CSIREXX

A Catalog Search Interface for REXX

Lennie Dymoke-Bradshaw
24 May 2021

Document Control

Version Control:

Version	*Author*	*Date*	*Changes*
0.01	Lennie Dymoke-Bradshaw	2021.05.24	First published version

Table of Contents

1. Introduction

CSIREXX is a routine which will enable users of the REXX language to interrogate the catalogs on a z/OS system and extract attributes of catalog entries.

It is a read-only interface.

At its heart it makes use of the routine IGGCSI00 which is supplied as part of z/OS. However, the interface to IGGCSI00 is complicated for a REXX user to use simply. CSIREXX makes the extraction and use of attributes about data sets' catalog entries far simpler.

Much of the interface of CSIREXX is modelled on the interface that IBM has supplied for the RACF data extraction routine IRRXUTIL, which is used for extracting data from the RACF database. However, there are some notable difference. These arise due to the nature of the IGGCSI00 routine.

2. How to use CSIREXX

CSIREXX is simple to invoke. It has a maximum of 6 parameters, but can be used for simple queries with as few as 3 parameters.

All parameters are positional.

The search selection capabilities are based on those specified in the documentation for IGGCSI00. This documentation can be found in the IBM manual,

z/OS DFSMS Managing Catalogs SC23-6853-40

The above manual is supplied with z/OS Version 2 Release 4.

IGGCSI00 is described in Chapter 11 of that manual.

Note that each call to CSIREXX will provide details about a maximum of 1 catalog entry. Many fields can be extract about that entry in the same call. There are over 100 such fields.

2.1 The parameters

The standard way to call CSIREXX is via the following statement.

```
Myrc = CSIREXX(verb,key,prefix,workarea,workarealen,extras)
```

2.1.1 Myrc

This rexx variable will be populated with the return codes from CSIREXX. There are three values returned and these are each 8 characters separated by a single blank.

For example, a normal call the in which data is returned, the return codes will be set to

```
00000000 00000000 00000000
```

Error conditions will be set into these fields where appropriate. Other indicators will be set here as well.

When an attempt is made to extract past the last entry the following return code is given.

```
B0000008 00000000 00000000
```

2.1.2 Verb

This value can be set to READ, READI, READQ, READNQ, READN or READNI.

READI is designed to extract index only entries for a single catalog entry. It can be used with a fully qualified name specified in the “key” parameter. The extras parameter will be ignored. No fields will be extracted other than those designated Index Fields in section 3.1.

READ is designed for retrieval of data about a single catalog entry. It can be used with a fully qualified name specified in the “key” parameter. It returns all the necessary fields to redefine the data set.

READQ is designed for “Quick” processing but with fields. You will need to specify all fields that you require.

READNI is designed for the retrieval of index entries for multiple catalog entries. The extras parameter will be ignored. No fields will be extracted.

READN is designed to retrieve index entries and data fields from multiple catalog entries with a sequence of CSIREXX calls. Each call retrieves the standard fields plus any extras specified.

READQN is designed to retrieve index entries and data fields from multiple catalog entries with a sequence of CSIREXX calls. Each call retrieves only those fields specified in extras.

Verb is a required parameter.

2.1.3 Key

The key is the selection mask used match catalog entries. Key should be between 1 and 44 characters in length.

In the IBM documentation for IGGCSI00 the following is described.

> *CSI uses a generic filter key supplied in CSIFILTK. A generic filter key is a character string that describes the catalog entry names or catalog names (Z entry type) for which you want information returned. The generic filter key can contain the following symbols and are interpreted as follows:*
>
> * *A single asterisk by itself indicates that either a qualifier or one or more characters within a qualifier can occupy that position. An asterisk can precede or follow a set of characters.*
>
> ** *A double asterisk indicates that zero or more qualifiers can occupy that position. A double asterisk cannot precede or follow any characters; it must be preceded or followed by either a period or a blank.*
>
> *For Z entry type requests, this filter key specified by itself will return information for all catalogs that have been opened since the system was IPLed.*
>
> *% A single percent sign by itself indicates that exactly one alphanumeric or national character can occupy that position.*
>
> *%%... One to eight percent signs can be specified in each qualifier.*
>
> *If an absolute GDS name is specified followed by a period and one of the previous symbols, the GDS name (if it exists) will not be returned from the search. However, it is possible that longer data set names that start with the same prefix as the absolute GDS name will be found.*

Note that Z type entry requests (Catalog requests) are not currently supported by CSIREXX.

An example of the key parameter could be set using the following REXX statement,

```
CSIREC= "LENNIE.A.**'
```

The *Key* parameter would then be set as CSIREC.

Key is a required parameter.

2.1.4 Prefix

CSIREXX sets values into REXX variables. The names of the variables are compound. They consist of the prefix and the name of the data item (or the field name). Field Names are described in 4 Field names.

Prefix should be set to a character value of length between 1 and 32.

Any valid REXX name is acceptable. Validation will check the characters used and will allow a period character (.). However, it may be possible to specify values which do not conform to REXX variable standards. Results are unpredictable if this is done.

Prefix is a required parameter.

2.1.5 Workarea

In order to provide details about a set of data sets it is necessary to keep track of what catalog entries CSIREXX has processed. IGGCSI00 can provide details about multiple entries in a single call, bit CSIREXX supplies details of only 1 catalog entry per call.

In order for CSIREXX to maintain positioning it is necessary for the caller to provide a work area to CSIREXX. This work area will be used to store 2 sections. The first section contains details that CSIREXX uses for processing. The second part is used for IGGCSI00 to return details to CSIREXX.

IGGCSI00 states that a work area of between 1024 and 8,388,608 bytes can be supplied. It also states that a work area of 64,000 is generally recommended. Clearly this is far more than needed for a single-entry lookup.

The *workarea* parameter specifies the name of a variable which contains the work area.

On the first call to CSIREXX the first 4 bytes of the work area must be set to binary zeroes. CSIREXX will increment this value on each call.

The work area should **not be altered** between calls to CSIREXX. (However, it can be examined for diagnosis. It is a useful source of diagnostic material. (See section 6).

The variable supplied in the *workarea* parameter should contain the name of the variable that contains the work area. The following sequence of REXX statements will set up the workarea for use.

```
worklen = 16384
work = COPIES('00'X,4)||COPIES('00'X,worklen-4)
workarea = 'work'
```

workarea is not a required parameter if READ or READI is specified for the *verb*.

If *workarea* is not supplied then a default work area of 8k (8192 bytes) will be used, unless over-ridden by the *workarealen* parameter.

2.1.6 Workarealen

workarealen is the length of the supplied work area. If *workarea* is supplied then this value must also be supplied.

If not supplied this value will default to 8192 bytes. This is designed to be used when a *workarea* parameter is not supplied.

This parameter can be between 4096 and 1048576 (4KB min, 1MB max).

2.1.7 Extras

Extras is a field which specifies the names of extra fields that are to be extracted. Field names are only extracted and returned if the verb parameter specifies READ or READN.

The names and attributes of the fields which can be extracted are shown in 4 Field names.

Some of these are always extracted if either READ or READN is specified in the verb parameter. However, if READI or READNI is specified then no fields will be returned. These are designed to include those values which are needed to recreate the data set represented by the catalog entry.

If extra fields are needed, then these can be extracted by including a variable which contains a set of 8-byte field names. These field names must be left justified and concatenated together.

For example, if the fields EXCPEXIT, FILESEQ and HKRBA were needed in addition to those normally supplied, then the following REXX extract would suffice.

```
CSIFIELDS  = ''
CSIFIELDS  = CSIFIELDS||SUBSTR('EXCPEXIT',1,8)
CSIFIELDS  = CSIFIELDS||SUBSTR('FILESEQ',1,8)
CSIFIELDS  = CSIFIELDS||SUBSTR('HKRBA',1,8)
extras     = 'CSIFIELDS'
```

'extras' is not a required field. If it is supplied its length must be a multiple of 8.

2.2 Examples

The following examples show how to use CSIREXX to extract data.

2.2.1 Example 1

This is a simple call to extract data for a data set called LENNIE.A.TEST.

```
verb       = 'READ'
key        = 'LENNIE.A.TEST'
prefix     = 'CSIREC'
myrc       = CSIREXX(verb,key,prefix)
```

2.2.2 Example 2

This is a simple call to extract data for a data set called LENNIE.A.TEST. but with some extra fields required.

```
Verb       = 'READ'
key        = 'LENNIE.A.TEST'
prefix     = 'CSIREC'
CSIFIELDS  = ''
CSIFIELDS  = CSIFIELDS||SUBSTR('EXCPEXIT',1,8)
CSIFIELDS  = CSIFIELDS||SUBSTR('FILESEQ',1,8)
CSIFIELDS  = CSIFIELDS||SUBSTR('HKRBA',1,8)
extras     = 'CSIFIELDS'
myrc = CSIREXX(verb,key,prefix,,,extras)
```

2.2.3 Example 3

This is a complex set of calls to extract data for all data sets starting LENNIE.A. Three fields are requested explicitly. Actually these fields would be selected by default. See Table 2 - Field values for CSIREXX.

```
verb       = 'READN'
key        = 'LENNIE.A.**   '
prefix     = 'CSIREC'
worklen    = 65536
WORKAREA   = COPIES('00'X,4)||COPIES('00'X,worklen-4)
workname   = 'WORKAREA'
CSIFIELDS  = ''
CSIFIELDS  = CSIFIELDS||SUBSTR('VOLSER',1,8)
CSIFIELDS  = CSIFIELDS||SUBSTR('DSCRDT2',1,8)
CSIFIELDS  = CSIFIELDS||SUBSTR('DSEXDT2',1,8)
extras     = 'CSIFIELDS'
do j = 1 to 9999
  myrc = CSIREXX(verb,key,prefix,workname,worklen,extras)
  if myrc = '00000000 00000000 00000000' then do
    /* the extracted entries can be processed here  */
    end
    else leave
  end
say j-1 'data sets found'
```

3. Values returned by CSIREXX

As long as CSIREXX finds an entry to match the selection criteria, data will be returned and REXX variables will be populated with that returned data.

Some variables are always returned.

For READI and READNI only the three index fields are returned.

For READ and READN these fields are divided into two sets; those that correspond to the index of that catalog and those that are contained within the data.

3.1 Index fields

Three fields are returned from the index of each entry. Note that these fields do not have structure with a numeric suffix.

3.1.1 CATALOG

This is the catalog that contains the entry in question. If a prefix value of CSIREC was used, then a REXX variable called CSIREC.CATALOG will be set up containing the 44-byte name of the catalog in EBCDIC.

3.1.2 ENTNAME

This is the name of the entry in question. If a prefix value of CSIREC was used, then a REXX variable called CSIREC.ENTNAME will be set up containing the 44-byte name of the entry in EBCDIC. This entry could be of many different types. The type is found by examining the next field.

3.1.3 ENTYPE

This is the type of the entry in question. If a prefix value of CSIREC was used, then a REXX variable called CSIREC.ENTYPE will be set up containing the 1-byte type of the entry in EBCDIC. This byte will contain one of the values as shown in Table 1 - ENTYPE values.

Table 1 - ENTYPE values

Type	*Description*	*Supported by CSIREXX*
A	Non-VSAM data set	Yes
B	Generation data group	Yes
C	Cluster	Yes
D	Data component	Yes
G	Alternate index	Yes
H	Generation data set	Yes
I	Index component	Yes
L	Tape volume catalog library entry	No
R	VSAM path	Yes
U	User catalog connector entry	No
W	Tape volume catalog volume entry	No
X	Alias	No
Z	Catalog control block data	No

3.2 Data fields

Data fields are only returned if READ or READN is specified in the verb parameter.

Data fields from IGGCSI00 come with a number of attributes, such as the following,

- The format of the data according to the IBM documentation can be one of the following. However, some fields have no description of their format. Others are combination fields. Some fields have EBCDIC characters but can also contain hexadecimal X'FF'.
  - Character (i.e. EBCDIC)
  - Date
  - Hexadecimal
  - Bit-string
  - Binary
  - Fixed binary value
  - Mixed
  - Bit
- Length. Most fields have a fixed length. However, others are variable length and are preceded with a length field. All fields defined as variable length are in character format.
- Repeated fields. Some fields have multiple instances. Common examples are VOLSER and ASSOC. Repeated fields can be variable length as well.
- There appears to be a rule that if a field is supplied as all X'FF' then it is to be treated as not present. CSIREXX follows this rule for all formats of data except Fixed binary values.

3.2.1 How CSIREXX treats these fields

For each possible field CSIREXX has an internal table indicating how the field should be treated. The formats outputted by CSIREXX are the following,

1. Bit-string. Bit-strings can be from 1 to 4 bytes in length. The data returned in the corresponding REXX variable will be a series of 1s and 0s formatted as EBCDIC characters. So, a 1-byte bit-string returns an 8 bytes character field, while a 4-byte bit-string will return a 32-byte character field of 1s and 0s.
2. Fixed format. This is a binary number of a fixed number of bytes from 1 to 8. This is converted to an EBCDIC value with no leading zeros.
3. Date format. There are several fields which are formatted as dates. These are returned in the form yyyy.ddd. That is a four-digit year and a three-digit day number separated with a period (.).
4. Character field. This is returned as a character field padded with blanks on the right to the length as specified in the IGGCSI00 documentation. So, for example, the ENTNAME field which contains a data set name will be padded on the right with blanks, up to 44 chars if necessary.
5. Hexadecimal field. Several fields can only sensibly be processed as hexadecimal. These field are returned as EBCDIC characters 0-9, A-F. Hence a 4-byte hex field will be returned as an 8-byte EBCDIC character field.

Each field that can be returned is mapped into one of the above formats. The details are shown in 4 Field names.

3.2.2 Presence of a field

Fields can be requested but they may not be returned by IGGCSI00. This may be for one of several reasons, including,

- Not appropriate for the entry type,
- Not supplied at time of the entry definition,

- Field is empty,
- and possibly other reasons which are undocumented.

If a field is not present what is returned will depend on the field's format and what is returned by IGGCSI00. Some fields may be set as zero length, while others do not exist.

Each field is returned as 1 or more entries. The entries form a single dimensional array using REXX compound variables.

So, if the field XXXX is requested with a *prefix* of CSIREC then as a minimum the field CSIREC.XXXX.0 will be returned with a 0 as its contents. This indicates that there are no more fields returned for the field XXXX.

If a field is returned with CSIREC.XXXX.0 equal to 1, then there will also exist a variable called CSIREC.XXXX.1 which will contain something. However, that something may be a null field.

If the field is a repeater, such as VOLUME or ASSOC, then the first (i.e. 0th) entry will contain the number of entries.

The IBM manual for IGGCSI00 states[1],

> *If the data field to be retrieved has fixed length and does not exist, ……. the data is set to be all X'FF'*

CSIREXX detects such fields and so does not return the field. So, in most cases, the "zero" value of the field variable will be set to '0' to show nothing is returned.

3.2.3 Example

The following example shows how data can be extracted for a single data set. The number of volumes for the data set is retrieved along with all the volumes, and the creation date and expiration date.

```
/*  REXX    */
 verb       = 'READ'
 key        = 'LENNIE.A.CNTL '
 prefix     = 'CSIREC'
 myrc = CSIREXX(verb,key,prefix)
 if myrc = '00000000 00000000 00000000' then do
   say '-------------------------------------------------------'
   say "CATALOG= '"||CSIREC.CATALOG||"'"
   say "ENTNAME= '"||CSIREC.ENTNAME||"'"
   say "ENTYPE=  '"||CSIREC.ENTYPE||"'"
   say "VOLSER.0= '"||CSIREC.VOLSER.0||"'"
   do i = 1 to CSIREC.VOLSER.0
     say "VOLSER."i"='"||CSIREC.VOLSER.i||"'"
     end
   say "DSCRDT2= '"||CSIREC.DSCRDT2.1||"'"
   say "DSEXDT2= '"||CSIREC.DSEXDT2.1||"'"
   say '-------------------------------------------------------'
   end
   else say myrc
return
```

In this instance I have assumed that the creation date and expiration date will exist. You may wish to test the '0' value for these fields (i.e. CSIREC.DSCRDT2.0 and CSIREC.DSEXDT2.0) before using them.

3.3 Differences with IDCAMS reports

In Appendix B of the IBM manual,

> *z/OS DFSMS Access Method Services Commands SC23-6846*

[1] This is a strange concept as some of the fixed length strings are bit strings.

is a description of the way that data is output by the LISTCAT command of the IDCAMS program.

Users of CSIREXX and IGGCSI00 may need to be aware of some differences in the way that items are reported. A prime example is that LISTCAT shows Generation Data Sets as the same as non-VSAM data sets, whereas CSIREXX and IGGCSI00 classify them differently as ENTYPE=H and ENTYPE=A. There are other fields that are interpreted differently. Indeed, many of the values in LISTCAT output cannot be found in IGGCSI00 or CSIREXX without interpretation.

4. Field names

This section includes a table of the valid field names that can be requested. Sadly, IBM have declined to document details of which fields are valid with which entry types. For each field refer to IBM documentation for what it is intended to contain (if you can find it). Some fields I have been unable to find any documentation for. I presume these are used internally and are not of any use for me. However, they are still supported for retrieval.

The 'Type' column in Table 2 - Field values for CSIREXX contains one of the following values, C, F, B, A, M, xxx. Or Undefined. The table shows how each field will be returned by CSIREXX. The term 'Fixed' means this is a fixed-length binary number. The CSIREXX shows the format that CSIREXX will recognise. The 'Treated as' column shows those cases where CSIREXX forces a different format for usability.

The 'Rep' column shows when fields are potentially repeated.

The 'Std' column shows those fields that are always retrieved. No *extras* processing is necessary for these fields. All other fields will require to be requested via the *extras* parameter.

Table 2 - Field values for CSIREXX

Field	*Len*	*Type*	*Rep*	*Description*	*Std*	*CSIREXX*	*Treated as*
ACTOKEN	36	Character		Active compression dictionary token		Character	
AKEYPOS	2	Fixed		Relative key position for AIX	Y	Fixed	
AMDCIREC	8	Fixed	Yes[2]	Control Interval size (4 bytes) Max record size (4 bytes)	Y	Binary	Treated as a repeated field of 4 bytes fixed.
AMDKEY	4	Fixed	Yes[2]	Relative key position (2 bytes) Key length for KSDS (2 bytes)	Y	Binary	Treated as a repeated field of 2 bytes fixed.
ASSOC	45	Character	yes	Associated records		Character	
ASSOCSYB	1	Bit-string		Symbolic indicator for Associated records		Bit-string	
ASSOCSYM	45	Character	yes	Associated records		Character	
ATTR1	1	Bit-string		Attributes	Y	Bit-string	
ATTR2	1	Bit-string		Share Attributes	Y	Bit-string	
BUFND	2	Fixed		Buffers requested for data	Y	Fixed	
BUFNI	2	Fixed		Buffers requested for index	Y	Fixed	
BUFSIZE[3]	4	Fixed		Max buffer size	Y	Fixed	
CATACT	2	Fixed		Catalog activity count		Fixed	Unsupported[4]
CATTR	1	Bit-string		Pagespace attributes		Bit-string	

[2] Field not defined as repeated by IBM but treated as repeated by CSIREXX.
[3] Corresponds to BUFFERSPACE.
[4] Specification of this field when not requesting Z entries can result in ABEND0C4 in catalog address space.

Field	*Len*	*Type*	*Rep*	*Description*	*Std*	*CSIREXX*	*Treated as*
CATUCB	4	Address		UCB pointer			Unsupported[4]
COMPIND	1	Bit-string		Compression indicator		Bit-string	
COMUDSIZ	8	Fixed		Compressed user size		Fixed	
DATACLAS	VL	Character		SMS data class	Y	Character	
DBALTKEY		Undefined		Alternate keys		Undefined	Unsupported[5]
DBALTKY1		Undefined		Alternate key stuff		Undefined	Hex
DEVTYP	4	Fixed	yes	Device type		Binary	Hex
DSCBTTR	3	Fixed	yes	TTR of Format1 DSCB		Binary	Hex
DSCRDT2	4	Mixed		Creation date	Y	Date	
DSEXDT2	4	Mixed		Expiration date	Y	Date	
EATTR	1	Binary		Data set extended attributes	Y	Bit-string	
ENCRYPTF	1	Binary		Encryption flag	Y	Bit-string	
ENCRYPTT	2	Fixed		Encryption type	Y	Mixed	Hex
ENCRYPTA	96	Character		All encryption fields	Y	Mixed	Hex
ENTNAME	44	Character		Entry name	Y	Character	
ENTYPE	1	Character		Entry type	Y	Character	
EXCPEXIT	8	Character		Name of exit		Character	
FILESEQ	2	Fixed	yes	File sequence number	Y	Fixed	
FRLOG[6]	1	Bit-string		FRLOG value	Y	Bit-string	
FSDSFLAG	1	Character		File System data set flag		Character	
GDGALTDT	4	Mixed		Last alteration date	Y	Date	
GDGATTR	1	Bit-string		GDG attributes	Y	Bit-string	
GDGLIMIT	1	Fixed		Max number allowed	Y	Fixed	
GDGLIMITE	2	Fixed		Max number allowed for GDG or GDGE	Y	Fixed	
GENLEVEL	4	Character	yes	Generation level	Y	Character	
HARBA	4	Fixed	yes	High allocated RBA		Fixed	
HARBADS	4	Fixed		Data set high-allocated RBA		Fixed	
HIKEYV	VL	Character	yes	High key on volume		Character	
HILVLRBA	4	Fixed		RBA of High-Level Index Record		Fixed	
HKRBA	4	Fixed	yes	RBA of data control interval with high key		Fixed	

[5] This field is under-described in the manual. It contains 7 fields of varying format, one of which is 2000 bytes.
[6] Undocumented field. Details supplied unofficially from IBM.

Field	*Len*	*Type*	*Rep*	*Description*	*Std*	*CSIREXX*	*Treated as*
HURBA	4	Fixed	yes	High-user RBA for the volume requested		Fixed	
HURBADS	4	Fixed		Data set high-used RBA		Fixed	
INDXLVLS	2	Fixed		Number of index levels	Y	Fixed	
ITYPEXT	1	Bit-string	yes	Type of extent	Y	Bit-string	
KEYLABEL	64	Character		Encryption key label	Y	Character	
LOCKSTNM	16	Character		Data set lock structure Name		Character	
LOGPARMS	1	Binary		Value of log parameter set	Y	Bit-string	
LOGSTRID	26	Character		Value of LOGSTREAMID parm	Y	Character	
LOKEYV	VL	Character	yes	Low Key on volume		Character	
LRECL	4	Fixed		Average logical record length	Y	Fixed	
LTBACKDT	8	Fixed		Last backup date in TOD format		TOD	Hex
MGMTCLAS	VL	Character		SMS management class	Y	Character	
NAME	44	Character	yes	The name of an associated entry	Y	Character	
NOBLKTRK	2	Fixed	yes	Number physical blocks per track		Fixed	
NOBYTAU	4	Fixed	yes	Number bytes per allocation unit		Fixed	
NOBYTTRK	4	Fixed	yes	Number of bytes per track		Fixed	
NOEXTNT	1	Fixed	yes	Number of extents	Y	Fixed	
NOTRKAU	2	Fixed	yes	Number of tracks per allocation unit		Fixed	
NVSMATTR	1	Character		Non-vsam attribute info	Y	Character	
OPENIND	1	Bit-string		Open indicator		Bit-string	
OWNERID	8	Character		Owner of the data set		Character	
PASSATMP	2	Fixed		Number of attempts to prompt for password		Fixed	
PASSPRMT	8	Character		Password prompt code name		Character	
PASSWORD	32	Character		Four 8-byte passwords		Character	
PHYBLKSZ	4	Fixed	yes	Physical blksize	Y	Fixed	
PRIMSPAC	3	Fixed		Primary space allocation	Y	Fixed	

Field	*Len*	*Type*	*Rep*	*Description*	*Std*	*CSIREXX*	*Treated as*
RECVTIME	8	Binary		Recovery time, TOD value, local		Binary	Hex
RECVTIMG	8	Binary		Recovery time, TOD value, GMT		Binary	Hex
RGATTR	1	Bit-string		Alt index/path attributes	Y	Bit-string	
RLSBWO	1	Bit		Value of BWO parameter	Y	Bit-string	
RLSFLAGS	1	Bit		No description provided	Y	Bit-string	
SCONSPAC	3	Fixed		Secondary space allocation	Y	Fixed	
SECFLAGS	1	Bit-string		Security Flag information		Bit-string	
SEQSTRBA	4	Fixed		RBA of first sequence set record		Fixed	
SMSSFLAG	1	Bit-string		SMS flags	Y	Bit-string	
SPACOPTN	1	Bit-string			Y	Bit-string	
STORCLAS	VL	Character		SMS Storage class	Y	Character	
STRIPCNT	2	Fixed		Striping counts for striped data sets		Fixed	
STRNO	1	Fixed		Number of concurrent requests		Fixed	
TRACKS	4	Fixed	yes	Total number of tracks per volume	Y	Fixed	
TYPE	1	Character	yes	Type of associated entry	Y	Character	
UDATASIZ	8	Fixed		User data size		Fixed	
USERAREC	VL	Character		User authorisation record		Character	
USVRMDUL	8	Character		User security module		Character	
VOLFLG	?	xxx.	yes		Y	Bit-string	
VOLSER	6	Character	yes		Y	Character	
VSAMREUS	1	Bit-string		VSAM data set information	Y	Bit-string	
VSAMSTAT	46	Fixed			Y	Mixed	Hex
VSAMTYPE	2	Bit-string			Y	Bit-string	
VVRNFLGS	2	Bit-string		Extended format flags	Y	Bit-string	
XACIFLAG	1	Bit-string		Extended attribute flags		Bit-string	
XHARBA	8	Fixed	yes	High allocated RBA		Fixed	
XHARBADS	8	Fixed		Data set high allocated RBS		Fixed	
XHKRBA	8	Fixed	yes	RBA of data control interval with high key		Fixed	

Field	*Len*	*Type*	*Rep*	*Description*	*Std*	*CSIREXX*	*Treated as*
XHURBA	8	Fixed	yes	High used RBA for the volume requested		Fixed	
XHURBADS	8	Fixed		data set high used RBA		Fixed	

5. Error return values from CSIREXX

The values returned in the three 8-byte fields shown various error and exception conditions. The following are recognised sets of values.

5.1 Common return values that are needed in normal processing.

The following codes will probably be needed in normal processing. Other codes will be for diagnosis of a problem only.

Codes	***Reason***
B0000008 00000000 00000000	End of list of datasets
80000040 00000000 CATALOG	Requested key did not match any entries
70000000 00000004 C6E20464	Requested a field not supported at this z/OS level[7] or unable to access VVDS for a dataset[8].
70000000 00000004 C6E21E7A	Probably a bad search argument in the *key* parameter[9]

5.2 Structure of codes

The first character of the first field tells us the source of the error or condition.

0 – Parsing error

1 – Problem getting storage for the work area

2 – Problem retrieving work area

3 – Problem found validating EXTRAS values

4 – Not used

5 - Too many Extra fields

6 - IGGCSI00 load failure

7 - IGGCSI00 bad return code

8 - IGGCSI00 reported error in field

9 - Failure returning field to REXX variable

A – Problem returning work area to REXX variable

B - Returning a field to REXX

5.3 Parameter errors

Parameter errors are shown with the following format,

000000PE 00000000 00000000

Where P is the parameter number

E is the error type.

Values for P are as follows,

1 = verb

2 = selection key

[7] An example is using ZFSFLAGS on a z/OS 2.2 system. ZFSFLAGS is supported on z/OS 2.3 and above.
[8] If you receive this code you may need to research which data set has the problem.
[9] Look for CSIFILTK description in "z/OS Managing Catalogs" for what is valid in a search argument.

3 = prefix variable

4 = workarea variable

5 = workarea length

6 = extras variable

Values for E are as follows,

1 = Parameter not present

2 = Parameter too short

3 = Parameter too long

4 = Parameter has invalid chars

5 = Parameter has invalid values

6 = Parameter has invalid 1st char

7 = Parameter is too small

8 = Parameter is too big

9 = Parameter has invalid last char

5.4 Problem getting storage for work area

Work area storage problems have the format,

10000000 SSSSSSSS LLLLLLLL

Where

SSSSSSSS is the return code from the STORAGE OBTAIN macro.

LLLLLLLL is the length of the storage requested.

5.5 Problem retrieving work area from REXX

Problems encountered retrieving a work area from REXX have the following format,

200000SS RRRRRRRR 00000000

Where

SS is the value of the SHVRET code from the IRXEXCOM invocation.

RRRRRRRR is the return code from IRXEXCOM.

5.6 Problem found validating extras field

Problem in the *extras* field have the following format,

3000T0SS RRRRRRRR FFFFFFFF

Where

T is one of the following

1 - Retrieve failure for '*extras*' REXX variable

2 - length not multiple of 8, or more than 100 fields supplied

3 - Name not valid

SS is the value of the SHVRET code from the IRXEXCOM invocation (if relevant, else 00).

RRRRRRRR is the return code from the IRXEXCOM invocation.

FFFFFFFF is the name of the failing field (or 00000000).

5.7 Too many extras fields

If too many extras fields are found the return values have the following format,

50000000 MMMMMMM 00000000

Where

MMMMMMMM is the maximum number of slots available.

5.8 IGGCSI00 load failure

If there is a failure to load IGGCSI00 the return values will have the following format,

60000000 RRRRRRR SSSSSSSS

Where

RRRRRRRR is the LOAD return code

SSSSSSSS is the LOAD reason code

5.9 IGGCSI00 return code

If IGGCSI00 issues an unexpected return code the return values will be formatted as,

700000000 RRRRRRRR SSSSSSSS

Where

RRRRRRR is the IGGCSO00 return code

SSSSSSSS is the 4 byte reason area (as described under return codes in the description of IGGCSI00).

SSSSSSSS is structured as follows,

Bytes 1-2	Module ID
Byte 2	Reason code
Byte 4	Return code

The reason codes and return codes are as described by message IDC3009I and can be found in the appropriate messages manual for z/OS.

5.10 IGGCSI00 reports error in field

If there is a problem in a returned field the return values will have the following format,

8000EECC RRRRRRRR 00000000

Where,

CC is the 2-byte hex value of the catalog information return code taken from the CSIEFLAG field.

EE is the 2-byte hex value of the entry information return code taken from the CSICFLAG field.

RRRRRRRR is the 4 bytes taken from the CSIRETN or CSICRETN field.

Refer to the IGGCSI00 documentation for the meaning of these fields. Further analysis can be found by examining the contents of the work area. Chapter 6 Diagnosis has details on how this can be formatted.

5.11 Failure returning field to REXX

If a failure occurs returning a field value to REXX the return values will have the following format,

90000000 000000SS FFFFFFFF

Where

SS is the value of the SHVRET code from the IRXEXCOM invocation (if relevant, else 00).

FFFFFFFF is the name of the failing field.

5.12 Failure returning work area

If a failure occurs returning a work area to REXX the return values will be formatted as follows,

A000000SS RRRRRRRR 00000000

Where

SS is the value of the SHVRET code from the IRXEXCOM invocation (if relevant, else 00).

RRRRRRRR is the return code from IRXEXCOM.

5.13 Problem returning a field to REXX

If a problem is found returning a field to REXX the return values will be formatted as follows,

B000000T 00000000 FFFFFFFF

Where,

T is one of the following

1 - Bit String too long

2 - More than 4095 repeaters

3 - Length error

FFFFFFFF is the failing field.

6. Diagnosis

Sometimes it helps to see the exact data that has been returned by IGGCSI00. This can be accomplished by formatting the work area variable. Remember that the work area will only be returned if READN, READNQ or READNI is specified in the verb parameter.

6.1 REXX to format work area

The work area can be formatted using code similar to the following. This uses CSIREXX to print details of a selection of data sets and then formats the final work area to a data set called "userid.A.SYSOUT".

```
/*  REXX                                                          */
 verb       = 'READN'
 key        = 'LENNIE.%.CNTL '
 prefix     = 'CSIREC'
 worklen    = 64 * 1024
 WORKAREA   = COPIES('00'X,4)||COPIES('00'X,worklen-4)
 workname   = 'WORKAREA'
 say LENGTH(WORKAREA)
 say SUBSTR(C2X(WORKAREA),1,32)
 do j = 1 to 9999 until myrc <> '00000000 00000000 00000000'
   say '-------------------------------------------------'
   myrc = CSIREXX(verb,key,prefix,workname,worklen)
   if myrc = '00000000 00000000 00000000' then do
   /*                                                        */
     say '-------------------------------------------------'
     say "CATALOG= '"||CSIREC.CATALOG||"'"
     say "ENTNAME= '"||CSIREC.ENTNAME||"'"
     say "ENTYPE=  '"||CSIREC.ENTYPE||"'"
     say "VOLSER.0= '"||CSIREC.VOLSER.0||"'"
     do i = 1 to CSIREC.VOLSER.0
       say "VOLSER."i"='"||CSIREC.VOLSER.i||"'"
       end
     say "DSCRDT2= '"||CSIREC.DSCRDT2.1||"'"
     say "DSEXDT2= '"||CSIREC.DSEXDT2.1||"'"
     say
     end
   else say myrc
   end
 say j-1 'data sets found'
 table = ''
 table = table||'................................'
 table = table||'................................'
 table = table||' ...........<(+|&.........$*);¬'
 table = table||"./........¦,%_>?..........:#¬'="||'"'
 table = table||'.abcdefghi.......jklmnopqr......'
 table = table||'..stuvwxyz......................'
 table = table||'{ABCDEFGHI......}JKLMNOPQR......'
 table = table||'\.STUVWXYZ......0123456789......'
ADDRESS TSO NEWSTACK
 "ALLOC F(OUT) DA(A.SYSOUT) SHR REUSE"
 "EXECIO 0 DISKW OUT (OPEN"
 do i = 0 to worklen-1 by 32
   w1 = SUBSTR(WORKAREA,i+1,32)
   w2 = C2X(SUBSTR(w1,1,4)) C2X(SUBSTR(w1,5,4)) C2X(SUBSTR(w1,9,4))
   w2 = w2 C2X(SUBSTR(w1,13,4))||'  '||C2X(SUBSTR(w1,17,4))
   w2 = w2 C2X(SUBSTR(w1,21,4)) C2X(SUBSTR(w1,25,4))
C2X(SUBSTR(w1,29,4))
   w1 = TRANSLATE(w1,table)
   i1 = RIGHT(D2X(i),5,'0')
```

```
   i2 = RIGHT(D2X(i+31),5,'0')
   push i1||'-'||i2 '| '||w2 '|'||w1||'| '
   "EXECIO 1 DISKW OUT"
   end
ADDRESS TSO DELSTACK
"EXECIO 0 DISKW OUT (FINIS"
exit
```

6.2 Format of the work area

The work area consists of two parts. The first part is used by CSIREXX to control repeated executions of CSIREXX. It also contains control information for diagnosis. This first area is 2048 bytes in length.

The remainder of the work area is used by IGGCSI00. The format of the returned work area from IGGCSI00 is described in the IBM documentation under the title “Return Work Area Format” in Chapter 11 of the manual,

z/OS DFSMS Managing Catalogs SC23-6853-40

The first section contains the following information.

Offset	***Type***	***Length***	***Description of contents***
0 (0)	Fullword binary	4 (4)	Count of times this has been passed to CSIREXX.
4 (4)	Fullword binary	4 (4)	Length of the entire REXX work area (both sections).
8 (8)	Fullword binary	4 (4)	Length of work area for IGGCSI00.
12 (C)	Fullword binary	4 (4)	Offset from start of IGGCSI00 work area to data for entry.
16 (10)	Fullword binary	4 (4)	Offset from first entry to next entry to be processed.
20 (14)	Fullword binary	4 (4)	Number of IGGCSI00 invocations.
24 (18)	Date	4 (4)	Date of work area creation.
28 (1C)	Time	4 (4)	Time of work area creation.
32 (20)	Character	32 (20)	Identification of the version of the CSIREXX program used to create this work area. Includes program name, date of assembly, time of assembly, local modification id.
64 (40)	Mixed	154 (9A)	Entire CSIFIELD structure used in 2nd parameter of IGGCSI00.
218 (DA)	Character	1280 (500)	Names of FIELDS selected. There is room for 160 field names. Each name is of length 8 bytes, left justified and blank padded.
1498 (5DA)	Character	44 (2C)	Name of catalog being processed.

6.3 VVDS unavailable

There are problems that can be encountered when a VVDS is unavailable.

If a search argument is specified in the key parameter which includes multiple data sets but a VVDS is unavailable (volume offline for example), then a return code will be given to the caller of CSIREXX indicating the error, such as ‘70000000 00000004 C6E20464’.

This return code may be associated with only one catalog entry, but the entire data retrieval is suppressed.

In order to find the offending data set, it is possible to use CSIREXX to read the index using a READNI *verb* parameter, and then subsequently read each catalog entry explicitly using a READ entry. This will enable the identification of the catalog entry at fault.

---------- End of Document ----------