO B X                        (MOVE Variable Length)
```

- An invalid keyword was used in an improper sequence or place; in this case, EQU cannot be substituted for EQ.

```
IF A EQU B                           should be:

IF A EQ B
```

- Invalid keyword or parameters were entered at the end of an otherwise valid statement. There is no THRU statement number option.

```
GET INF ATEND 700 THRU 900          should be:

GET INF ATEND 700
```

- Invalid keyword or parameters were entered. There is no GIVING optional parameter to the ADD verb.

```
ADD A TO B GIVING C                 should be:

ADD A TO B                          (results were never put in C)
```

- Initialization constants must follow edit specification. Some earlier releases did not enforce this coding rule completely.

```
EQU fldname WST1-10 ZEROS 0C        should be:

EQU fldname WST1-10 0C ZEROS
```

- One or more spaces did not follow the statement before the optional comments.

```
MOVE A TO B/*                       should be:

MOVE A TO B  /*
```

- The MULT and DIVD verb did not have the required BY and GIVING keywords.

```
DIVD A B C                          should be:

DIVD A BY B GIVING C
```

- The Option R for rounding in the MULT and DIVD verb cannot be separated by a space from flddef3.

```
MULT flddef1 0D  BY  flddef2 0D  GIVING  flddef3 0D  R
                                    should be:

MULT flddef1 0D  BY  flddef2 0D  GIVING  flddef3 0DR
```

- You cannot add fields that contain non-leading blanks or spaces.

```
ADD INF1 TO WST1                    where INF is "1 234"
```

- Duplicate sequence numbers are not allowed, even with OPTION SEQCHK=NO.

```
190 MOVE A TO B

190 MOVE C TO D                     should be:
200 MOVE C TO D
```

- Previous releases checked only the first letter of the required keyword.

```
TRACE Lxxxxx or  TRACE Axxxxx or  TRACE Oxxxxx
                                    should be:

TRACE LAST50  or  TRACE ALL   or  TRACE OFF
```

- Numeric fields should be initialized or values moved in prior to any arithmetic operations. EBCDIC fields should be initialized to zeros (X'F0F0....'). Blanks are accepted, but not recommended. Packed fields should be initialized to packed zeros (X'....000C'). Binary fields should be initialized to binary zeros (X'0000'). The most convenient way to initialize storage at the beginning of a job is with an EQUate statement.

```
EQU PK-FIELD WST1-3-P ZERO

ADD P'1'  TO  PK-FIELD
```

  It should not be assumed that an area of storage is initialized properly for arithmetic operations. Assume, in the following example, that working storage area, WST1-3, has not been properly initialized (WST is initialized to low values at the beginning of a job unless set to another value by an EQUate statement).

```
ADD P'1' TO WST1-3-P                should be:

MOVE ZEROS TO WST1-3-P
...
ADD P'1' TO WST1-3-P
```

  When moving packed and binary fields to PTR or PTX, and editing is to be performed, you must code the type of editing required. The number of decimal points wanted (0 through 9) is a requirement. By using the number of decimal editing, all positions from the packed or binary field will be printed. In the following example, assume that 12345 is in WST1-3-P:

```
MOVE WST1-3-P TO PRT1 0             Results= 12345

MOVE WST1-3-P TO PRT1 0C            Results= 12,345
MOVE WST1-3-P TO PRT1               Results= 5  (no editing performed)
```

  When specifying the number of decimal points, you cannot use 'D' on the MOVE statement. In this example, again assume that 12345 is in WST1-3-P:

```
MOVE WST1-3-P TO PRT1 0D            Results= error
```

  Packed fields in the prior releases always put a blank edit fill character in the first print position you specified when moving packed fields to PRT. Again, in this example, assume that 12345 is in WST1-3-P:

```
MOVE WST1-3-P TO PRT1 0             Results= depend on release
```

  The results in prior releases would have a blank in print position 1 and 12345 in positions 2 through 6, which was invalid and should have been 12345 in positions 1 through 5.

- Only reference an offset in a header that is smaller than you are allowed in the PRTSIZE option. Assume that PRTSIZE is set to 121.

```
MOVE HDA1-133 TO OFA1               should be:

MOVE HDA1-120 TO OFA1               /*  HDA121 is invalid
```

- Using the OPTION SEQCHK=YES, all statements other than the OPTION and VSE I/O statements must have sequence numbers coded, including EQU statements. Assume OPTION SEQCHK=YES was specified:

```
EQU ABC WST1-8                    should be:
010   EQU ABC WST1-8
```

  Because most source programs are stored on disk, the usage of SEQCHK=YES is not needed in most cases.

- Duplicate EQU names, unless used in redefining an area previously defined (by the same name), are not allowed.

- On the BREAK statement, the obsolete SB and SA parameters M, N, and P were ignored in previous releases, but not flagged as errors. It can only be 0, 1, or 2.

```
BREAK 1 INF...  SB N SA M          should be:
BREAK 1 INF...  SB 1 SA 0
```

- When specifying number of decimal points, 'D' cannot be used on the MOVE statement. nD is not a valid option for an edit code; only C, E, or N is supported. An edit code other than C, E, or N is accepted only if defined in the user edit macro table, and it is specified without a numeric prefix. Assume that WST1-3-P contains 12345:

```
MOVE WST1-3-P TO PRT1 0D          should be:
MOVE WST1-3-P TO PRT1 0C
```

- On the RETURN SORTED statement, 'INTO' is the correct syntax, not 'TO'.

```
RETURN SORTED TO WST1             should be:
RETURN SORTED INTO WST1
```

- When using STMTEND=nn option, you should not have any data in the next column after the column a statement ends in, as the next position immediately to the right (nn plus 1) designates whether or not the current statement is to be continued to the next statement. This is true, even if it is part of a comment (/*). Ensure that comments terminate before the STMTEND column. Assume that STMTEND=71:

```
                                                          COL72
                                                            ↓
MOVE ABC TO  XYZ                                  /* MOVE AR-ACCOUNT-CODE
```

  The preceding statement fails with an error because the 'E' in 'CODE' was in the continuation column 72 and there was no continuation statement on the next line.

```
MOVE ABC TO  XYZ     /* MOVE AR-ACCOUNT-CODE
```

  The preceding statement is correct because it ends before the specified STMEND column.

- When using the ACCUM command into a CTX counter, be careful that the number of bytes specified elsewhere is addressed correctly by using the low order.

```
REPORT field 1  CTA1-2-P            should be:

REPORT field 1  CTA7-8-P
. . .
. . .
ACCUM field 1   IN A 2 BYTE CTA
```

- The HDR statement must begin in positions 5, 6, or 7, allowing possible sequence numbers to be in positions 1 through 4. The positioning of the HDR statement must follow the syntax:

```
.+...10....+

HDR etc                          should be:
   HDR etc
```

If you receive a diagnostic error that you did not receive in a previous release, verify the syntax in the *Advantage VISION:Report Advantage VISION:Forms Reference Guide*. If, after changing the statement to conform with the syntax, it still does not compile without diagnostics, then contact Computer Associates Technical Support.

Memory Allocation

To increase the number of available input and output files, VISION:Report now allocates and utilizes memory (storage) and input/output buffers more efficiently. Specifically, VISION:Report obtains only the amount of memory requested, directly or indirectly.

If programs written under previous releases referenced memory beyond what was defined, they could abnormally terminate, usually with an operation or protection exception. This generally occurs when storage is overlaid by an incorrect memory specification.

The following is a list of situations that can cause memory allocation errors.

- Loading a table and exceeding the space requested or searching through a table and exceeding its limit. When loading or searching through tables, a check must be made to ensure that its limit is not exceeded. SET PTR instructions should be examined to ensure that they do not exceed the table's specified maximum entries.
- Specifying a record length and/or block size smaller than the file characteristics and then addressing or moving data to an area beyond what was specified. Because I/O buffers for the exact record size are requested, this could cause a protection exception. Sometimes this can be subtle, such as a VSAM file defined outside the job stream or a variable length file exceeding its specified maximum record length or block size.

- Referencing a field in a file area before opening the file. If you coded an explicit OPEN file statement, this can cause an ABEND by deferring the allocation of memory for the file area. Code the OPEN statement before referencing the field.
- Moving a field to another area that is smaller or beyond the maximum limits. This could cause memory to be overlaid or cause an ABEND. For example, if a SAVAREA was defined as 500 bytes maximum and you coded MOVE WST1-10 TO SAV495, the memory after SAV500 would be overlaid, resulting in unpredictable results.
- Migrating to another release of an operating system. This could cause an ABEND, for the same reasons stated above. The memory that was obtained from the new operating system can be in a different area or more restrictive; violations will cause an ABEND.

Printer Output

If programs written under previous releases were adjusted to accommodate these inconsistencies, you can experience spacing problems in reports that are sensitive to exact alignment (for example, preprinted forms or labels). If your installation creates these types of reports, before using this new release of VISION:Report in a production environment, run these jobs to ensure that they perform as intended.

Index

3

8

A

C

D

E

F

H

I

K

L

M

N

O

P

Q

R

S

T

V

W

Z