```
CCLDEBUG.DOC
(1)call trace  (2)ccldefault logical  (3)memory traces  (4)pool types  (5)Trace command  (6)Modules
The CCLDEFAULT_CCL logical should just be set as a process logical and not in ccldir:ccldomain.dat.
CCLDEFAULT_CCL="NNNN,TRACE(WARNING)"
CCLDEFAULT_CCL="NNN6,TRACE(WARNING,MEMCHK)"
CCLDEFAULT_CCL="NNN7,TRACE(WARNING,MEMCHK)"
Preferred debug ccldefault settings for interactive ccl and script server which can be set in ccldir:ccldomain.dat init
file.
CCLDEFAULT="NNN6,TRACE(RDBCOMMENT,MEMCHK,MEMCHK2,LOCK)"
CCLDEFAULT_0000="NNNN,TRACE(WARNING)"
CCLDEFAULT_0051="NNN6,TRACE(RDBCOMMENT,MEMCHK,MEMCHK2,LOCK),INSTANCE(1)"
CCLDEFAULT_0051="NNN6,TRACE(RDBCOMMENT,DEPRECATED,COST,CODECOVER=0,MEMCHK,MEMCHK2,LOCK),INSTANCE(1)"
================================================================================================
>>>1)CALL TRACE(<trace_option_num>))
      01: set ccloutput to stdout
     *02: set ccloutput to ccllogfile
      03: show active subroutines without tree
      04: show active subroutines with tree
      05: show env
     *06: pool check
     *07: show cache
     *08: set ccloutput and ccllogfile to stdout
      09: debug a script break routine
     *10: pool check with print detail memory info
      11: get thread stack size
      12: forced logfile flush
     *13: pool check with print summary memory info
      14: print current symbol table
     *15: check all active record structures and check all memory objects
      16: display the current count of an object type
      17: echo all active label names and levels
      18: show env without uar subroutines
      19: show symbol info for call trace(19,<symbol_num>)
     *20: show env with trace and settings only
     *21: check an individual active record structure
      22: show full cclversion from banner
      23: show all subrtn and check for valid tree and print ccl tree
      24: show ccl subroutines and check for valid tree and print ccl tree
      25: check all memory if memory trace active
     *26: one mode to check all memory
================================================================================================
FreePages(freepages,last_freepages_used)Pool(bucket_total,bucket_sym,bucket_tree,poolsz)Misc(misc_total,miscsz)
    Rdb(ccldsri_mem_info)Fun(parfun_cnt,varfun_cnt)Ret(varreturn_cnt)Sub(subrtn_cnt)Prg(prg_cnt)Rng(rng_cnt)
    Rec(rec_cnt,varlist_cnt,varchar_cnt,vclient_cnt,memvarlist_cnt,memvarlist2_cnt,memvarchar_cnt,memvarchar2_cnt)
srv memcost&memcost2: CCLSET_DBSET mem(freepages) mode(mode) ekm(ekm) pool(total) sym(last_sym,bucket_sym,used_sym)
mark(mark)
ccl memcost&memcost2: MEMCOST mem(freepages) pool(total) sym(last_sym,bucket_sym,used_sym)
================================================================================================
```

```
>>>2)CCLDEFAULT LOGICAL
  Handle the ccldefault setting before startup, used for internal testing and debugging.

<1:dio><2:sigbus><3:sigfpe><4:mem>[<5:diskerr><6:rtlver><7:findimage><8:dmsprint><9:rdbmsinit>]{,<pooltyp>}[,<trace>][,
<inst>]
  Check for CCLDEFAULT[_<srvnum>] first and if not found check for CCLDEFAULT.
  CCLDEFAULT_0000 can be used for just setting for interactive ccl.
  ccldefault may be a process, group, or system logical, ccldefault logical contains options denoted by Y or N.
  Y will override the default, anything else will be ignored.
  <memory>    will allow N,Y,1,2,3,4,5,6,7
  <pooltyp>   grouping of 2 digit numbers separated by commas
              to allow the memory4 for individual show allocate,free of a memory type
  <traces>    TRACE(<trace>{,<trace>})
  <instances> INSTANCE(<instance>{,<instance>})
  Examples:
   define/group ccldefault      "NNNY"                    ;turn on memory
   define/group ccldefault_0000 "NNNN,TRACE(RDBCOMMENT)"  ;send rdbcomment
   define/group ccldefault_0051 "NNN6,55,56,57"           ;varchar,varfun,varlist
   define/group ccldefault_0051 "NNN6,02,46,51,55,56,57"  ;altlist,recstruct,strlink,varchar,varfun,varlist
   define/group ccldefault_0051 "NNN6,11,38"              ;dictable,progcache
   define/group ccldefault_0051 "NNN6,TRACE(COST,MEMCOST,MEMABORT),INSTANCE(1,2)" ;instances 1,2 of 51 server.
================================================================================================
>>>3)MEMORY TRACES
    MEMABORT    ;abort process if memory corruption detected (memory trace must also be active)
    MEMCHK      ;memory checking (sets multiple traces)
    MEMCOST     ;show memory cost, also needs to be on for extensive memory checking
    MEMCOST2    ;control details level 2 for memcost
    MEMCOST3    ;control details level 3 for memcost
    MEMDEBUG1   ;memory debug for persist variable declares and subrtn copy from parent to child
    MEMDEBUG2   ;memory debug for lexscan
    MEMDEBUG3   ;memory debug for tree
    MEMDEBUG4   ;memory debug for longjmp
    MEMORY      ;pad memory on alloc, check for corruption,overflow on free
    MEMORY2     ;varchar and varlist memory check
    MEMORY3     ;treepool and sympool memory check
    MEMORY4     ;memory(detail memory show alloc,free) nomemory(show pool stats)
    MEMORY5     ;show memory pools on fatal error and at exit time
    MEMORY6     ;show specific alloc by object, periodic mem check, show all memory at exit
    MEMORY7     ;dont save objects in memcheck list
================================================================================================
>>>4)Pool types (varlist=2,client=5,recstruct=46,symbol=53,varchar=55,varfun=56,varlist2=57,varchar2=59)
01   CCLPOOL_SYS
02   CCLPOOL_VARLIST
03   CCLPOOL_CCV
04   CCLPOOL_CEF
05   CCLPOOL_CLIENT
06   CCLPOOL_CXT
07   CCLPOOL_DEBUG
```

```
08   CCLPOOL_DICFILE
09   CCLPOOL_DICMACREC
10   CCLPOOL_DICRECTYPE
11   CCLPOOL_DICTABLE
12   CCLPOOL_DICTABLEATTR
13   CCLPOOL_DISPLAYER
14   CCLPOOL_DSRI
15   CCLPOOL_EKS
16   CCLPOOL_EKSLIST
17   CCLPOOL_FIXUP
18   CCLPOOL_GENERATE
19   CCLPOOL_HELPBUF
20   CCLPOOL_INCLUDEPRE
21   CCLPOOL_LEXINPUT
22   CCLPOOL_MEMPROGCACHE
23   CCLPOOL_MEMRNGCACHE
24   CCLPOOL_MEMQUALATTR
25   CCLPOOL_MEMCLAUSE
26   CCLPOOL_MEMKEYPRED
27   CCLPOOL_MEMTREE
28   CCLPOOL_MEMTREEDATA
29   CCLPOOL_MEMTARLIST
30   CCLPOOL_MEMPRHEIAR
31   CCLPOOL_MEMGVAR
32   CCLPOOL_MEMPROGLABEL
33   CCLPOOL_MENU
34   CCLPOOL_PAGEBUF
35   CCLPOOL_PAINT
36   CCLPOOL_PAINTWIN
37   CCLPOOL_PARSER
38   CCLPOOL_PROGCACHE
39   CCLPOOL_PROGCACHESTORE
40   CCLPOOL_QRYWORK
41   CCLPOOL_QUAL
42   CCLPOOL_RDBMSBIND
43   CCLPOOL_RECALL
44   CCLPOOL_RECBUF
45   CCLPOOL_RECHEAD
46   CCLPOOL_RECSTRUCT
47   CCLPOOL_REPORTVAR
48   CCLPOOL_RES
49   CCLPOOL_REPORTBUF
50   CCLPOOL_SMG
51   CCLPOOL_STRLINK
52   CCLPOOL_SUBRTN
53   CCLPOOL_SYMBOL
54   CCLPOOL_UARPARAM
55   CCLPOOL_VARCHAR
```

```
56   CCLPOOL_VARFUN
57   CCLPOOL_VARLIST2
58   CCLPOOL_AMI
59   CCLPOOL_VARCHAR2
60   CCLPOOL_MEMSRVREQLNK
61   CCLPOOL_SUBRTNPACK
62   CCLPOOL_VARRETURN
63   CCLPOOL_CODECOVERLNK
64   CCLPOOL_XMLBUF
65   CCLPOOL_VARFUNLIST
66   CCLPOOL_VARRETURNLIST
67   CCLPOOL_SYMBOL2
68   CCLPOOL_METAPHONE
69   CCLPOOL_OCI
70   CCLPOOL_SYSLOG
================================================================================================
>>>5)Trace commands
    ALTER             ;on if skip echo of record for cclparse input command
    ALTERLISTINIT     ;apply alterlist init of 1 to null varlist of set of rec struct
    AUTOLOCK          ;at conclusion of user startup script then turn on lock trace
    CALLECHO          ;on: enable call echo, off: disable call echo
    CALLECHOLOCK      ;on: SET TRACE CALLECHO can not be changed: SET TRACE CALLECHO can be changed
    CALLECHOTAG       ;call echo with tag (program and location)
    CALLPARSER        ;max length for call parser
    CHECKUAR          ;check uar calls
    CODECOVER         ;enable code coverage for scripts compiled with debug
    COST              ;trace for show of ccl cost of queries and programs
    COST2             ;trace for show of ccl cost of queries and programs
    DDL               ;trace data definition commands
    DEPRECATED        ;flag as errors use of any deprecated features
    DISCERNCACHE      ;use discerncache shared memory cache
    DISCERNCACHELOG   ;debug logging for discerncache shared memory cache
    ECHOINPUT         ;on if echo of cclparse input, off if no echo of cclparse input
    ECHOINPUT2        ;on if echo of cclparse2 input, off if no echo of cclparse2 input
    ECHOPROG          ;echo out execute program names
    ECHOPROGALL       ;echo out all execute and sub calls
    ECHOPROGSUB       ;echo out execute sub program names
    ECHORECDEBUG      ;echo debug for record defines
    ECHORECORD        ;echo record definition
    ECHOSUB           ;echo out subroutine names
    EKM               ;on if this is ekmodule debug, off if not ekmodule trace
    EKM2              ;on if this is ekmodule2 debug, off if not ekmodule2 trace
    ERROR             ;trace ccl errors
    ERRORCLEAR        ;if on then wont clear errors for subprogram for ERROR() ccl user function
    ERRORCLEARCOM     ;if on then wont clear errors for command (except execute program)
    EXITCLEANUP       ;if on then cleanup all ccl allocated memory on exit
    FLUSH             ;on if flush of rtl file after each cclparse call
    GRAMMAR           ;trace grammar builder
```

```
HIPAA             ;hipaa auditing enabled
HIPAA2            ;hipaa2 auditing enabled
IMAGE             ;on: disable exception handler for find image symbol
ISAM              ;trace for isam calls
ISAM2             ;trace for isam calls
ISAMLOCK          ;enable isam locking
LOADPERSIST       ;loadpersist option
LOCK              ;lock out all set trace changes (dont tell anyone about this option)
MEMABORT          ;abort process if memory corruption detected (memory trace must also be active)
MEMCHK            ;memory checking (sets multiple traces)
MEMCOST           ;show memory cost, also needs to be on for extensive memory checking
MEMCOST2          ;control details level 2 for memcost
MEMCOST3          ;control details level 3 for memcost
MEMDEBUG1         ;memory debug for persist variable declares and subrtn copy from parent to child
MEMDEBUG2         ;memory debug for lexscan
MEMDEBUG3         ;memory debug for tree
MEMDEBUG4         ;memory debug for longjmp
MEMORY            ;pad memory on alloc, check for corruption,overflow on free
MEMORY2           ;varchar and varlist memory check
MEMORY3           ;treepool and sympool memory check
MEMORY4           ;memory(detail memory show alloc,free) nomemory(show pool stats)
MEMORY5           ;show memory pools on fatal error and at exit time
MEMORY6           ;show specific alloc by object, periodic mem check, show all memory at exit
MEMORY7           ;dont save objects in memcheck list
MEMSORT           ;default to using memsort if none specified
ODBC              ;on if this is odbc access, off if not odbc access
PLANCL            ;trace query plan
PRINTDSP          ;trace for disable/enable print from displayer
PRINTNOSETUP      ;indicates print setup module not supported
QUERY             ;trace query statistics
QUERYLOCK         ;on if controlc lockout for query, off if controlc nolockout for query
QUERYTIMEOUT      ;default timeout for query which dont specify with time=<val>
RANGECACHE        ;make temp ranges permament except when MAXRANGECACHE is exceeded
RDBARRAYFETCH     ;set default number for array fetch to use from rdb
RDBARRAYINSERT    ;array insert to user for rdb insert
RDBBIND           ;trace for debug of rdb bind input variables
RDBBINDCONS       ;on: bind rdb constants nested select, off: dont bind rdb constants nested select
RDBBUFFER         ;rdb buffer allocation size
RDBCOMMENT        ;on to send server number in comment, progname still sent regardless
RDBDEBUG          ;trace for debug of rdb
RDBDEBUG1         ;rdb debug misc 1
RDBDEBUG2         ;rdb debug misc 2
TRACE RDBPASSCONS       ;on: pass rdb constants(nested or unnested), off: bind rdb constants
RDBPLAN           ;trace rdb plan
RDBPROGRAM        ;send hint for program name originating rdb statement (obsolete)
RDBSKIPCAST       ;skip casting for db2
RDBSRVERROR       ;convert rdbms server fatal errors to normal errors
RDBTRIMBIND       ;on if using implicit trimbind for insert,update set target list
```

```
    REBUILD           ;trace for rebuilding dictionary so ignore any EXECUTE
    RECPERSIST        ;on if record defined in script is persistent, off if not
    REFLOG            ;turn on/off cclreflog
    REGISTERPRG       ;require register of compiled programs (objects=P)
    SAVEPOINT         ;savepoint for trace settings
    SCAN              ;trace lexscan
    SERVER            ;on if this is server, off no server
    SHOWUAR           ;show uar calls
    SHOWUARPAR        ;show uar parameters before uar call
    SHOWUARPAR2       ;show uar parameters after uar call
    SHOWUARPAR3       ;trace for subset of uars that allocate and free handles (showuar must be on)
    SKIPABORT         ;skip abort for servers if fatal ccl error encountered
    SKIPAST           ;on to disable any ast timers
    SKIPDCL           ;skip dcl calls
    SKIPPERSIST       ;treat persist as persistscript instead of global record
    SKIPPERSISTSCRIPT ;ignore any persist or persistscript
    SKIPRECACHE       ;skip recache check
    SKIPRECONNECT     ;skip the reconnect logic and treat as fatal error
    SKIPUAR           ;disable uar call
    SKIPVERSION       ;skip file versioning
    SRVTRAILVAR       ;modify dbrtl to retain trailing spaces for varchar record fields
    SRVUINT           ;enable cclsrv unsigned int csa reply types
    SYMBOLRESET       ;if on then wont do cclset_symbol mark/release
    TEST              ;unused
    TREE              ;trace tree
    TRUNCATECHECK     ;show warning for char variable truncation occurs
    UTCDEBUG          ;utc debug level to enable
    VARLIST           ;on: debug varlist
    VIEWCL            ;trace views
    WARNING           ;show warnings for debugging level 1
    WARNING2          ;show warnings for debugging level 2
    WARNING3          ;show warnings for debugging level 3
================================================================================================
>>>6)Modules
  Module identifiers consist of the following and is used to tag the ccl module that raised the error:
  %CCL-<severity>-<errnum>-<program_name>(<row>,<col>)S<server>L<nest_lev>.<prg_lev><module>{<symbol>}<message>
  %CCL-E-67-JCM1(0,0)S0L1.1q1{VAR1}Invalid select variable (VAR1) encountered.
  CCLAMIGEN=a1
  CCLCACHE=c1 CCLCALL=c2 CCLCALLUAR=c3 CCLCREATECPP=c4 CCLCREATEDEF=c5
  CCLDEBUGGER=d1 CCLDICTIONARY=d2 CCLDIO=d3 CCLDISPLAY=d4 CCLDROPDEF=d5 CCLDSRI=d6
  CCLEDIT=e1 CCLEKS=e2
  CCLFORMAT=f1
  CCLGENERATE=g1
  CCLLEXSCAN=l1
  CCLMAIN=m1 CCLMEMPOOL=m2 CCLMEMSORT=m3 CCLMESSAGE=m4 CCLMISC=m5 CCLMODIFY=m6
  CCLPAINT=p1 CCLPARSER=p2 CCLPLAN=p3 CCLPRCTREE=p4 CCLPROGRAM=p5
  CCLQUAL=q1 CCLQUERY=q2
  CCLRANGE=r1 CCLRECSTRUCT=r2 CCLREPORT=r3
```

```
CCLSECURITY=s1 CCLSORT=s2 CCLSYS=s3 CCLSUB=s4
CCLTAM=t1 CCLTREE=t2 CCLXML=x1
```