
SirZap User's Guide and Reference

Load Module Maintenance Facility



Sirius Software, Inc.
875 Massachusetts Avenue, Suite 21
Cambridge, MA 02139

Telephone: (617) 876-6677

FAX: (617) 234-1200

E-mail: support@sirius-software.com

World Wide Web: <http://sirius-software.com>

March 2, 2009

© 2009 Sirius Software, Inc.

Proprietary Notices

SirZap is a proprietary product of Sirius Software, Inc.:

**Sirius Software, Inc.
875 Massachusetts Avenue, Suite 21
Cambridge, Massachusetts 02139
USA**

Model 204™ is a proprietary product of Computer Corporation of America:

**Computer Corporation of America
200 West St.
3rd Floor West
Waltham, MA 02451
USA**

Contents

Proprietary Notices	iii
Contents	v
Summary of Changes	vii
SirZap Version 1.6	vii
Chapter 1: Introduction	1
Versions	1
Chapter 2: <i>SirZap</i> Features	3
Chapter 3: Differences between <i>SirZap</i> and IMASPZAP	5
Chapter 4: Differences between <i>SirZap</i> and CMS ZAP	7
Chapter 5: CMS Installation	9
Chapter 6: MVS Installation	11
Chapter 7: <i>SirZap</i> Options	13
BACKOUT	13
UPPER	13
REPORT	13
TXTLIB txtlib	14
EXTRACT filetype	14
Chapter 8: <i>SirZap</i> Control Statements	15
VER	15
REP	16
NAME	16
DUMP or DUMPT	17
* !VERIFY	17
* !NOVERIFY	18
Chapter 9: Using <i>SirZap</i> under MVS	19
<i>SirZap</i> DD statements	19
SirZap sample JCL	19

Chapter 10: Using <i>SirZap</i> under CMS	21
Using <i>SirZap</i> with CMS text libraries	21
Chapter 11: Messages	23
Chapter 12: <i>SirZap</i> Return Codes	29
Appendix A: Deprecated Options	31
NOVERIFY	31
VERIFY	31
Appendix B: Date Processing	33
Index	35

Summary of Changes

This section describes significant changes to the documentation. In most cases these changes correspond to enhancements made to the underlying product.

SirZap Version 1.6

The base version of this manual was converted for version 1.6.

CHAPTER 1 ***Introduction***

SirZap is a load module maintenance facility, similar to MVS IMASPZAP or the CMS ZAP command. *SirZap* modifies load modules (or for CMS, both load modules and text libraries) based on information from control statements supplied by the user. *SirZap* is not a plug compatible replacement for the IBM ZAP utilities, but it provides functional and performance enhancements for load module maintenance.

SirZap is distributed as part of a software maintenance agreement between Sirius Software Inc. and its customers. *SirZap* is provided for the purpose of applying maintenance to Sirius Software's products, but the customer need not limit use of *SirZap* to Sirius' products.

1.1 Versions

This document, the ***SirZap User's Guide and Reference***, assumes that a site is running *SirZap* version 1.6 or later. Any documented feature or facility that requires a later version of *SirZap* will be clearly marked to indicate this. If a feature or parameter is not indicated as requiring any specific version of *SirZap*, it can be assumed that it is available in all versions covered by this document; that is, versions 1.6 and later of *SirZap*.

CHAPTER 2 *SirZap Features*

SirZap can save significant amounts of elapsed time and CPU for long ZAP input streams. This is because *SirZap* applies all ZAPs in the input stream before writing a module back to disk. IMASPZAP and CMS ZAP both write parts of the load module back to disk each time a new NAME statement is encountered, reducing performance significantly.

SirZap detects and reports ZAPs that are already applied. Using IMASPZAP or CMS ZAP, the user must inspect long CSECT dumps produced by the ZAP utility in order to determine if the ZAP is already applied, or simply in error.

SirZap can “back out” or remove ZAPs without requiring modification to the ZAP control statements. By using the BACKOUT option, ZAPs can be removed from the load module and replaced by the original Verify data.

SirZap limits error reporting to the specific statements in error, and it reports only the information necessary to diagnose and correct the error.

SirZap only writes modified load modules. If the ZAPs are already completely applied, *SirZap* does not write the load module back to disk.

SirZap enforces stricter rules on ZAP input statements. Verify statements are required for all replaced bytes. This ensures that the load module contents are known before they are modified, and that a ZAP can be removed or “backed out” if necessary. You can override this feature, however, using the * **!NOVERIFY** statement.

Differences between SirZap and IMASPZAP

- *SirZap* requires that every byte in the load module that is replaced is also verified, unless the * **!NOVERIFY** statement is used.
- *SirZap* only modifies direct access load libraries. *SirZap* does not modify any other types of direct access datasets.
- The following SPZAP statements are not supported but are ignored when encountered in the input stream: SETSSI, IDRDATA, and CHECKSUM.
- The following IMASPZAP statements are not supported and will cause *SirZap* to terminate when they are encountered: CONSOLE, ABSDUMP, ABSDUMPT, and RECORD.

Differences between SirZap and CMS ZAP

- *SirZap* requires that every byte in the load module that is replaced is also verified, unless the * **!NOVERIFY** statement is used.
- *SirZap* does not support CMS LOADLIBs.
- The following ZAP statements are not supported but are ignored when encountered in the input stream: COMMENT, and LOG.

SirZap is distributed as a single load module in VMFPLC2 format. A set of object decks is also included in case linkedit is required. (This may be necessary for some older CMS versions).

SirZap will be installed when you install the other Sirius products you ordered. Usually, no additional steps are required.

The *SirZap* load module is linked AMODE=31. If *SirZap* must be relinked on your system, use the following commands:

```
LOAD SIRZAP (RESET SIRZAP RMODE 24 AMODE 31
GENMOD SIRZAP (FROM APPL
```

Note: You can omit the RMODE and AMODE parameters if you are running in a non-XA environment.

CHAPTER 6 ***MVS Installation***

The SIRZAP load module and library are distributed in IEBCOPY unload format and are merged with any other Sirius products you have purchased.

SirZap will be installed when you install the other Sirius products you ordered. Usually, no additional steps are required.

SIRZAP can be copied to SYS1.LINKLIB, or SYS1.LPALIB and is reentrant.

CHAPTER 7 *SirZap Options*

MVS users supply *SirZap* options in the PARM parameter of the EXEC job control statement. For example, to force uppercase output and to backout ZAPs, use the following EXEC statement:

```
//SIRZAP EXEC SIRZAP,PARM='UPPER,BACKOUT'
```

CMS users specify options following a left parenthesis on the SIRZAP command line. For example, to force uppercase output and to backout the ZAPs in the file “LOCAL ZAP A1”, specify the following:

```
SIRZAP LOCAL ( BACKOUT UPPER
```

7.1 BACKOUT

The BACKOUT option removes a ZAP without requiring any modification to the ZAP input statements. BACKOUT reverses the effect of a ZAP by replacing the original Verify data in the module. If the ZAPs were applied with IMASPZAP, CMS ZAP, or with *SirZap* using the * **!NOVERIFY** statement, *SirZap* may not be able to perform a backout, because all the original Verify data may not be present.

7.2 UPPER

The UPPER option forces all *SirZap* output to uppercase. This is usually unnecessary, except on certain Japanese terminals and when sending output to uppercase-only printers. UPPER is the default setting for Hitachi and Fujitsu systems.

7.3 REPORT

The REPORT option indicates that *SirZap* will run normally except that it will not update any load modules. This is useful in a situation in which you have a production load module you don't want to modify, but need to determine whether or not a particular set of ZAPs are applied.

7.4 TXTLIB txtlib

The TXTLIB option is valid under CMS only. TXTLIB specifies that the ZAPs should be applied to a CMS text library instead of a load module.

You must provide the name of the text library following the TXTLIB parameter. The first parameter of the name statement (the module name) is ignored when using the TXTLIB option. Only the CSECT name is used for text libraries. No load modules are inspected or modified when the TXTLIB option is specified.

7.5 EXTRACT filetype

The EXTRACT option allows you to extract individual object files from a CMS text library. EXTRACT can only be used when the TXTLIB option is also specified.

EXTRACT requires a parameter that specifies the file type of the object files it creates. One object file is created for each member of the text library. The text library member name is used as the CMS file name.

SirZap Control Statements

SirZap control statements appear in columns 1 to 72 in fixed length, 80 byte records. Statements that begin with an asterisk are treated as comments and are ignored, except for the two special inline parameter statements * **!VERIFY** and * **!NOVERIFY**. Control statements can begin in any column, but line continuation is not allowed.

8.1 VER

The VER statement identifies a range of bytes to be matched with the load module contents. VER statements must follow a NAME statement. A VER statement must be present for each range of bytes that will be replaced in the load module. This is to ensure that the ZAP can be backed out later if necessary. The format of the VER statement is:

```
VER offset verify-values comments
```

- offset** The hexadecimal displacement from the start of the CSECT specified in the previous NAME statement, or from the start of the load module if no CSECT was specified.
- verify-values** Pairs of hexadecimal characters optionally separated by commas. The values are the expected contents of the load module at the specified offset. Uppercase or lowercase hexadecimal data are acceptable.
- comments** Any characters after the first blank after the last *verify-values* character are ignored by *SirZap*.

The following are all valid VER statements:

```
VER 00 010203,040506 Verify bytes at offset 0.
```

```
VER 01ca 47f0,c0e9
```

```
VER 23EC D20310004000,90ECD000,07FC
```

8.2 REP

The REP statement identifies a range of bytes to replace the current contents of a load module. The REP statement must be preceded by VER statements that cover the entire range of bytes that will be replaced. The format of the REP statement is:

```
REP offset replace-value comments
```

- offset** The hexadecimal displacement from the start of the CSECT in the previous NAME statement.
- replace-value** Pairs of hexadecimal characters optionally separated by commas. The values will replace the contents of the load module at the specified offset. Uppercase or lowercase hexadecimal data are acceptable.
- comments** Any characters after the first blank after the last *replace-value* character are ignored by *SirZap*.

The following are all valid REP statements:

```
REP 00 FFFFFFF,FFFFFF Replace bytes at offset 0.
```

```
REP 01ca 47f0,c0e8
```

```
REP 23EC D203101C4000,90ECD008,07FE
```

8.3 NAME

The NAME statement identifies a load module, and optionally a CSECT that defines the context for the VER and REP statements that follow. **NAMEX** is a synonym for NAME which allows compatibility with the Fujitsu OSIV ZAP utility.

The format of the NAME statement is:

```
NAME module [csect]
```

Where:

- module** The name of the load module. Under CMS the filetype must be "MODULE". Under MVS this is the name of a PDS member in the load library.
- csect** The name of the control section in the load module. The csect name is optional. If the csect name is missing, subsequent verify and replace displacements are calculated from offset 0 in the load module.

8.4 DUMP or DUMPT

The DUMP statement prints all or part of the contents of a load module or CMS text library to SYSPRINT. Specifying DUMPT has the same effect as DUMP.

The format of the DUMP or DUMPT statement is:

```
DUMP[T] name [csect [offset [length]]]
```

Where:

- name** The name of the load module or text library. Under CMS, specify the filename (only) of a file whose filetype must be “MODULE” or “TXTLIB”. Under MVS, specify the name of a PDS member in the load library.
- csect** The name of the control section. The csect name is optional. If the csect clause is missing, the entire load module or text library contents are dumped.
- offset** The hexadecimal starting offset within the csect. This is optional and may only be specified when a csect name is specified. Use the offset parameter when you want to dump only a part of a csect. If the offset is omitted, it defaults to 0.
- length** The number of bytes (in hexadecimal) of the csect to dump. This is an optional parameter and may only be specified if a csect and starting offset are specified.

8.5 * !VERIFY

The special statement * **!VERIFY** (and * **!NOVERIFY**) allows you to control whether individual ZAPs are verified before being applied.

The main purpose of these statements is to override whether the current portion of the zap requires verify data. This can be useful when a single ZAP in an input stream containing many ZAPs has no verify data.

The asterisk and space are necessary so that other ZAP utilities treat the statements as comments.

Use * **!VERIFY** anywhere within the ZAP input stream. ZAPs following the * **!VERIFY** statement will be verified (the VER data must match the module contents) before the ZAP will be applied.

Since the * **!NOVERIFY** statement is new in *SirZap* version 1.6, you may have *SirZap* JCL or EXECs that use the NOVERIFY program option to accomplish this purpose, but at a dangerous job step level. We strongly recommend you do not use the NOVERIFY program option, but use the * **!NOVERIFY** statement instead, and only when needed.

8.6 * **!NOVERIFY**

The special inline statement * **!NOVERIFY** (and * **!VERIFY**) allows you control whether individual ZAPs are verified before being applied. The main purpose of these statements is to override whether the current portion of the zap requires verify data. This can be useful when a single ZAP in an input stream containing many ZAPs has no verify data or the verify data is known to be incorrect.

Use * **!NOVERIFY** anywhere within the ZAP input stream. ZAPs following the * **!NOVERIFY** statement will not be verified (the VER data may be missing, or not match the module contents) before the ZAP will be applied.

Since the * **!NOVERIFY** statement is new in *SirZap* version 1.6, you may have *SirZap* JCL or EXECs which use the NOVERIFY program option to accomplish this purpose, but at a dangerous job step level. We strongly recommend you do not use the NOVERIFY program option, but use the * **!NOVERIFY** statement instead, and only when needed.

Using SirZap under MVS

Job Control requirements for *SirZap* are the same as for IMASPZAP. Most *SirZap* storage is allocate above the 16 megabyte line, so the default region size should be sufficient.

9.1 *SirZap* DD statements

SYSLIB is required and points to the load library PDS (Partitioned Data Set) containing the load module you want to ZAP.

SYSIN is required and contains the ZAP control statements and must have the DCB attributes of RECFM=F or FB, and LRECL=80.

SYSPRINT is the report data set and is required.

9.2 *SirZap* sample JCL

```
//SIRZAP  JOB  CLASS=A,MSGCLASS=A
//ZAP    EXEC SIRZAP,PARM='sirzap parms'
//STEPLIB DD  DSN=SIRZAP.V106.LOAD,DISP=SHR
//SYSLIB DD  DSN=loadlib,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSIN  DD  *
zap control statements
/*
```

CHAPTER 10 *Using SirZap under CMS*

The format of the SIRZAP command is:

```
SIRZAP fn [ft [fm]] ( options
```

SirZap is invoked as a CMS command. Simply enter `SIRZAP` followed by the name of the input file. The default filetype is “ZAP”, and the default filemode is “*”.

Options are supplied in any order following the left parenthesis. See “[SirZap Options](#)” on page 13 for a list of options.

SirZap writes the report and any modified load modules to the disk accessed as the “A” disk. Make sure that the “A” disk is accessed in Read/Write mode.

10.1 Using *SirZap* with CMS text libraries

When *SirZap* processes a text library, the resulting text library or object files usually contain fewer records than the original. No information is lost and the resulting text library or object file is logically the same as the original. Text records in the object file are always written with the maximum amount of object code per record, so often fewer records are required than are typically written by the compiler or assembler that originally generated the object. This object file 'normalization' is similar to how a linkage editor processes an object file. *SirZap* does this so that ZAPs can be applied more efficiently.

CHAPTER 11 *Messages***ZAP0002** **Missing input filename**

SirZap requires an input filename under CMS.

ZAP0003 **Invalid filemode *mode***

An invalid CMS filemode was supplied.

ZAP0004 **Invalid parameter *parm***

An invalid parameter was supplied. *SirZap* does not support IMASPZAP or CMS ZAP options.

ZAP0005 **Invalid input filename *filename***

An invalid CMS filename was supplied to *SirZap*. Check to make sure the filename was valid.

ZAP0006 **Invalid syntax for *type* in statement *N***

The *SirZap* control statement was coded incorrectly. For NAME statements, either the module or CSECT name is longer than 8 characters. For REP and VER statements, you may have supplied invalid hexadecimal data, or an invalid number of hexadecimal characters.

ZAP0007 **NAME statement must precede VER/REP in statement *N***

The verify or replace statement must be preceded by a NAME statement to identify the module and CSECT.

ZAP0008 **Unknown control word *word* in statement *N***

The statement contains an unrecognized control word. *SirZap* terminates without applying any ZAPs.

ZAP0009 **Can't open *filename***

SirZap was unable to locate the input file.

ZAP0010 **Insufficient storage for SIRZAP**

SirZap requires a larger region (or virtual machine) to execute. *SirZap* requires more storage than either IMASPZAP or CMS ZAP.

ZAP0011 Error writing *filename*, RC=*N*

SirZap was unable to write the new load module. Check CMS FSWRITE return codes for the meaning of RC. A common cause is a disk full condition, indicated by a return code of 13.

ZAP0012 Offset *offset* in statement *N* exceeds maximum *maxoffset* for CSECT *csect*

A CSECT offset given in a VER statement was greater than the length of the CSECT. The load module probably does not contain the correct version of the CSECT you want to modify. Compare the CSECT length listed in the linkedit with the assembly listing to determine the source of the problem.

ZAP0013 Statement *N* references *hex length* byte undefined area at offset *hex offset* CSECT *csect*.

The ZAP references a location in a CSECT which is not defined. In assembly language terms, the area is defined with DS (define storage) instead of DC (define constant) statements. The area is not represented in the load module or the text library, and so cannot be zapped.

ZAP0014 Verify data in statement *X* does not match verify data in statement *Y*

Statements X and Y both verify at least one of the same bytes in a load module, but disagree on the value of some of the bytes. No replace statement between these two statements modifies the bytes, so at least one of the statements is in error.

ZAP0015 Verify data in statement *X* does not match replace data in statement *Y*

Statement Y will modify bytes in a load module which are later verified by statement X. These two values must match. Change the verify data in statement X to match the replace data in statement Y.

ZAP0016 Replace data in statement *M* was not verified

No verify statement was coded for the replace in statement N. All replace statements must have a corresponding verify statement.

ZAP0018 Summary for module *module*:

This is an informational message that precedes the summary report for each module.

ZAP0019 Can't open module *module*

The indicated module could not be opened. It is probably not on any accessed disk (CMS), or not in SYSLIB (MVS).

ZAP0021 Incorrect verify data in statement *N* at *csect* offset

Neither the verify data nor any replace data in the input stream match the module contents at the location indicated by *csect* and hexadecimal *offset*. This message is followed by ZAP0022 which prints the verify data, and by message ZAP0023, which prints the module contents. The module will not be written back to disk.

ZAP0022 ZAP *data=hexdata*

This informational message shows the contents of the verify data in the statement referred to by message ZAP0021.

ZAP0023 Module *data=hexdata*

This informational message shows the contents of the module data at the location specified by message ZAP0021. *SirZap* prints the contents of the module at the time the error occurred. Any previous replace operations will be reflected in the printed values, thus they may not match the disk contents.

ZAP0024 ZAPs in statements *N-N* applied

An informational message indicating that the ZAPs were applied successfully. The modified load module will be written back to disk.

ZAP0025 ZAPs in statements *N-N* already applied

An informational message indicating that the ZAPs in the indicated range were already applied. If all ZAPs were already applied, the load module will not be written back to disk.

ZAP0026 ZAPs in statements *N-N* backed out

This informational message indicates that a backout operation was attempted and the ZAPs in the indicated range were replaced by the corresponding verify values. The modified load module will be written back to disk.

ZAP0027 ZAPs in statements *N-N* already backed out

This informational message indicates that a backout operation was attempted and the ZAPs in the indicated range had not been applied, and thus were not backed out. If no ZAPs were backed out, the load module will not be written back to disk.

ZAP0028 ZAPs in statements *N-N* are circular

This message indicates that the ZAPs in the named range do not change the load module. The initial verify value, the ending replace value, and the module contents match exactly. The ZAPs could be removed from the input stream.

ZAP0029 ZAPs in statements *N-N* ignored, CSECT *csect* is missing

The ZAPs in the indicated range will not be applied to the load module because the indicated CSECT is missing. The return code is set to 4, and *SirZap* operation continues.

ZAP0030 End of *SirZap*

This informational message indicates that *SirZap* has completed operation.

ZAP0031 Invalid option *option*

The option on the command line is invalid. *SirZap* terminates with a return code 8.

ZAP0032 ZAPs in statements *N-N* are not applied

The ZAPs in the statements indicated are not applied to the module. This message only appears when the REPORT option has been specified. No changes are made to the load module.

ZAP0033 Not all bytes were verified, backout may be incomplete

The ZAPs were backed out of the load module, but some bytes that were replaced did not have associated verify statements, so the entire ZAP cannot be backed out. The ZAP may have been applied with another ZAP utility, or possibly with *SirZap* using the * **!NOVERIFY** statement.

ZAP0034 TXTLIB parameter is valid for CMS only

The TXTLIB option cannot be used under MVS or VSE. *SirZap* can only modify load modules under these operating systems.

ZAP0035 Non-EDF TXTLIBs are not supported

The CMS text library is in an old (non-EDF) format. *SirZap* can not process non-EDF text libraries.

ZAP0036 Record *N* invalid in *fn ft fm*

A record in the text library is invalid. The text library may be damaged, or in an unsupported format. *SirZap* can not process the text library and terminates.

ZAP0038 *fn* TXTLIB is not composed of fixed, 80 byte records.

The file indicated is not a valid text library. *SirZap* cannot process the file and terminates immediately.

ZAP0039 Error writing *fn ft fm*

SirZap failed while writing the indicated CMS file. Check to make sure the disk is accessed in read/write mode and that it has sufficient free space for the file.

ZAP0042 DUMP of *type name*, CSECT *csect*, Length *length*

This is an informational message produced when a CSECT from a load module or text library is processed with a DUMP statement. A dump of the CSECT follows.

ZAP0044 Statements *n-n* processed.

This is an informational message produced in the module summary. It simply indicates that the statements indicated were successfully processed. This applies to DUMP or DUMPT statements only.

CHAPTER 12 *SirZap Return Codes*

- 0** ZAPs were processed successfully. Any modified load modules are written back to disk.
- 4** ZAPs were processed successfully, but one or more warning messages were produced. This usually indicates a CSECT was missing from one or more of the load modules. Check the *SirZap* report for the name of the module and CSECT. If any load modules were modified, they are written back to disk.
- 8** One or more errors occurred. Check the *SirZap* report for details.
- 12** *SirZap* was unable to open the input file. Make sure the filename is correct and that the file or PDS member is present.
- 40** *SirZap* could not open the report file (SYSPRINT). For MVS, make sure a DD statement for SYSPRINT is present. For CMS, the internal FILEDEF for SYSPRINT may have failed. Make sure the “A” disk is accessed in read/write mode.

APPENDIX A *Deprecated Options*

The following program options are no longer necessary. Sirius recommends that you not use them, and that if you discover them in old JCL or EXECs, you correct them to use the newer recommended approaches.

A.1 NOVERIFY

The NOVERIFY option allows you to apply ZAPs that have no verify data. Use the NOVERIFY option with caution. Once a ZAP is applied using NOVERIFY, the original contents of the load module are lost.

Note:

Since the **!NOVERIFY* statement can provide the functionality needed for applying a **portion** of a zap without verify data, we strongly recommend you do not use the NOVERIFY option, which may extend to other portions of the zap with incorrect verify data.

A.2 VERIFY

The VERIFY option ensures all replace data has been successfully verified before updating the load module. The VERIFY option is a *SirZap* default, and need not be specified.

Note:

Since we recommend you do not use the NOVERIFY option, you may wish to avoid the VERIFY option as well, thus eliminating any tendency to control requirement of verify data at the job step level.

APPENDIX B *Date Processing*

This chapter is included to note that there are no date processing issues for *SirZap*. *SirZap* does not produce any results which depend on the content of any data which may be date values. Moreover, *SirZap* does not validate whether the change made by any zap to the program it is modifying is correct (other than validating VER statements and so on), including any effect that zap may have upon date processing.

To correctly use *SirZap* past the year 1999, *SirZap* version 1.6 or later is required.

You must examine all uses of date values in your applications to ensure that each of your applications produces correct results.

Index

* !NOVERIFY statement ... 18

* !VERIFY statement ... 17

B

BACKOUT option ... 13

C

CMS ZAP ... 13

SirZap differences ... 7

Control statements ... 15

* !NOVERIFY ... 18

* !VERIFY ... 17

DUMP ... 17

NAME ... 16

REP ... 16

VER ... 15

D

DD statements ... 19

Deprecated options ... 31

NOVERIFY ... 31

VERIFY ... 31

DUMP statement ... 17

DUMPT ... 17

DUMPT statement ... 17

E

EXTRACT option ... 14

F

Features ... 3

I

IMASPZAP ... 13

SirZap differences ... 5

Installation

CMS ... 9

MVS ... 11

M

Messages ... 23

N

NAME statement ... 16

NAMEX ... 16

NAMEX statement ... 16

O

Options ... 13

BACKOUT ... 13

EXTRACT ... 14

REPORT ... 13

TXTLIB ... 14

UPPER ... 13

R

REP statement ... 16

REPORT option ... 13

Return codes ... 29

S

SIRZAP command ... 21

T

TXTLIB option ... 14

U

UPPER option ... 13

V

VER statement ... 15

