IBM MQ for z/OS

Administration – Introduction to MQ Batch Utilities

IBM Washington Systems Center
Lyn Elkins – elkinsc@us.ibm.com
Mitch Johnson – mitchj@us.ibm.com
Dorothy Quincy – Dorothy.qunicy@ibm.com

An
IBM Washington
Systems Center Wildfire
Workshop

Oh Dear, Oh Dear

- This is SOOOOO dull

Agenda

- MQ for z/OS Batch Utilities
  - CSQJU003 – Change Log and BSDS Utility, primary uses
    - Used to register MQ active logs
    - Deleting information about obsolete MQ active or archive logs
  - CSQ1LOGP – Print information from MQ Logs or BSDS
    - Often used to extract messages from the MQ active or archive log
  - CSQ4LOGS – Sample job to replay and summarize log data
  - CSQUTIL – This is the workhorse of the utilities
    - Used when queue manager is both active and inactive
    - Used to make back-ups and restoring object definitions
    - Copying queues and pagesets
    - Resetting the pages when the log RBA is getting to the limit

CSQJU003

- The queue manager should be down when using this utility
- First use is formatting the MQ log files
  - Log files are used to keep track of units of recovery and provide recovery for persistent messages
  - This program is used primarily to register new logs to the Boot Strap Data Set (BSDS) and delete old archive logs
    - NEWLOG also sets the log to COPY1 or COPY2
  - JCL samples
    - CSQ4BSDS – initial definition of 4 dual log files, including registering them with the BSDS and formatting the logs
    - CSQ4LREC – definition of a new dual log, registering it with the BSDS and formatting the log
      - For some reason uses the same name as the third log file defined in the CSQ4BSDS sample

CSQ1LOGP

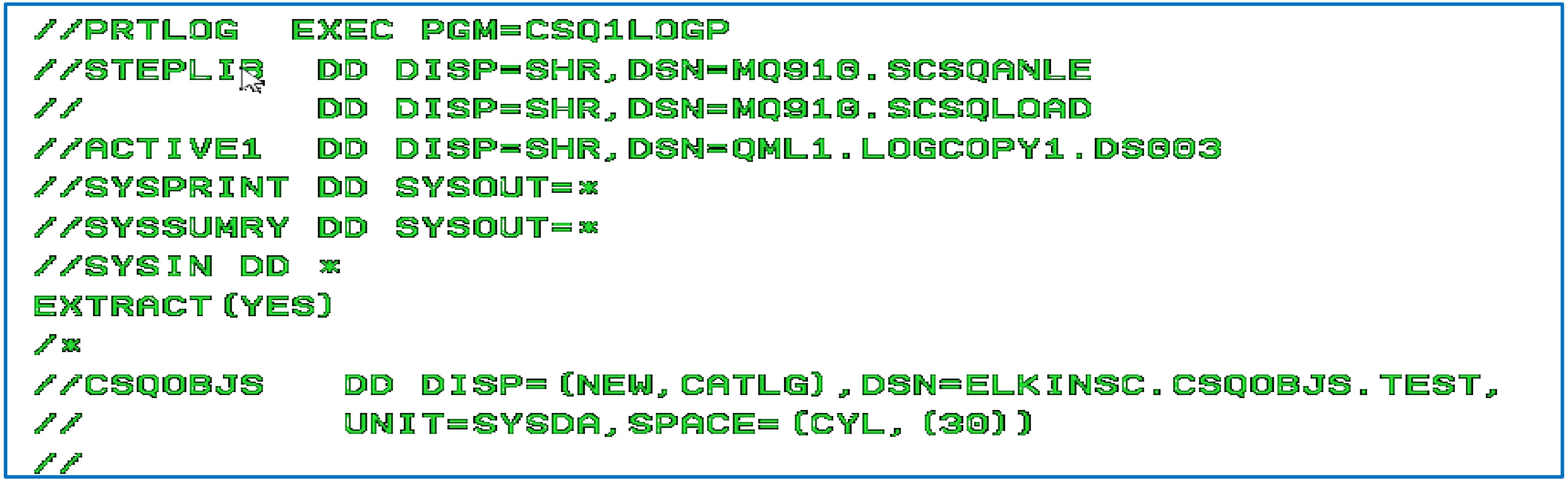
- This utility is primarily used for log analysis, message extract and replay
  - Will work with both active and archive log files
  - There are some limitations of the messages that can be replayed
    - Messages with properties are known not to work.
  - Unique for MQ on z/OS
    - There are commercial providers that provide similar functionality for distributed platforms.
- The Extract function can produce the following output files:
  - CSQBACK – persistent messages that were rolled back
  - CSQCMT – persistent messages that were committed
  - CSQINFLT – persistent messages that remain inflight
  - CSQOBJS – MQ object changes

CSQ4LOGS –The log replay sample

- This is a sample program
  - Source delivered, written in C
  - Can replay messages or provide summary information about the log(s)
  - Might be used to replay messages when there has been an application failure causing messages to be discarded.
  - To summarize activity

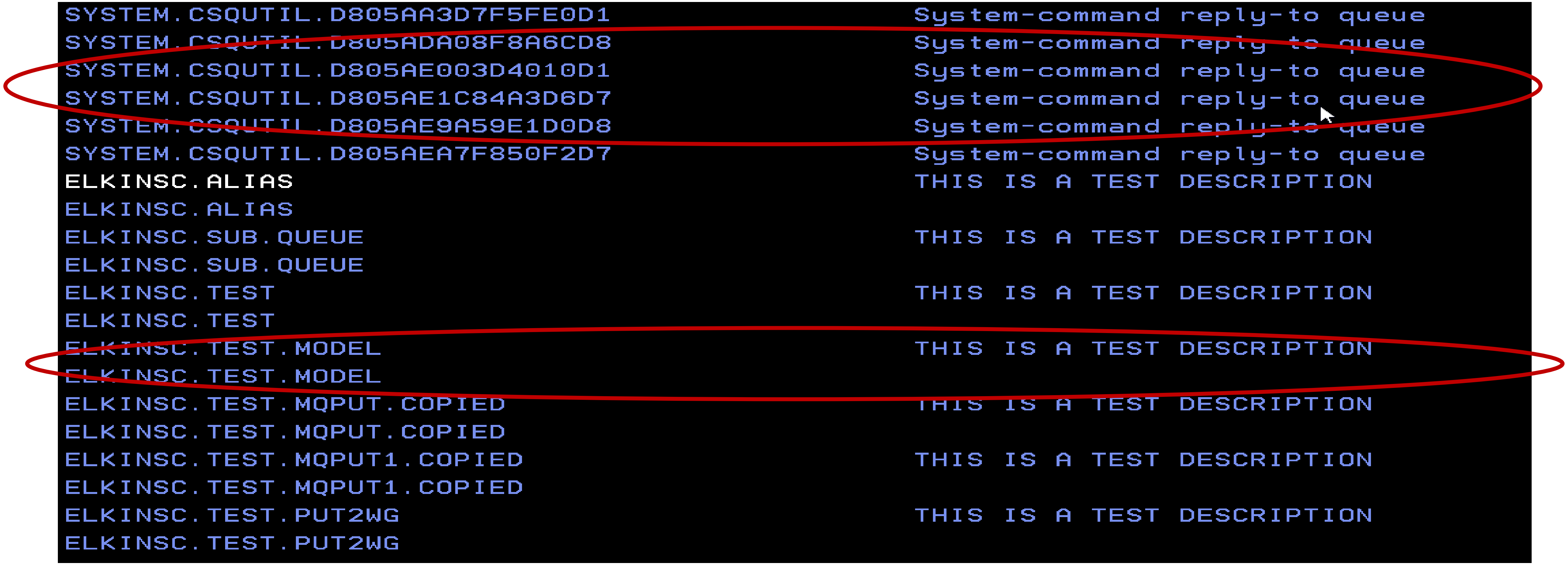
Auditing Object changes – using the CSQOBJS Output

- One use of the LOG Extract program is to audit changes made to object.
  - The log contains records of changes. To get a file run CSQ1LOGP with the CSQOBJS output as shown:



```
//PRTLOG  EXEC PGM=CSQ1LOGP
//STEPLIB  DD DISP=SHR,DSN=MQ910.SCSQANLE
//         DD DISP=SHR,DSN=MQ910.SCSQLOAD
//ACTIVE1  DD DISP=SHR,DSN=QML1.LOGCOPY1.DS003
//SYSPRINT DD SYSOUT=*
//SYSSUMRY DD SYSOUT=*
//SYSIN DD *
EXTRACT(YES)
/*
//CSQOBJS   DD DISP=(NEW,CATLG),DSN=ELKINSC.CSQOBJS.TEST,
//          UNIT=SYSDA,SPACE=(CYL,(30))
//
```

The CSQOBJS Output file



```
SYSTEM.CSQUTIL.D805AA3D7F5FE0D1                 System-command reply-to queue
SYSTEM.CSQUTIL.D805ADA08F8A6CD8                 System-command reply-to queue
SYSTEM.CSQUTIL.D805AE003D4010D1                 System-command reply-to queue
SYSTEM.CSQUTIL.D805AE1C84A3D6D7                 System-command reply-to queue
SYSTEM.CSQUTIL.D805AE9A59E1D0D8                 System-command reply-to queue
SYSTEM.CSQUTIL.D805AEA7F850F2D7                 System-command reply-to queue
ELKINSC.ALIAS                                   THIS IS A TEST DESCRIPTION
ELKINSC.ALIAS
ELKINSC.SUB.QUEUE                               THIS IS A TEST DESCRIPTION
ELKINSC.SUB.QUEUE
ELKINSC.TEST                                    THIS IS A TEST DESCRIPTION
ELKINSC.TEST
ELKINSC.TEST.MODEL                              THIS IS A TEST DESCRIPTION
ELKINSC.TEST.MODEL
ELKINSC.TEST.MQPUT.COPIED                       THIS IS A TEST DESCRIPTION
ELKINSC.TEST.MQPUT.COPIED
ELKINSC.TEST.MQPUT1.COPIED                      THIS IS A TEST DESCRIPTION
ELKINSC.TEST.MQPUT1.COPIED
ELKINSC.TEST.PUT2WG                             THIS IS A TEST DESCRIPTION
ELKINSC.TEST.PUT2WG
```

The CSQOBJS Output file

- The file snippet shows what was changes (in the next section you will be able to see how the changes to the ELKINSC queues were created).
  - This is in human readable format, so if I was looking for a change had been made in the past few minutes I could see that
  - This is run against an active log, it could be done against an archive log
- Other information in the records includes:
  - Date and time stamp (in GMT)
  - User ID
  - What the object change was (DEFINE,ALTER, DELETE)
  - There are records for the AFTER and BEFORE object definition
  - The object types, for example
    - MQAO = Alias Queue
    - MQLO = Local Queue
    - MQRO = Remote Queue
  - If the type is ALIAS the base name is included

The CSQOBJS Output file WARNING!

- In spite of the statement that this can be used to
  “Re-create object definitions for recovery purposes after a major failure.”
- The object definitions and changes are only valid for private queues
  - Shared queue information is stored in Db2, so not logged to the MQ logs

CSQUTIL – The workhorse

- From the IBM MQ Knowledge Center:
  The CSQUTIL utility program is provided with IBM® MQ to help you to perform backup, restoration, and reorganization tasks, and to issue IBM MQ commands.
  - https://www.ibm.com/support/knowledgecenter/SSFKSJ_9.1.0/com.ibm.mq.ref.adm.doc/q088720_.htm

CSQUTIL – One of many ways to send commands to the QMGR

- There are so many ways to get commands into a queue manager these days!
  - MQ Explorer
  - MQ Console
  - CSQINP2 – at queue manager start-up
  - RUNMQSC in client mode
  - CSQUTIL
  - Etc.
- This section will deal with using CSQUTIL for doing all kinds of useful tasks
  - Back-ups and restoration of messages on a queue
  - Back-ups of object definitions
  - Creating other commands

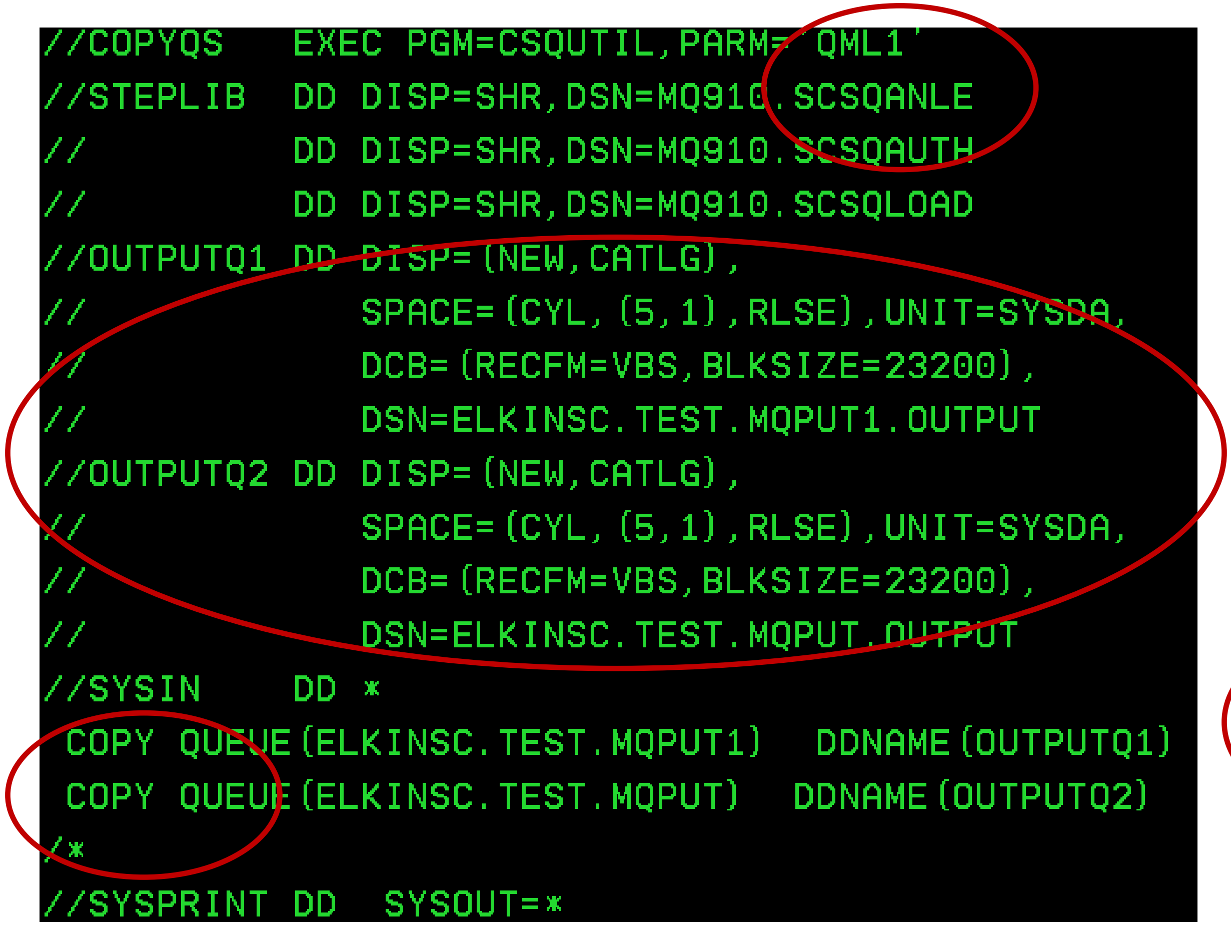
CSQUTIL – backing up a queue

- Ever needed to remove all messages from a queue, but felt like they might be needed again?
  - Especially when those messages may have been flagged as nonpersistent, but drive transactions
- CSQUTIL can copy messages from a queue (or a pageset) onto a dataset as a back-up
  - The queue manager can be running (COPY) or not (SCOPY)
  - When using the COPY functions the queue (or queues) must not be in use
- Also used
  - When preparing for a queue manager shutdown
  - Moving queues to a new storage class to improve performance
  - Isolating queues to prevent clashes
  - Copying messages from production to test
- Recommendation:
  - Have sample tested JCL available to do this copy at all times.
    - Its not difficult, until you are being screamed at

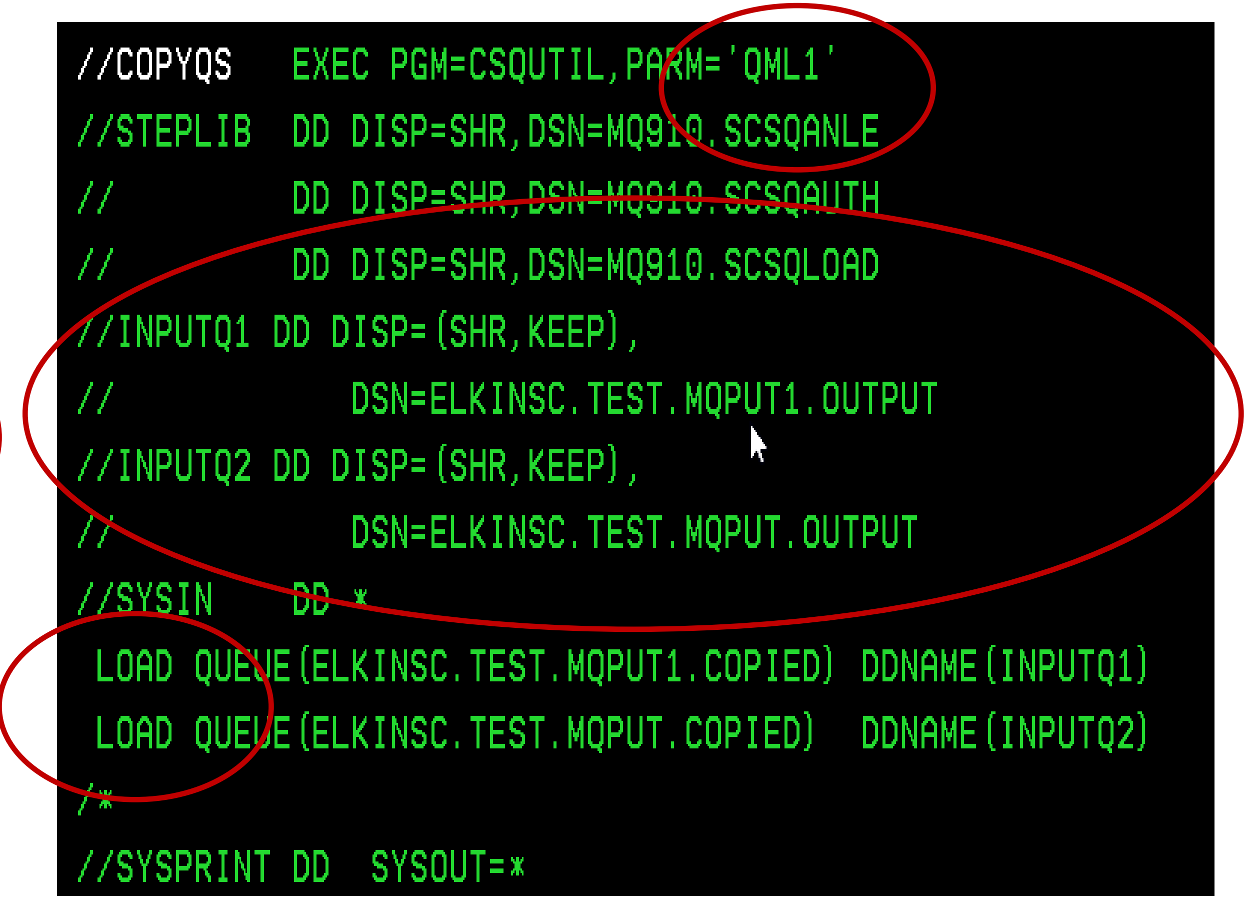
CSQUTIL – Loading (restoring) a queue

- CSQUTIL can restore messages to a queue (or a pageset) from a dataset back-up
  - The queue manager can be running (LOAD) or not (SLOAD)
- Also used
  - After bringing a queue manager back up
  - Moving messages to a new queue
  - Copying captured messages from production to test environments
- Recommendation:
  - Have sample tested JCL available to do a queue load at all times.
    - Just like the copy, its not difficult, until you are being screamed at

Sample Unload and Load JCL



```
//COPYQS    EXEC PGM=CSQUTIL,PARM='QML1'
//STEPLIB   DD DISP=SHR,DSN=MQ910.SCSQANLE
//          DD DISP=SHR,DSN=MQ910.SCSQAUTH
//          DD DISP=SHR,DSN=MQ910.SCSQLOAD
//OUTPUTQ1 DD DISP=(NEW,CATLG),
//             SPACE=(CYL,(5,1),RLSE),UNIT=SYSDA,
//             DCB=(RECFM=VBS,BLKSIZE=23200),
//             DSN=ELKINSC.TEST.MQPUT1.OUTPUT
//OUTPUTQ2 DD DISP=(NEW,CATLG),
//             SPACE=(CYL,(5,1),RLSE),UNIT=SYSDA,
//             DCB=(RECFM=VBS,BLKSIZE=23200),
//             DSN=ELKINSC.TEST.MQPUT.OUTPUT
//SYSIN     DD *
 COPY  QUEUE(ELKINSC.TEST.MQPUT1)   DDNAME(OUTPUTQ1)
 COPY  QUEUE(ELKINSC.TEST.MQPUT)   DDNAME(OUTPUTQ2)
/*
//SYSPRINT DD  SYSOUT=*
```

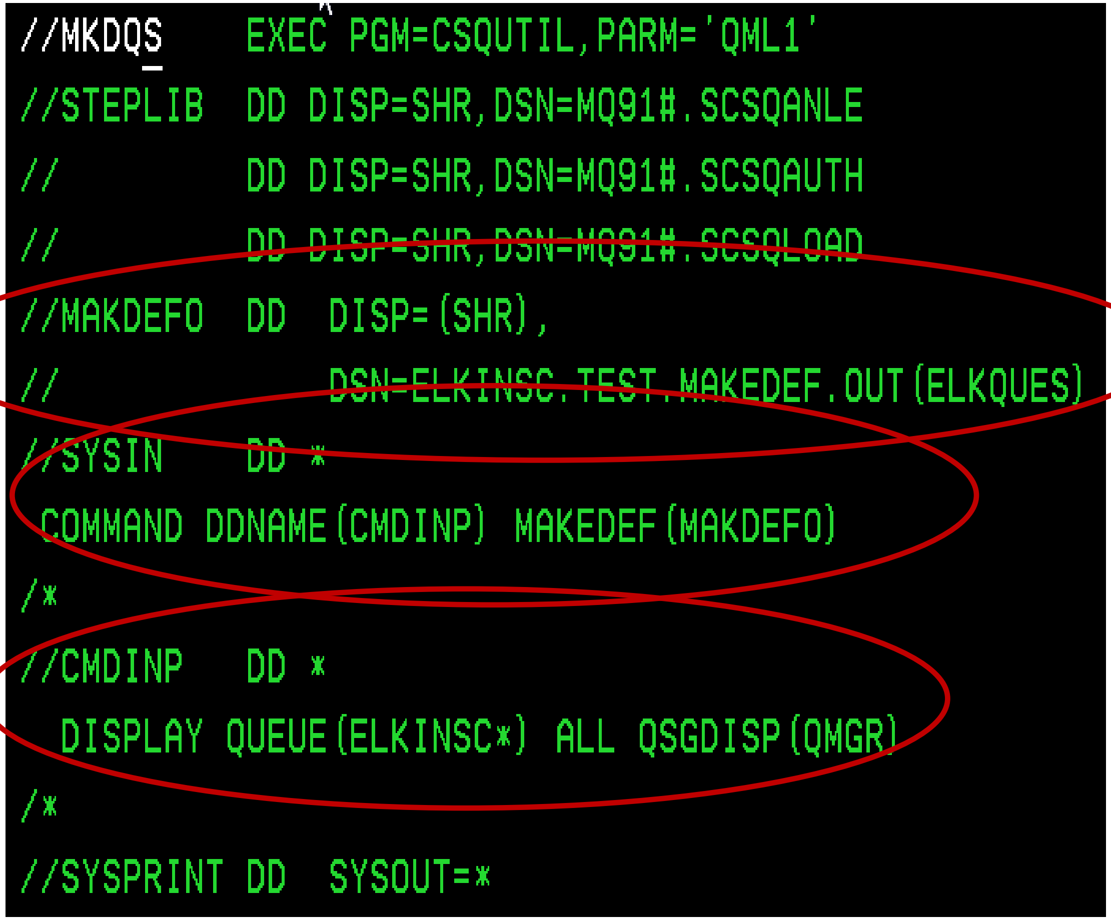


```
//COPYQS    EXEC PGM=CSQUTIL,PARM='QML1'
//STEPLIB   DD DISP=SHR,DSN=MQ910.SCSQANLE
//          DD DISP=SHR,DSN=MQ910.SCSQAUTH
//          DD DISP=SHR,DSN=MQ910.SCSQLOAD
//INPUTQ1 DD DISP=(SHR,KEEP),
//             DSN=ELKINSC.TEST.MQPUT1.OUTPUT
//INPUTQ2 DD DISP=(SHR,KEEP),
//             DSN=ELKINSC.TEST.MQPUT.OUTPUT
//SYSIN     DD *
 LOAD  QUEUE(ELKINSC.TEST.MQPUT1.COPIED)  DDNAME(INPUTQ1)
 LOAD  QUEUE(ELKINSC.TEST.MQPUT.COPIED)   DDNAME(INPUTQ2)
/*
//SYSPRINT DD  SYSOUT=*
```

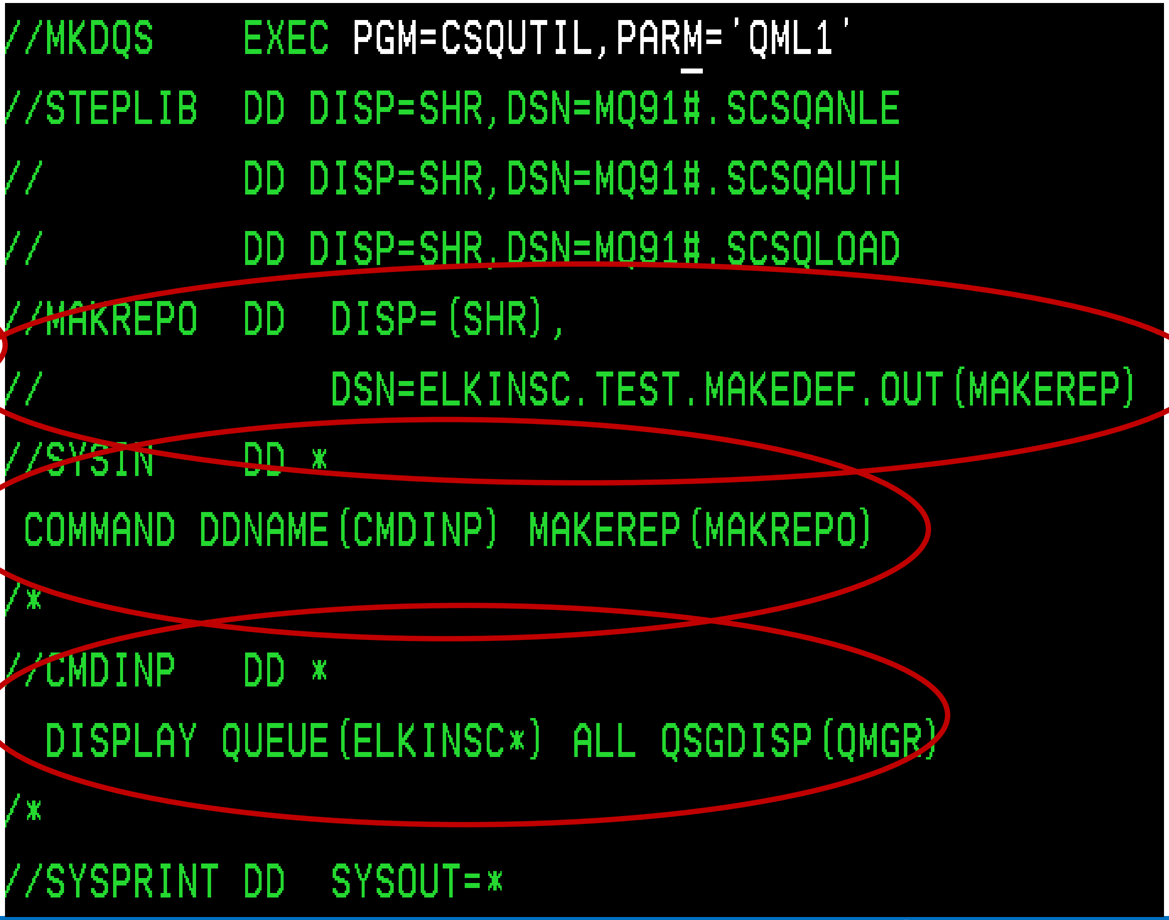
CSQUTIL and the MAKE commands

- MAKE* Commands
  - Generally used in conjunction with a DISPLAY command to generate commands
  - MAKEDEF creates DEFINE commands for all displayed resources with the ‘NOREPLACE’ option
    - Often used for complete back-ups of objects defined to a queue manager
    - Can also be used for capturing subsets for migration between development, QA and production environments
    - Output may be used as part of the CSQINP2 concatenation at queue manager start-up
  - MAKEREP creates DEFINE commands for all displayed resources with the ‘REPLACE’ option
  - MAKEDEL create DELETE commands for all displayed resources
    - May be used to remove objects no longer in use, etc.
  - MAKEALT creates ALTER definitions from the input defined in the DATA keyword

MAKEDEF and MAKEREP Examples

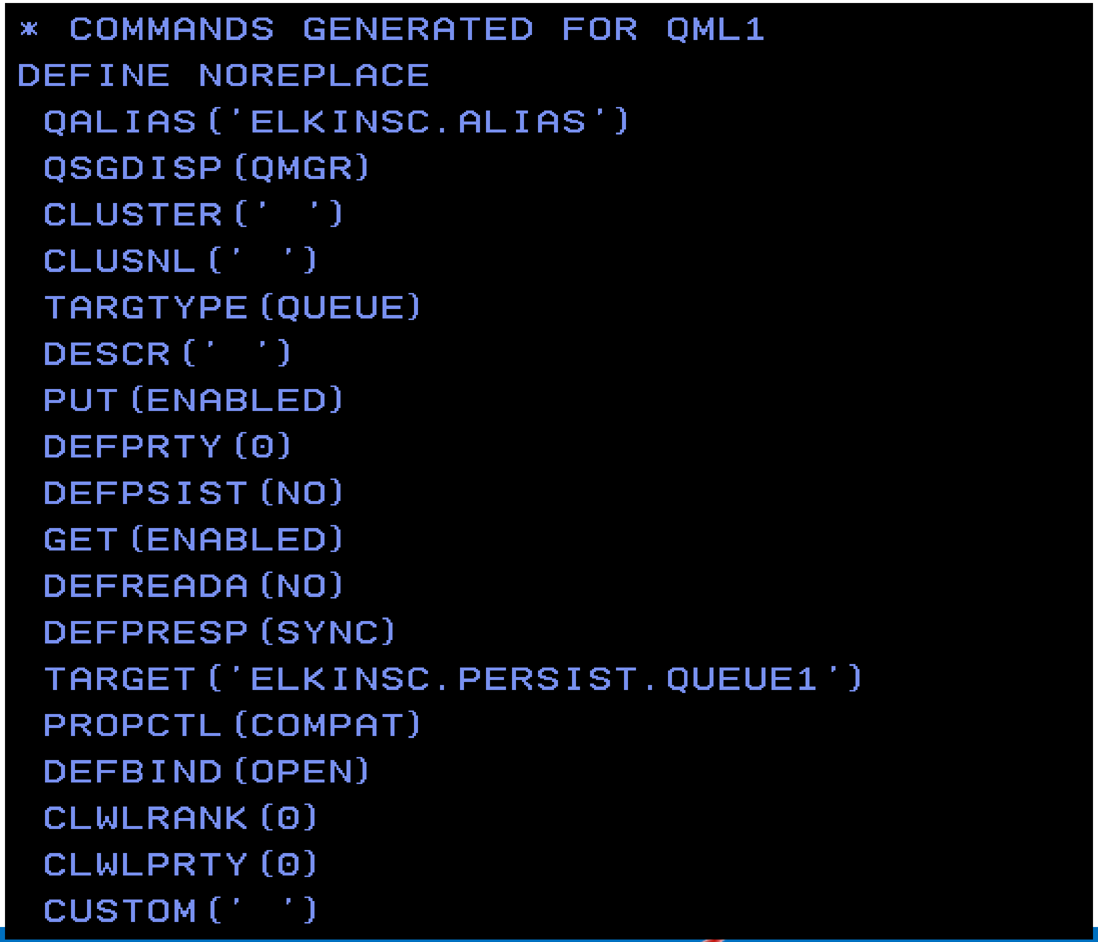


```
//MKDQS    EXEC PGM=CSQUTIL,PARM='QML1'
//STEPLIB  DD DISP=SHR,DSN=MQ91#.SCSQANLE
//         DD DISP=SHR,DSN=MQ91#.SCSQAUTH
//         DD DISP=SHR,DSN=MQ91#.SCSQLOAD
//MAKDEFO  DD  DISP=(SHR),
//             DSN=ELKINSC.TEST.MAKEDEF.OUT(ELKQUES)
//SYSIN    DD *
 COMMAND DDNAME(CMDINP) MAKEDEF(MAKDEFO)
/*
//CMDINP   DD *
  DISPLAY QUEUE(ELKINSC*) ALL QSGDISP(QMGR)
/*
//SYSPRINT DD  SYSOUT=*
```

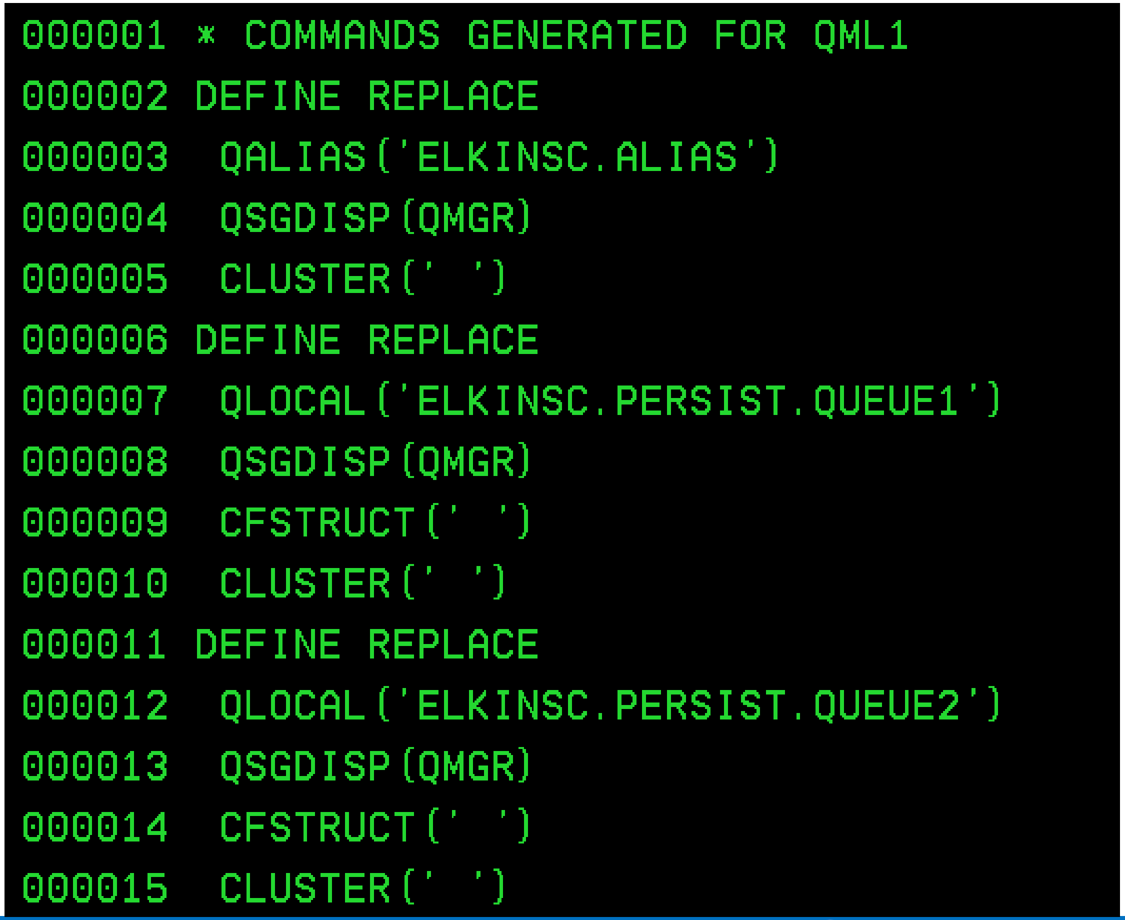


```
//MKDQS    EXEC PGM=CSQUTIL,PARM='QML1'
//STEPLIB  DD DISP=SHR,DSN=MQ91#.SCSQANLE
//         DD DISP=SHR,DSN=MQ91#.SCSQAUTH
//         DD DISP=SHR,DSN=MQ91#.SCSQLOAD
//MAKREPO  DD  DISP=(SHR),
//             DSN=ELKINSC.TEST.MAKEDEF.OUT(MAKEREP)
//SYSIN    DD *
 COMMAND DDNAME(CMDINP) MAKEREP(MAKREPO)
/*
//CMDINP   DD *
  DISPLAY QUEUE(ELKINSC*) ALL QSGDISP(QMGR)
/*
//SYSPRINT DD  SYSOUT=*
```

MAKEDEF and MAKEREP - Output Sample

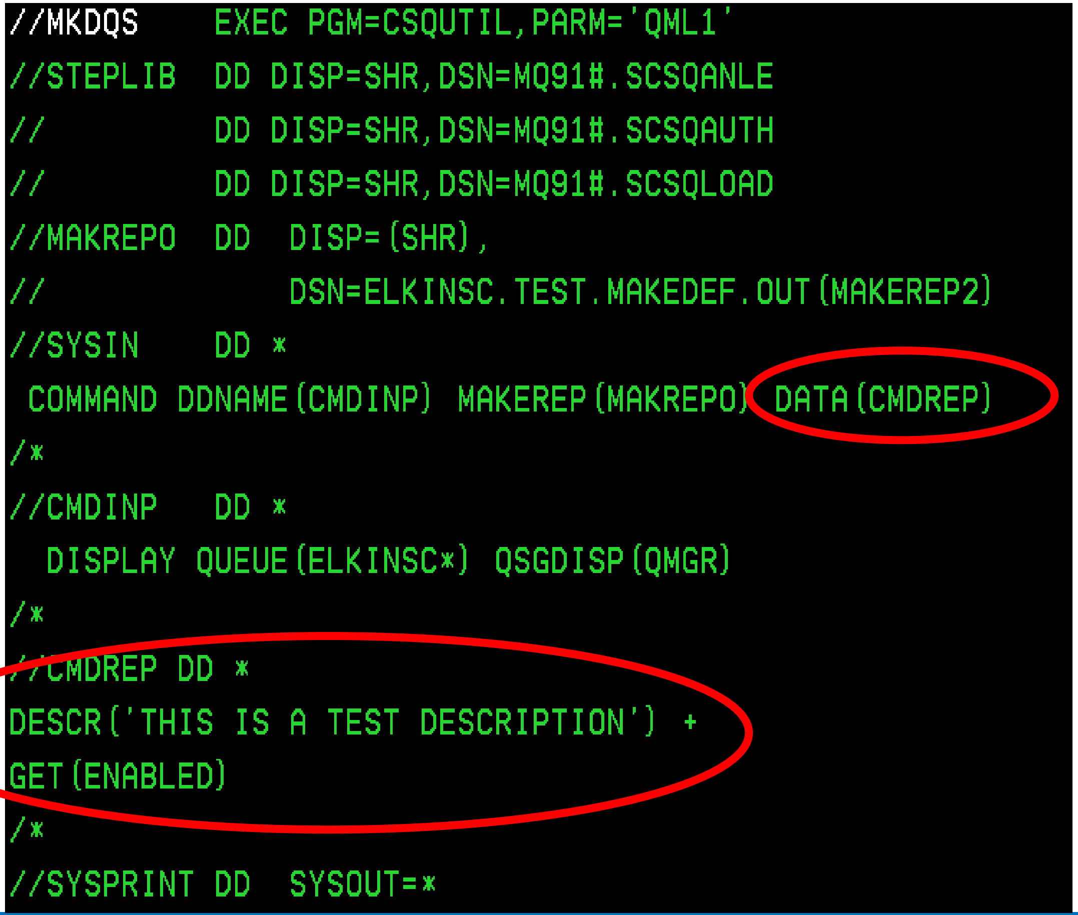


```
* COMMANDS GENERATED FOR QML1
DEFINE NOREPLACE
 QALIAS('ELKINSC.ALIAS')
 QSGDISP(QMGR)
 CLUSTER('  ')
 CLUSNL('  ')
 TARGTYPE(QUEUE)
 DESCR('  ')
 PUT(ENABLED)
 DEFPRTY(0)
 DEFPSIST(NO)
 GET(ENABLED)
 DEFREADA(NO)
 DEFPRESP(SYNC)
 TARGET('ELKINSC.PERSIST.QUEUE1')
 PROPCTL(COMPAT)
 DEFBIND(OPEN)
 CLWLRANK(0)
 CLWLPRTY(0)
 CUSTOM('  ')
```

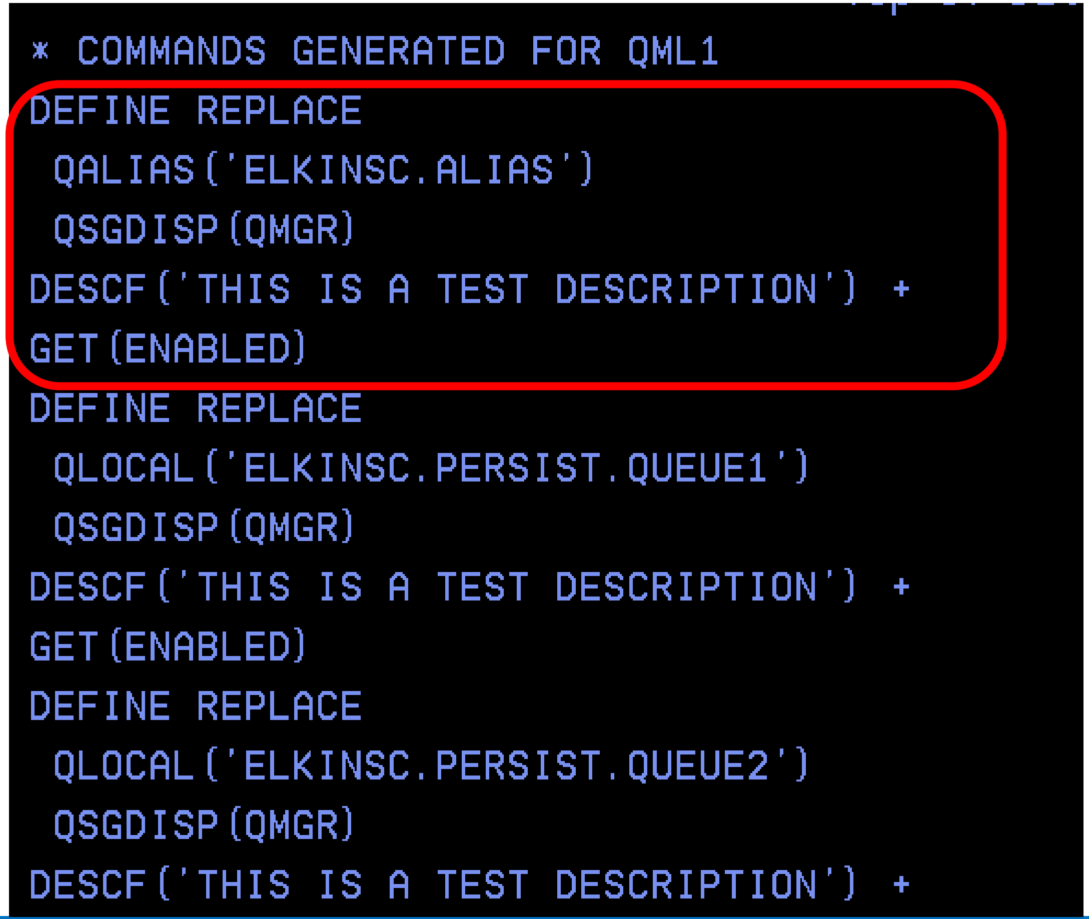


```
* COMMANDS GENERATED FOR QML1
DEFINE REPLACE
 QALIAS('ELKINSC.ALIAS')
 QSGDISP(QMGR)
 CLUSTER('  ')
DEFINE REPLACE
 QLOCAL('ELKINSC.PERSIST.QUEUE1')
 QSGDISP(QMGR)
 CFSTRUCT('  ')
 CLUSTER('  ')
DEFINE REPLACE
 QLOCAL('ELKINSC.PERSIST.QUEUE2')
 QSGDISP(QMGR)
 CFSTRUCT('  ')
 CLUSTER('  ')
```

MAKEREP – hummm, That didn’t look right

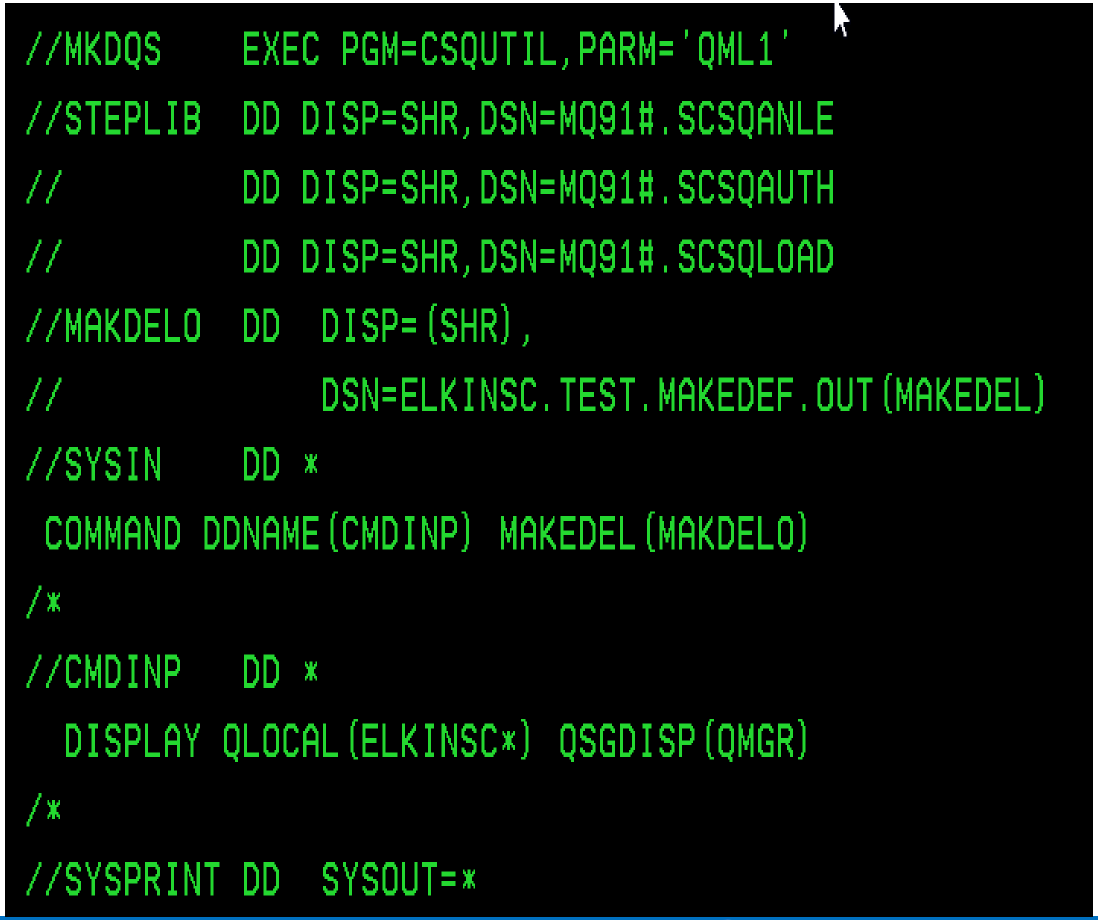


```
//MKDQS    EXEC PGM=CSQUTIL,PARM='QML1'
//STEPLIB  DD DISP=SHR,DSN=MQ91#.SCSQANLE
//         DD DISP=SHR,DSN=MQ91#.SCSQAUTH
//         DD DISP=SHR,DSN=MQ91#.SCSQLOAD
//MAKREPO  DD  DISP=(SHR),
//             DSN=ELKINSC.TEST.MAKEDEF.OUT(MAKEREP2)
//SYSIN    DD *
 COMMAND DDNAME(CMDINP) MAKEREP(MAKREPO) DATA(CMDREP)
/*
//CMDINP   DD *
  DISPLAY QUEUE(ELKINSC*) QSGDISP(QMGR)
/*
//CMDREP DD *
DESCR('THIS IS A TEST DESCRIPTION') +
GET(ENABLED)
/*
//SYSPRINT DD  SYSOUT=*
```

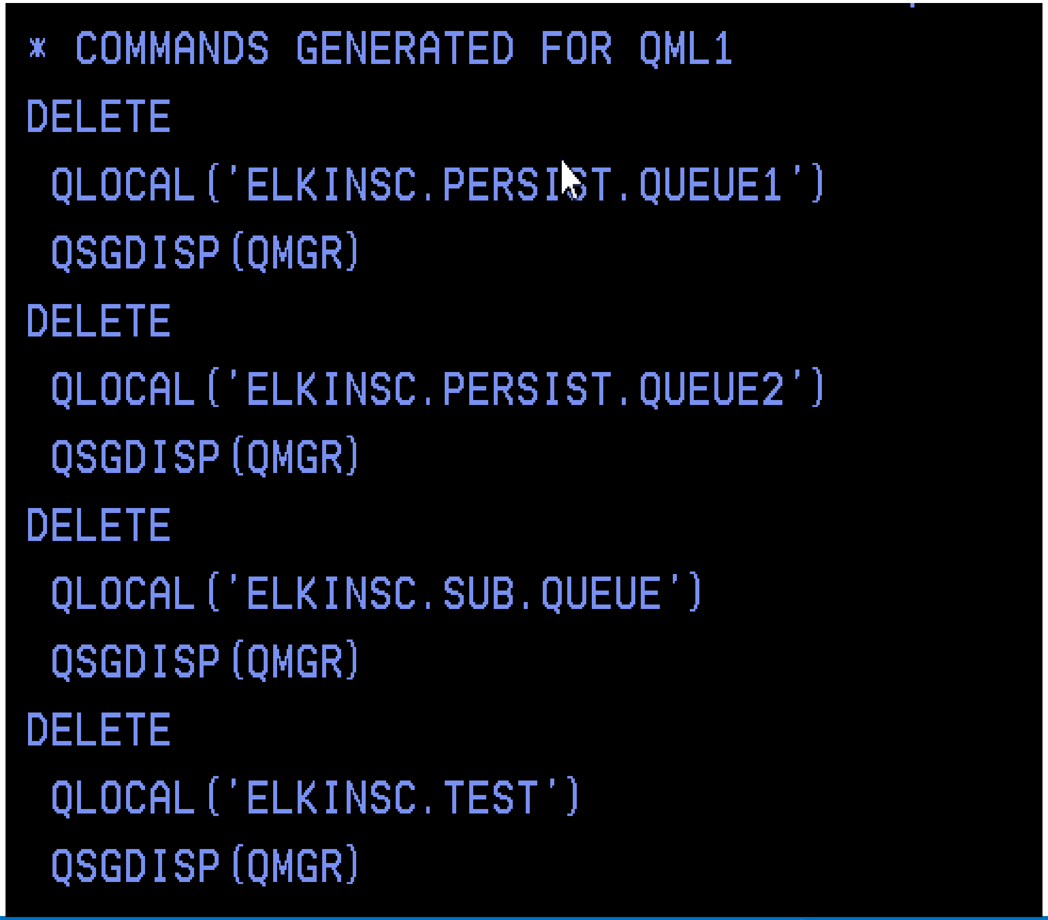


```
* COMMANDS GENERATED FOR QML1
DEFINE REPLACE
 QALIAS('ELKINSC.ALIAS')
 QSGDISP(QMGR)
DESCF('THIS IS A TEST DESCRIPTION') +
GET(ENABLED)
DEFINE REPLACE
 QLOCAL('ELKINSC.PERSIST.QUEUE1')
 QSGDISP(QMGR)
DESCF('THIS IS A TEST DESCRIPTION') +
GET(ENABLED)
DEFINE REPLACE
 QLOCAL('ELKINSC.PERSIST.QUEUE2')
 QSGDISP(QMGR)
DESCF('THIS IS A TEST DESCRIPTION') +
```

MAKEDEL Examples



```
//MKDQS    EXEC PGM=CSQUTIL,PARM='QML1'
//STEPLIB  DD DISP=SHR,DSN=MQ91#.SCSQANLE
//         DD DISP=SHR,DSN=MQ91#.SCSQAUTH
//         DD DISP=SHR,DSN=MQ91#.SCSQLOAD
//MAKDELO  DD  DISP=(SHR),
//             DSN=ELKINSC.TEST.MAKEDEF.OUT(MAKEDEL)
//SYSIN    DD *
 COMMAND DDNAME(CMDINP) MAKEDEL(MAKDELO)
/*
//CMDINP   DD *
  DISPLAY QLOCAL(ELKINSC*) QSGDISP(QMGR)
/*
//SYSPRINT DD  SYSOUT=*
```



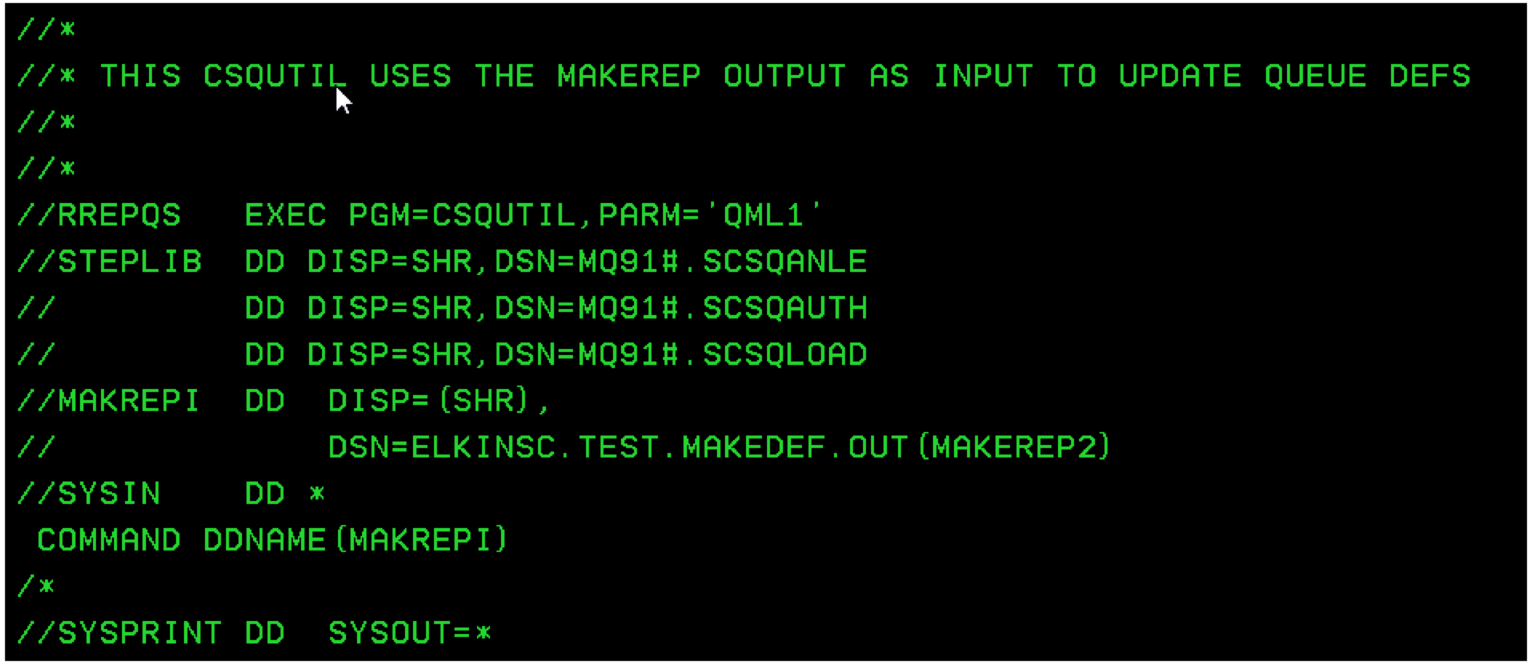
```
* COMMANDS GENERATED FOR QML1
DELETE
 QLOCAL('ELKINSC.PERSIST.QUEUE1')
 QSGDISP(QMGR)
DELETE
 QLOCAL('ELKINSC.PERSIST.QUEUE2')
 QSGDISP(QMGR)
DELETE
 QLOCAL('ELKINSC.SUB.QUEUE')
 QSGDISP(QMGR)
DELETE
 QLOCAL('ELKINSC.TEST')
 QSGDISP(QMGR)
```

In conclusion

- There is much, much more!
- But I hope this has given enough background to be useful.
  - Or enough to make a new admin familiar with some areas to explore.

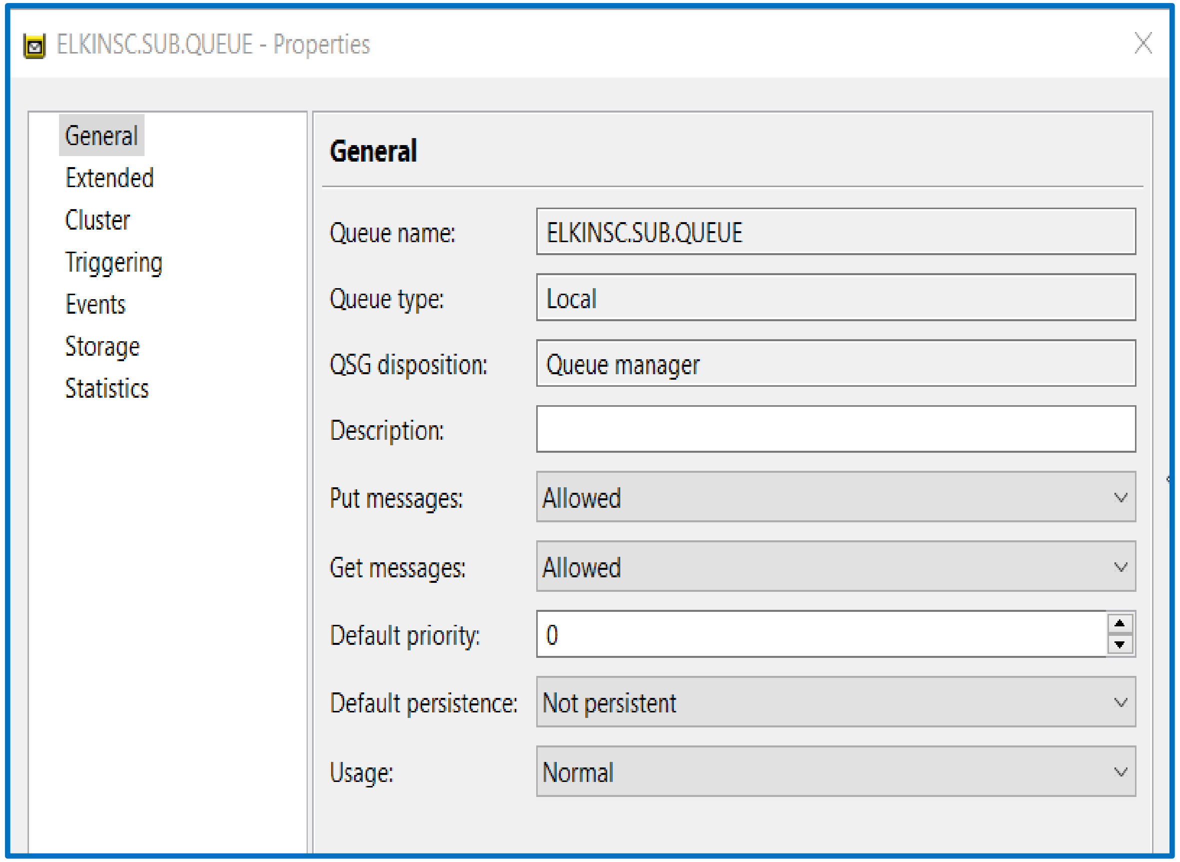
To use the commands you have created

- CSQUTIL allows you to input the commands created as shown



```
//*
//* THIS CSQUTIL USES THE MAKEREP OUTPUT AS INPUT TO UPDATE QUEUE DEFS
//*
//*
//RREPQS   EXEC PGM=CSQUTIL,PARM='QML1'
//STEPLIB  DD DISP=SHR,DSN=MQ91#.SCSQANLE
//         DD DISP=SHR,DSN=MQ91#.SCSQAUTH
//         DD DISP=SHR,DSN=MQ91#.SCSQLOAD
//MAKREPI  DD  DISP=(SHR),
//             DSN=ELKINSC.TEST.MAKEDEF.OUT(MAKEREP2)
//SYSIN    DD *
 COMMAND DDNAME(MAKREPI)
/*
//SYSPRINT DD  SYSOUT=*
```

Before and After



ELKINSC.SUB.QUEUE - Properties

General

Queue name: ELKINSC.SUB.QUEUE

Queue type: Local

QSG disposition: Queue manager

Description:

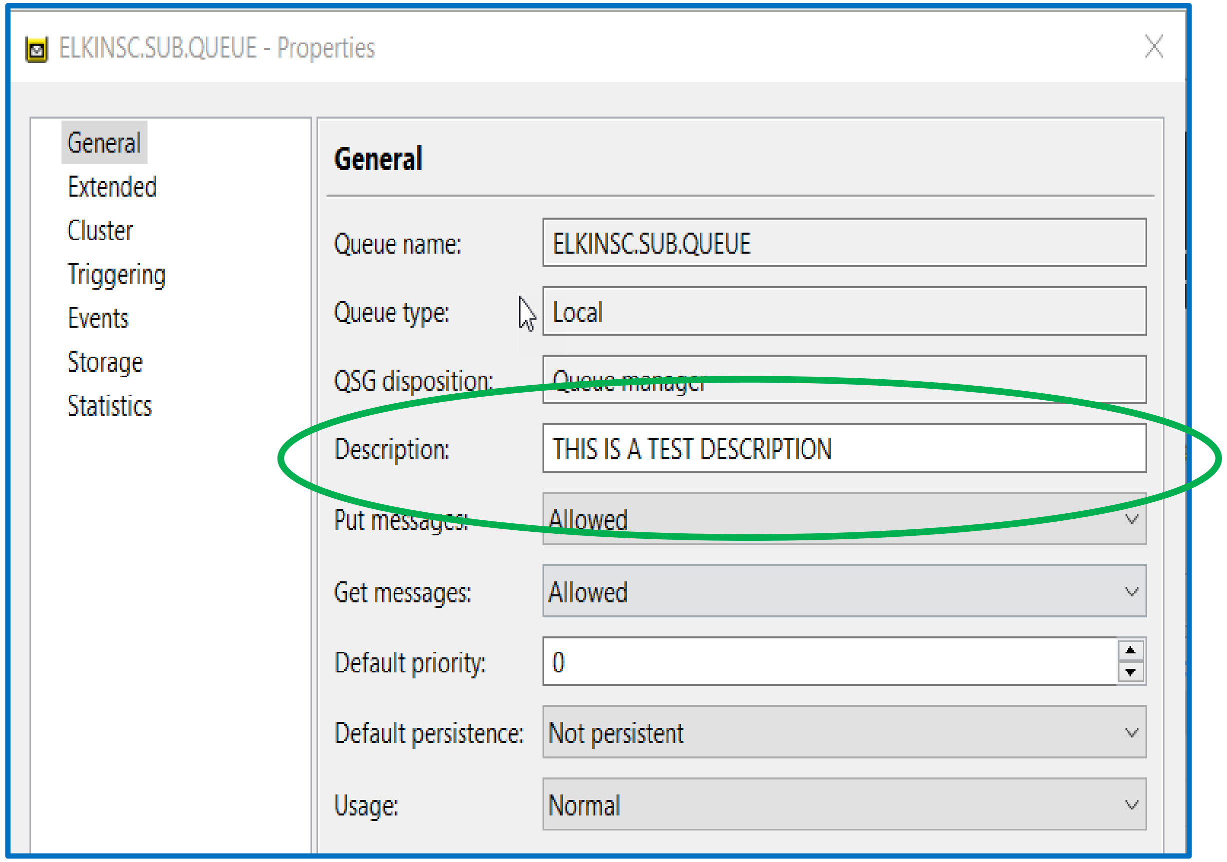
Put messages: Allowed

Get messages: Allowed

Default priority: 0

Default persistence: Not persistent

Usage: Normal



ELKINSC.SUB.QUEUE - Properties

General

Queue name: ELKINSC.SUB.QUEUE

Queue type: Local

QSG disposition: Queue manager

Description: THIS IS A TEST DESCRIPTION

Put messages: Allowed

Get messages: Allowed

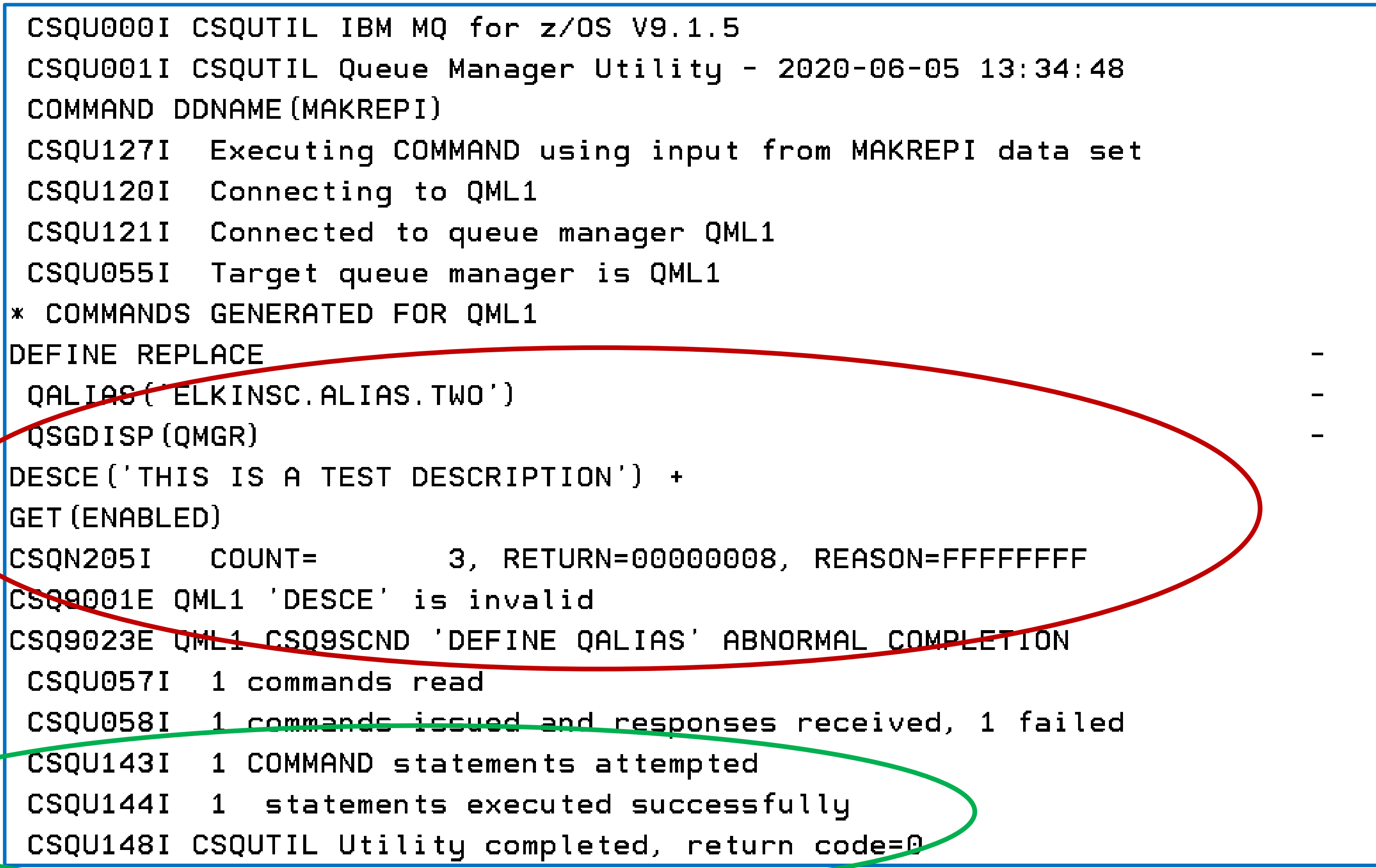
Default priority: 0

Default persistence: Not persistent

Usage: Normal

CSQUTIL Gotchas!

- You get a ‘0’ return code
  - Even when no commands work



```
 CSQU000I CSQUTIL IBM MQ for z/OS V9.1.5
 CSQU001I CSQUTIL Queue Manager Utility - 2020-06-05 13:34:48
 COMMAND DDNAME(MAKREPI)
 CSQU127I  Executing COMMAND using input from MAKREPI data set
 CSQU120I  Connecting to QML1
 CSQU121I  Connected to queue manager QML1
 CSQU055I  Target queue manager is QML1
* COMMANDS GENERATED FOR QML1
DEFINE REPLACE                                                  -
 QALIAS('ELKINSC.ALIAS.TWO')                                    -
 QSGDISP(QMGR)                                                  -
DESCE('THIS IS A TEST DESCRIPTION') +
GET(ENABLED)
CSQN205I   COUNT=       3, RETURN=00000008, REASON=FFFFFFFF
CSQ9001E QML1 'DESCE' is invalid
CSQ9023E QML1 CSQ9SCND 'DEFINE QALIAS' ABNORMAL COMPLETION
 CSQU057I  1 commands read
 CSQU058I  1 commands issued and responses received, 1 failed
 CSQU143I  1 COMMAND statements attempted
 CSQU144I  1  statements executed successfully
 CSQU148I CSQUTIL Utility completed, return code=0
```

Other information

- If the execution cannot connect to the queue manager or there are other critical errors (resource not found for example), then a bad return code is expected.