Learn to use SDSF Rexx

SHARE Phoenix, Session 24671

March 14, 2019

Rob Scott

rscott@rocketsoftware.com

Overview

- z/OS V1R9 SDSF adds support for the REXX programming language (GA September 2007)
- Use REXX to quickly develop scripts to perform complex and repetitive tasks
- Simpler and more powerful alternative to SDSF batch
  - Include logic to scripts
  - Protection from screen position issues – access table cell values directly
- SDSF Batch functionally stabilized

Overview

- With SDSF’s REXX, you can perform most of the tasks that you can perform interactively, such as :

  - Display and modify jobs
  - Display and modify resources and devices
  - Browse SYSOUT datasets
  - Print SYSOUT datasets
  - Issue system commands
  - Read SYSLOG/OPERLOG

- Use the same panel commands, action characters and column overtypes as with interactive SDSF.

GETTING STARTED

Getting Started

The basic structure of an SDSF REXX exec is :

- Add the SDSF REXX host environment using ISFCALLS
- Exec can now use “ADDRESS SDSF” statements
- Issue an SDSF command to access a panel using ISFEXEC
- Issue an action character or “overtype” a column value using ISFACT
- Remove the SDSF REXX host environment using ISFCALLS

Data is returned to the Rexx exec from SDSF in stem variables.

- The “.0” stem variable contains the number of rows
- The “.*n*” stem variable contains the column value for the *n*th row of the panel

Special REXX variables to control results

For example, ISFOWNER corresponds to the OWNER value.

Getting Started – Row Tokens

The SDSF Rexx interface is stateless.

SDSF panels typically contain one or more logical rows with data cells for each column.

Each logical row has an associated Row Token that contains encoded information that enables SDSF to replay the SDSF actions to gather the data.

Row token values are returned to the exec in the “token.i” stem variable.

Rows can “vanish” in between invocations of ISFEXEC/ISFACT if the underlying data changes.

- For example, the ISFEXEC/ISFACT was for the “DA” panel and an address space terminates.

Getting Started

Quick start example – Cancel a job

```
lastrc=isfcalls(“ON”)
address SDSF “ISFEXEC ST”
do i = 1 to JNAME.0
   if pos(“KEN”,JNAME.i) = 1 then do
      address SDSF “ISFACT ST TOKEN(‘”token.i”’) PARM(NP P)”
   end
end
lastrc=isfcalls(“OFF”)
```

Add host environment

Access the ST panel

Variable names same as FLD names

Issue the “P” action on the row

Remove SDSF host environment

ACCESSING SDSF PANELS

Accessing SDSF panels - ISFEXEC

Use the ISFEXEC statement to access a specific SDSF panel

Syntax :

address SDSF “ISFEXEC *cmd* (*options*)”

cmd is the same SDSF command you would issue interactively including any parameters, for example :

Address SDSF “ISFEXEC DA”

Address SDSF “ISFEXEC CK ALL”

Accessing SDSF panels – ISFEXEC OPTIONS

Options you can use when accessing a panel via ISFEXEC (and ISFACT) :

- PREFIX
  - Specify a prefix for the REXX variables created.
  - Do not confuse with SDSF PREFIX setting.
- ALTERNATE
  - Use the alternate FLD list – note new commands in z/OS 2.3+ do not have alternate FLD lists.
- DELAYED
  - Included delayed access columns
- NOMODIFY
  - Do not return row tokens
- VERBOSE
  - Produce extra diagnostic messages
- WAIT
  - Wait for full delay interval for retrieving responses to commands (ISFACT only)

Accessing SDSF panels - Data

SDSF builds Rexx stem variables that correspond to the panel’s rows and columns

Variable name format : *column_name.index*

Note that *column_name* is the name used on the FLDENT statement and NOT the column title, for example :

```
FLDENT COLUMN(OWNERID),TITLE(OWNER),WIDTH(8)
FLDENT COLUMN(JNAME),TITLE(JOBNAME),WIDTH(8)
```

Use the COLSHELP command to display the column names for any display.

Accessing SDSF panels – Data Example

Stem variables and values for columns on the status (ST) panel

JNAME.0	=	2
JNAME.1	=	KENA
JNAME.2	=	ROBB

OWNERID.0	=	2
OWNERID.1	=	KEN
OWNERID.2	=	ROB

…and so on

Accessing SDSF panels – Special Variables

Special variables can be used to :

- Control or limit SDSF panel response
- Access to other panel information, for example the panel title line

All special variable names start “ISF”, examples of common usage :

ISFJESNAME / ISFJES3NAME	JES2/JES3 subsystem name
ISFSYSNAME	Sysname pattern for cross-system requests
ISFOWNER	Owner pattern
ISFPREFIX	Prefix pattern
ISFFILTER	Apply SDSF filter text
ISFCOLS	List of columns in panel response (input/output)
ISFSORT	Sort criteria

When used to limit the panel response, populate the variable before the ISFEXEC or ISFACT statement.

Accessing SDSF panels – Special Variables

Other SDSF special variables include :

ISFTLINE	Panel title line
ISFROWS	Number of rows returned
ISFMSG	Short message
ISFMSG2	Stem variables containing numbered SDSF messages
ISFULOG	Stem variables for ULOG contents

There are lots more …

Accessing SDSF panels – Show all panel data

Example SDSF Rexx to show contents of all table cells

```
lastrc=ISFCALLS("ON")
address SDSF "ISFEXEC DA"
fixedfield = word(ISFCOLS,1)
say "Number of rows returned "ISFROWS
do rowindex = 1 to ISFROWS
   say "Now processing job : "value(fixedfield"."rowindex)
   do colindex = 1 to words(ISFCOLS)
      column = word(ISFCOLS,colindex)
      say column"."rowindex "has the value : "value(column"."rowindex)
   end
end
lastrc=ISFCALLS("OFF")
```

Fixed field always first column

Loop thru all rows

Loop thru all columns

Using PREFIX and Multiple Invocations

```
commands = 'DA ST ENC DEV NA LPA CK PS RM'
repeat = 2

lastrc=ISFCALLS("ON")

x = time(reset)

START_UCPU = sysvar('SYSCPU')
START_USER = sysvar('SYSUID')

ISFPREFIX = "SDSF*"
address SDSF "ISFEXEC DA (PREFIX START_  VERBOSE)"

ISFPREFIX = "*"
do i = 1 to words(commands)
  do j = 1 to repeat
    ISFCOLS = ""
    address SDSF "ISFEXEC "word(commands,i)
  end
end

END_UCPU = sysvar('SYSCPU')
```

List of commands to repeat

Reset CPU time

Take initial sample of SDSF servers and prefix rexx variables with “START_”

Reset ISFCOLS between calls !

Issue SDSF command

Using PREFIX and Multiple Invocations (cont.)

```
say left(START_USER,8) "CPU:"START_UCPU
say left(START_USER,8) "CPU:"END_UCPU
say " "
say "CPU Delta    : "END_UCPU-START_UCPU
say "Elapsed time : "format(time(elapsed),,2)
say " "

ISFPREFIX = "SDSF*"

ISFCOLS = ""
address SDSF "ISFEXEC DA (PREFIX END_ VERBOSE)"

do i = 1 to ISFROWS
  Say left(START_JNAME.i,8) "CPU:"START_CPU.i "Real:"START_REAL.i
  Say left(END_JNAME.i,8) "CPU:"END_CPU.i "Real:"END_REAL.i
end

lastrc=ISFCALLS("OFF")
```

Report on TSO userid CPU usage and elapsed time

Take ending sample of SDSF servers and prefix variables with “END_”

Display starting and ending values for SDSF server CPU and REAL

Sample Output

```
USERROB   CPU:      0.24
USERROB   CPU:      5.20

CPU Delta     : 4.96
Elapsed time  : 5.83

SDSF     CPU:35.40  Real:1701
SDSF     CPU:35.40  Real:1701
SDSFAUX  CPU:150.80 Real:1605
SDSFAUX  CPU:150.85 Real:1605
```

TAKING ACTIONS

Taking Actions - ISFACT

Use the ISFACT statement to issue an action character or modify a value (overtype a column).

Syntax :

address SDSF “ISFACT *cmd* TOKEN(‘”*token*”’) PARM(*parm*) (*options*)”

cmd	The same SDSF command that was used in the ISFEXEC statement to generate the panel.
token	The row token(s) that represents the row(s) in the SDSF table displayed, each enclosed in single quotes.

Taking Actions - ISFACT

Syntax :

address SDSF “ISFACT *cmd* TOKEN(‘”*token*”’) PARM(*parm*) (*options*)”

parm The action or column modification in “column value” format.

To issue an action, use “PARM(NP *cccc*)” where *cccc* is one of the panel’s defined actions.

Example: “PARM(NP C)”

To modify one or more column values, use “PARM(*col1 val1 col2 val2…*)” where *coln valn* represent column name/value pairs.

Example: “PARM(OCLASS A FORMS 1234)”

Taking Actions - Example

Simple example – Change Output Forms

```
ISFPREFIX="**"
ISFOWNER="ROB"
address SDSF "ISFEXEC O"
do index =1 to JNAME.0
   if pos("KEN",JNAME.index) = 1 then
      address SDSF "ISFACT O TOKEN('"token.index"') PARM(FORMS 1234)"
   end
end
```

Set Filters

Hunt for job owned by ROB that starts "KEN"

Overtype the FORMS column

BROWSING OUTPUT

Browsing Output - Methods

There are two methods to browse a job’s output

Use ISFACT with the special “SA” action to allocate the JES spool datasets and then process with EXECIO

- Ideal for small to medium number of records
- Typically all records for a single dataset are transferred to stem variable(s)
- Possible storage problems for large number of records

Use the ISFBROWSE command

- Supports user defined cursor positioning
- Supports user defined number of records to process
- Removes storage constraint for large JES datasets

Browsing Job Datasets - ISFACT

To browse a job’s datasets, use ISFACT to issue the “SA” action character against a job.

- SA allocates each separate job DD with FREE=CLOSE
- SA action only available in SDSF REXX
- Allocated DD names are returned in the ISFDDNAME stem variable.
- Corresponding JES spool dataset name returned in the ISFDSNAME stem variable

Use EXECIO to read the dataset(s)

Browsing Job Output – ISFACT Example

```
ISFPREFIX="ROBUNIQ1"
address SDSF "ISFEXEC ST"
address SDSF "ISFACT ST TOKEN('"token.1"') PARM(NP SA)"
do ddindex = 1 to ISFDDNAME.0
   say "Now reading : "ISFDSNAME.ddindex
   address TSO "EXECIO * DISKR "ISFDDNAME.ddindex " (STEM line. FINIS"
   say "Lines read : "line.0
   do lineindex = 1 to line.0
      say substr(line.lineindex,1,72)
   end
end
```

Unique Job Name

Allocate datasets

Process all DD Names

Use EXECIO

Browsing Job Output - ISFBROWSE

Use the ISFBROWSE statement to browse job output and healthchecks

Syntax :

address SDSF “ISFBROWSE *cmd* TOKEN(‘*token*’) (*options*)”

cmd	The same SDSF command you would issue interactively.		
token	The row token returned by ISFEXEC or ISFACT		
options	List of processing options :		
	JCL	-	Browse just the JCL (jobs only)
	VERBOSE	-	Add diagnostic messages

ISFBROWSE – Special Variables

ISFLINE	Output data
ISFLINELIM	Maximum number of lines to read
ISFFIRSTLINEDSID	Dataset ID to position the cursor within
ISFFIRSTLINERECNO	The record number within the DSID to read first
ISFLASTLINEDSID	Dataset ID of the last line read
ISFLASTLINERECNO	The record number within the DSID last read
ISFNEXTLINETOKEN	Token corresponding to the next unread line of data
ISFSTARTLINETOKEN	Token specifying cursor position on next read

Browsing Job Output - ISFBROWSE Example

Simple example to browse job output

```
ISFPREFIX=“ROBUNIQ1”
address SDSF “ISFEXEC ST”
ISFLINELIM = 2000
do until ISFNEXTLINETOKEN=‘’
   address SDSF “ISFBROWSE ST TOKEN(‘”token.1”’)”
   do lineindex = 1 to ISFLINE.0
      say ISFLINE.lineindex
   end
   ISFSTARTLINETOKEN = ISFNEXTLINETOKEN
end
```

Unique Job Name

Max records read

Repeat until EOF

Data in ISFLINE

Position cursor

BROWSING THE SYSTEM LOG

Browsing System Log

You can browse both the single system SYSLOG or the sysplex-wide OPERLOG using the ISFLOG command.

For the SYSLOG type, there are two processing methods

address SDSF "ISFLOG ALLOC TYPE(SYSLOG) (*options*)"

- Indicates that the logical SYSLOG is to be allocated for use by a utility such as EXECIO.
- DD Name returned in ISFDDNAME, Dataset name in ISFDSNAME

address SDSF "ISFLOG READ TYPE(SYSLOG) (*options*)"

- Indicates that the SYSLOG is to be read directly
- Records returned via the ISFLINE stem variable
- Amount of data returned controlled by special variables

Browsing System Log - OPERLOG

To browse the OPERLOG, use the following syntax :

address SDSF “ISFLOG READ TYPE(OPERLOG) (*options*)”

- Indicates that the OPERLOG is to be read directly
- Records returned via the ISFLINE stem variable
- Amount of data returned controlled by special variables

Browsing System Log – Special Variables

The default for ISFLOG READ is to get all records for the current day. This behaviour can be changed by using the following special variables.

ISFLINELIM	Maximum number of ISFLINE variables
ISFSYSID	SYSLOG sysid value (not OPERLOG)
ISFLOGSTARTTIME	Time of first log record to read (hh:mm:ss.th) *
ISFLOGSTARTDATE	Date of first log record to read (yyyy.ddd) **
ISFLOGSTOPTIME	Time of last log record to read (hh:mm:ss.th) *
ISFLOGSTOPDATE	Date of last log record to read (yyyy.ddd) **

* Only the *hh:mm* portion is required

** The date formats can be changed using the ISFDATE special variable which accepts the same inputs as the “SET DATE” interactive command.

Browsing System Log – Scrolling Variables

Special variables allow the caller to simulate scrolling through the data.

ISFFIRSTLINETOKEN

Set as the token of the first line of data to be read.

ISFNEXTLINETOKEN

Set as the token of the next unread line of data (or null for end-of-file)

ISFSTARTLINETOKEN

Specifies the token for the first line of data to be read

Similar to ISFBROWSE controls

Browsing System Log - Options

Options that can be specified on ISFLOG

WTORS

Cause any outstanding WTORs to be returned in the ISFWTOR stem variable

VERBOSE

Adds diagnostic messages to ISFMSG2 stem variable

Browsing System Log - Example

Simple example to hunt SYSLOG for a message

```
ISFLINELIM = 2000
do until ISFNEXTLINETOKEN=‛’
   address SDSF “ISFLOG READ TYPE(SYSLOG)”
   do lineindex = 1 to ISFLINE.0
      if pos(“$HASP395”,ISFLINE.lineindex) > 0 then say ISFLINE.lineindex
   end
   ISFSTARTLINETOKEN = ISFNEXTLINETOKEN
end
```

Hunt in chunks of 2000 lines

Null value is end-of-file

Process the current set of lines

Position to next section

ISSUING SYSTEM COMMANDS

Issuing System Commands

You can issue one or more system commands using the ISFSLASH command.

Syntax :

address SDSF “ISFSLASH (*stem) / list* (*options*)”

stem	The name of a stem variable containing the list of system commands to be issued.
list	The list of system commands separated by a blank or comma. Mutually exclusive with the *stem* method.

When the system command contains an embedded space, special character or required lower case character(s), enclose the system command with single quotation marks.

The maximum length of a single system command is 126 characters.

Issuing System Commands – Special Variables

Special variables that are employed by the ISFSLASH command.

ISFDELAY

Set the delay interval in seconds to wait for a response from a system command.

ISFULOG

A stem variable that contains the system command responses.

ISFCONS

Specifies the EMCS console name used for system command responses

ISFCONMOD

Specifies if the EMCS console name can be modified if there is already another console of the same name active in the sysplex.

Issuing System Command - Options

Options that can be used on the ISFSLASH command.

INTERNAL

Specifies that console ID 0 (INTERNAL) should be used to issue the command

WAIT

Specifies that SDSF should wait the full delay interval before attempting to retrieve the responses. This option is strongly recommended to ensure that the responses are accessible via the ISFULOG special variable.

Issuing System Commands - Examples

Examples of using ISFSLASH

```
address SDSF  "ISFSLASH $DA (WAIT)"
do respindex = 1 to ISFULOG.0
  say ISFULOG.respindex
end

ISFDELAY = 3
mycmd.0  = 2
mycmd.1  = "D A,L"
mycmd.2  = "D T"
address SDSF  "ISFSLASH ("mycmd.") (WAIT)"
do respindex = 1 to ISFULOG.0
  say ISFULOG.respindex
end
```

Example using a literal value

Example using stem variable

TROUBLE SHOOTING AND HELP

ISFCALLS Return Codes

The return codes from the ISFCALLS function are as follows :

00	Function completed successfully.
01	Host command environment query failed, environment not added.
02	Host command environment add failed.
03	Host command environment delete failed.
04	Options syntax error or options not defined.

SDSF Rexx Return Codes

The return codes from all ISFxxxx commands are as follows :

00	The request completed successfully.
04	The request completed successfully but not all functions were performed.
08	An incorrect or invalid parameter was specified for an option or command.
12	A syntax error occurred in parsing a host environment command.
16	The user is not authorized to invoke SDSF.
20	A request failed due to an environmental error.
24	Insufficient storage was available to complete a request.

Diagnosing Problems

- Check the ISFMSG special variable for short message information
- Check the ISFMSG2 stem special variable for more detailed messages
- Using the VERBOSE option on ISFEXEC and ISFACT issues a message to ISFMSG2 for each variable set

```
address SDSF “ISFEXEC DA (VERBOSE)”
```

```
ISF146I REXX variable JOBID.1 set, return code 00000001 value is ‘J0001234’
ISF146I REXX variable OWNERID.1 set, return code 00000001 value is ‘ROB’
```

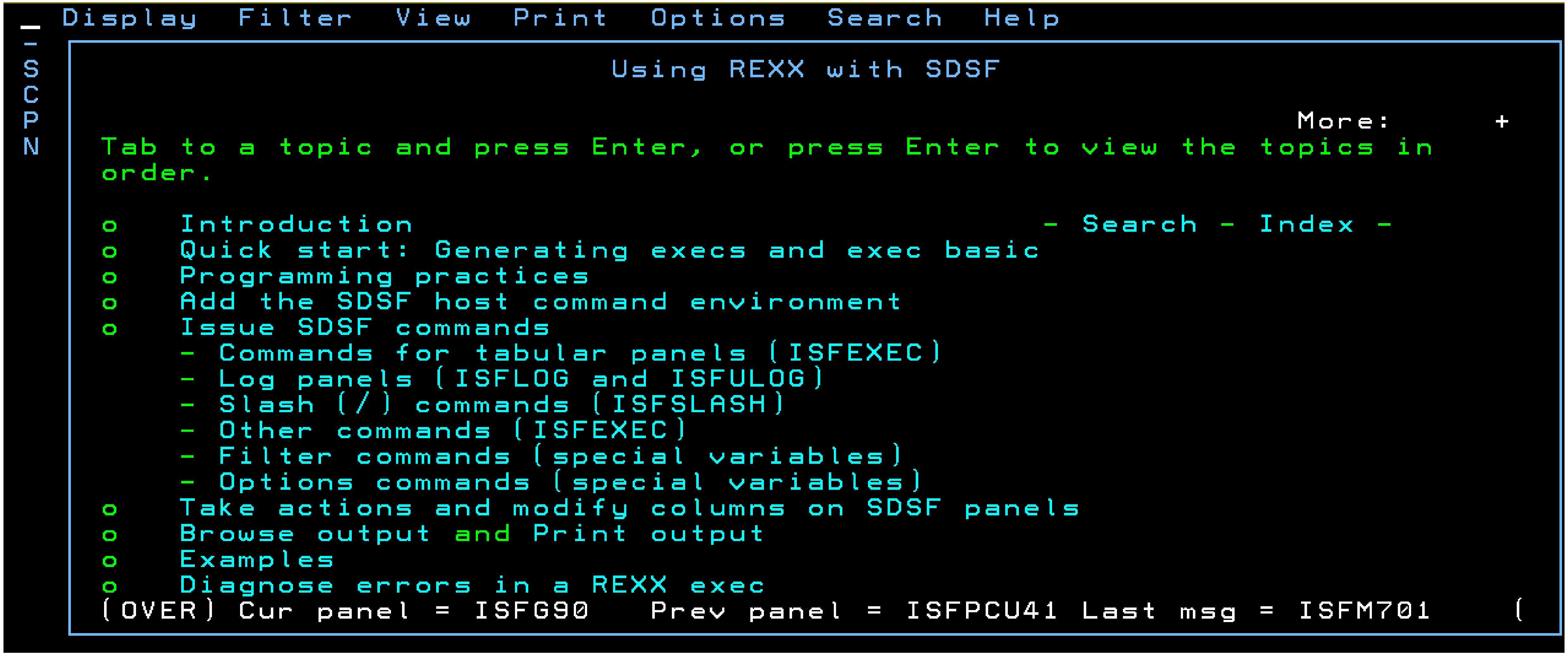
Diagnosing Problems

If not using SAF security, most common problem reported is due to being placed in the wrong SDSF group.

Issue the WHO command and output the responses returned in the ISFRESP stem.

```
address SDSF "ISFEXEC WHO"
do respindex = 1 to ISFRESP.0
   say ISFRESP.respindex
end
```

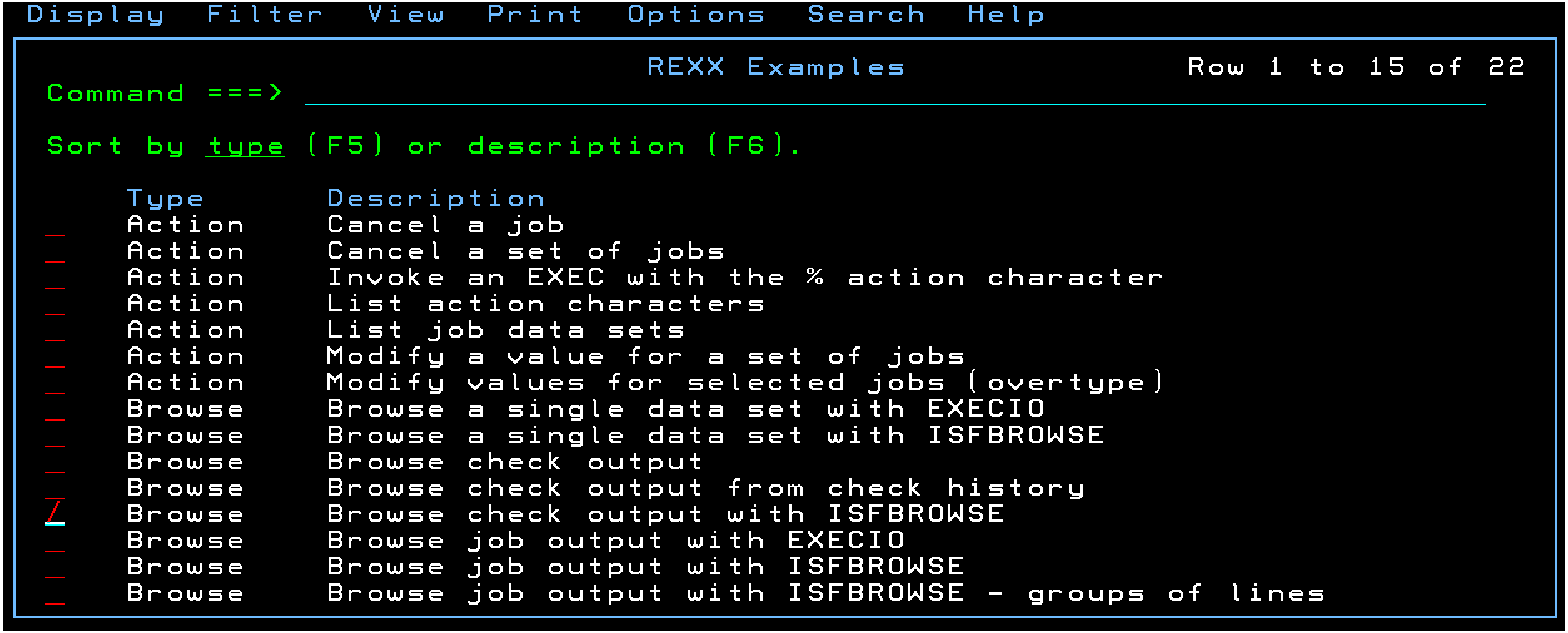
The REXXHELP command in SDSF



```
_ Display  Filter  View  Print  Options  Search  Help
-
S                       Using REXX with SDSF
C
P                                                          More:     +
N  Tab to a topic and press Enter, or press Enter to view the topics in
   order.

   o    Introduction                            - Search - Index -
   o    Quick start: Generating execs and exec basic
   o    Programming practices
   o    Add the SDSF host command environment
   o    Issue SDSF commands
        - Commands for tabular panels (ISFEXEC)
        - Log panels (ISFLOG and ISFULOG)
        - Slash (/) commands (ISFSLASH)
        - Other commands (ISFEXEC)
        - Filter commands (special variables)
        - Options commands (special variables)
   o    Take actions and modify columns on SDSF panels
   o    Browse output and Print output
   o    Examples
   o    Diagnose errors in a REXX exec
   (OVER) Cur panel = ISFG90   Prev panel = ISFPCU41 Last msg = ISFM701    (
```

Generate SDSF Rexx - RGEN



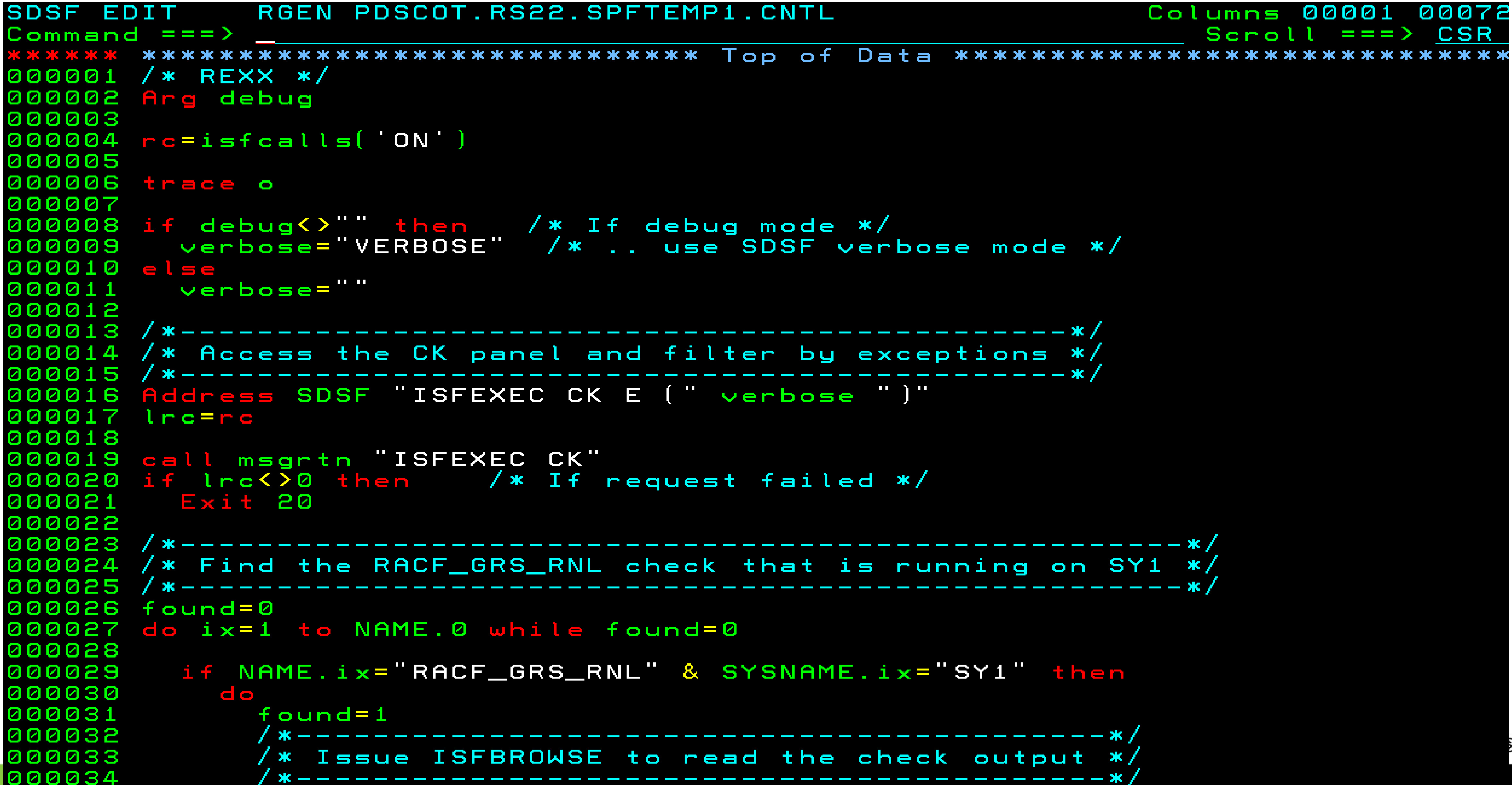
```
 Display  Filter  View  Print  Options  Search  Help

                          REXX Examples                     Row 1 to 15 of 22
 Command ===> ____________________________________________________________

 Sort by type (F5) or description (F6).

     Type        Description
 _   Action      Cancel a job
 _   Action      Cancel a set of jobs
 _   Action      Invoke an EXEC with the % action character
 _   Action      List action characters
 _   Action      List job data sets
 _   Action      Modify a value for a set of jobs
 _   Action      Modify values for selected jobs (overtype)
 _   Browse      Browse a single data set with EXECIO
 _   Browse      Browse a single data set with ISFBROWSE
 _   Browse      Browse check output
 _   Browse      Browse check output from check history
 /   Browse      Browse check output with ISFBROWSE
 _   Browse      Browse job output with EXECIO
 _   Browse      Browse job output with ISFBROWSE
 _   Browse      Browse job output with ISFBROWSE - groups of lines
```

RGEN – Example Code



```
SDSF EDIT     RGEN PDSCOT.RS22.SPFTEMP1.CNTL                     Columns 00001 00072
Command ===> _                                                      Scroll ===> CSR
****** ************************** Top of Data ******************************
/* REXX */
Arg debug

rc=isfcalls('ON')

trace o

if debug<>"" then   /* If debug mode */
  verbose="VERBOSE"  /* .. use SDSF verbose mode */
else
  verbose=""

/*----------------------------------------------*/
/* Access the CK panel and filter by exceptions */
/*----------------------------------------------*/
Address SDSF "ISFEXEC CK E (" verbose ")"
lrc=rc

call msgrtn "ISFEXEC CK"
if lrc<>0 then    /* If request failed */
  Exit 20

/*----------------------------------------------------*/
/* Find the RACF_GRS_RNL check that is running on SY1 */
/*----------------------------------------------------*/
found=0
do ix=1 to NAME.0 while found=0

  if NAME.ix="RACF_GRS_RNL" & SYSNAME.ix="SY1" then
    do
      found=1
      /*----------------------------------------*/
      /* Issue ISFBROWSE to read the check output */
      /*----------------------------------------*/
```

QUESTIONS ?