```
#!/usr/bin/ksh
#
# ARL-DSRC Sdata wrapper
# James C. Ianni 2011
# james.ianni@us.army.mil
#
# RCS : $Date: 2012/07/02 14:54:52 $ $Revision: 1.26 $
#


function cleanup {
       rm  ${temp1} >/dev/null 2>&1
       rm  ${temp2} >/dev/null 2>&1
        exit
}

function myexit {
   cleanup
#   rm -f $LFILE
           exit ${1}
}
function line2 {
            echo "
**********************************************************************  "
}

function line1 {
            echo " "
                echo "
**********************************************************************  "
}

function echo2 {
             printf '%6s %-64s %6s\n' '   ** ' "${1}" ' ***  '
}

function header {
          line2
          echo2 "${1}"
          line2
}

function getout {
           line1 ; echo2 "SEVERE ERROR HAPPENED!!" ; echo2 "${1}" ; line2
#   cleanup
#   rm -f ${LFILE}
              exit 1
}

function cerr {
if [ $? -ne 0 ]; then
     echo "Problem/error happened!!!"
     echo ${1}
    return 1
fi
   return
}
```

```
 56
 57 function qerr {
 58 if [ $? -ne 0 ]; then
 59     echo "Problem/error happened!!!"
 60     echo ${1}
 61     myexit 1
 62 fi
 63   return
 64 }
 65
 66 function pause {
 67   echo2 " SCRIPT PAUSED."
 68   read bleepo
 69   return
 70 }
 71
 72 function cdate {
 73 a=$(date +%Y%m%d%H)
 74 return
 75 }
 76
 77 # print compress date-time to expanded
 78 function prdate {
 79 typeset hours  datetime hours0  stringa  stringb stringc  stringd
 80 datetime=${1}
 81 hours0=$(echo ${datetime} | cut -c9-10 )
 82 hours="${hours0} hundred hours"
 83 stringa=$(echo ${datetime}| cut -c5-6 )
 84 stringb=$(echo ${datetime}| cut -c7-8)
 85 stringc=$(echo ${datetime}| cut -c1-4 )
 86 string="${stringa}/${stringb}/${stringc} at ${hours}"
 87 return
 88 }
 89
 90 function myerror {
 91
 92   case "${1}" in
 93      1)
 94         echo "User Error: Missing argument to option \"${2}\" !"
 95         exit 1
 96         ;;
 97      2)
 98        echo "Unknown option \"${2}\" given on command line!"
 99        exit 1
100        ;;
101      *) echo "Software error at $LINE , $LINENO"
102        ;;
103   esac
104 }
105
106 function displayall {
107 typeset lines tlines
108
109   if [ ${schemeset} -eq 1 ]; then
110     if [ "${scheme}" == "Name_Value" ]; then
111       Sscheme ${rflag} -l -scheme Name_value ${1} |  grep -v '\ -row ' >
    ${temp2}
```

```
112       lines=`wc -l ${temp2}| cut -f1 -d" "`
113       (( lines = lines - 5 ))
114       if [ "${preflag}" -eq 1 ]; then
115           filepref2=$(echo ${1} | sed 's/\//\\\//g')
116          tail -${lines} ${temp2} | cut -c120- | sed 's/^$/xs1a/g' | tr '\012' '
    ' | sed 's/xs1a/\n/g' | sed 's/\"  *\"/\=/g' | sed 's/^  *//g'  |  tr -s '\012'
    | sed "s/^/${filepref2}:/g"
117       else
118          tail -${lines} ${temp2} | cut -c120- | sed 's/^$/xs1a/g' | tr '\012' '
    ' | sed 's/xs1a/\n/g' | sed 's/\"  *\"/\=/g' |  sed 's/^  *//g'  |  tr -s '\012'
119       fi
120       echo " "
121     elif [ "${scheme}" == "Dublin_Core" ]; then
122     # Now the dublin core
123       Sscheme ${rflag} -l -scheme 'Dublin_core' ${1} | grep -v '\ -row ' >
    ${temp2}
124       lines=`wc -l ${temp2}| cut -f1 -d" "`
125       (( lines = lines - 5 ))
126       (( tlines = lines - 2 ))
127       if [ "${preflag}" -eq 1 ]; then
128       #  echo -e "${filepref}:\c";
129         filepref2=$(echo ${1} | sed 's/\//\\\//g')
130         tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/
    *\"/=\"/g' | grep -v '""' |  sed "s/^/${filepref2}:/g"
131       else
132
133         tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/
    *\"/=\"/g' | grep -v '""'
134       fi
135     #  echo " "
136     fi
137     return
138   fi
139
140 #
141 # Display Everything because scheme was not set on command line
142 #
143 Sscheme ${rflag} -l -scheme Name_value ${1}  | grep -v '\ -row ' > ${temp2}
144 lines=`wc -l ${temp2}| cut -f1 -d" "`
145 (( lines = lines - 5 ))
146 if [ "${lines}" -ge "0" ]; then
147   if [ "${preflag}" -eq 1 ]; then
148     filepref2=$(echo ${1} | sed 's/\//\\\//g')
149       tail -${lines} ${temp2} | cut -c120- | sed 's/^$/xs1a/g' | tr '\012' ' '
    | sed 's/xs1a/\n/g' | sed 's/\"  *\"/\=/g' |  sed 's/^  *//g'  |  tr -s '\012'
    | sed "s/^/${filepref2}:/g"
150   else
151      tail -${lines} ${temp2} | cut -c120- | sed 's/^$/xs1a/g' | tr '\012' ' ' |
    sed 's/xs1a/\n/g' | sed 's/\"  *\"/\=/g' |  sed 's/^  *//g'  |  tr -s '\012'
152   fi
153 fi
154 # Now the dublin core
155
156 Sscheme ${rflag} -l -scheme 'Dublin_core' ${1} |  grep -v '\ -row ' > ${temp2}
157 lines=`wc -l ${temp2}| cut -f1 -d" "`
158 (( lines = lines - 5 ))
159 (( tlines = lines - 2 ))
```

```
160
161 if [ "${lines}" -ge "0" ]; then
162       if [ "${preflag}" -eq 1 ]; then
163         filepref2=$(echo ${1} | sed 's/\//\\\//g')
164         tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/
    *\"/=\"/g' | grep -v '""' | sed "s/^/${filepref2}:/g"
165       else
166        tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/
    *\"/=\"/g' | grep -v '""'
167       fi
168   #echo " "
169 fi
170 }
171
172 function preparekey {
173 typeset i  j   k  f  p1  p2
174 i=0; j=0; k=0; f=0; p1=0; p2=0
175 #       butill-0002.arl.hpc.mil>     Sscheme -l -scheme Comment_Scheme abutil*
176 #
177 #Sscheme  -w -val
    'Admin.Retention_Period::90,Admin.Last_Review_Time::2011-04-25-13.48.05,Admin.HP
    CMP_Project_ID::HPCMO92330SIS' test.txt
178 # You can see the Schemes with SgetS and the -x option lists all column names.
179        if [ "${1}" == "" ]; then
180                cerr "missing operand in prepare object!"
181        fi
182       if [ ${search} -eq 0 ]; then
183          echo ${*} |  sed 's/\;/\n/g' >> ${temp1}
184       else
185         echo ${*} | sed 's/\;/ AND\n/g' | sed 's/xs1b/ OR\n/g' >> ${temp1}
186       fi
187 }
188
189 function findrow {
190
191 #echo "0,1,2=$0,$1,$2"
192   export row=$(echo "($(Sscheme -l -scheme Name_Value ${1}  |  grep -v '\ -row
    ' | grep -n \"${2}\" | cut -f1 -d:)-6)/3" | bc)
193   if [ "${row}" == "" ]; then
194      echo "Soft error: row is null in findrow{} !"
195   fi
196   return
197 }
198
199 function askuser {
200   typeset ent
201   echo "
202 ${1}
203 "
204 echo "Are you sure you want to do the above action?"
205 echo "Press Y or y, any other key means No"
206 read ent
207 if [ "${ent}" == "" ]; then
208   ent=n
209 fi
210 if [ "${ent}" == "Y" -o "${ent}" == "y" ]; then
211    export goyes=1
```

```
212 else
213    export goyes=0
214 fi
215 echo " "
216 return
217 }
218
219 function maxrow {
220 #  echo "MAX1"
221   export row=$(echo "(`Sscheme -l -scheme Name_Value ${1} | grep -v '\ -row ' |
    wc -l`-6)/3" | bc)
222 #  echo "MAX2"
223   return
224 }
225
226 #
227 # Search feature of Sdata
228 #
229
230 function searchit {
231 typeset i j k f p1 p2 srch addme
232 typeset -i lg lg2 setor setnot latch contand contand2
233   lg=0
234   lg2=0
235   setor=0
236   latch=0
237   setnt=0
238   contand=0
239   contand2=0
240   addme=""
241   notl=""
242   tn=""
243   sq=$(echo -e '\047')
244   srch="-policy \""
245
246   if [ "${1}" == "" ]; then
247     getout "missing operand in searchit!"
248   fi
249   which SgetD >/dev/null 2>&1
250   if [ "$?" -ne "0" ]; then
251     getout "*ERROR* Cannot locate SgetD command. Is the SRB module loaded?"
252     echo "*ERROR $LINENO"
253     myexit 1
254   fi
255   while read line ; do
256           if [ ${contand} -ge 1 ]; then
257                (( contand2=contand2+1 ))
258           fi
259
260 #     echo "line=\"$line\""
261
262        if [ "${latch}" -eq 1 -a  ${contand2} -lt 1 ]; then
263          srch="${srch} ${addme}"
264        fi
265      if [ "${line:0:6}" == "SCHEME!" ]; then
266           export scheme=${line:7}
267           test ${verbose} -eq 1 && echo "New scheme detected and set to
```

```
    \"${scheme}\" ..........."
268           continue
269       fi
270 # test for AND OR operator at end of line
271       addme=" OR "
272       lg=${#line}
273       (( lg2=lg-4 ))
274       k=${line:${lg2}:4}
275       k=$( echo ${line}|awk '{print $NF}')
276       case "${k}" in
277        "AND")
278 #echo "IN CASE AND"
279           (( contand=contand+1 ))
280           latch=1
281           addme=" AND "
282           (( lg2=lg-4 ))
283           line=${line:0:${lg2}}
284         ;;
285        "OR")
286           latch=1
287           addme=" OR "
288           (( lg2=lg-3 ))
289           line=${line:0:${lg2}}
290 # JCI new 6-19-2012
291            if [ ${contand} -ge 1 ]; then
292               while [ ${contand} -ge 1 ] ; do
293                 srch="${srch})"
294                 (( contand=contand-1 ))
295               done
296             fi
297         ;;
298         *)
299         #   latch=1
300         #   addme=" OR "
301         #   (( lg2=lg2-1 ))
302         #   line=${line:0:${lg2}}
303 # JCI 6-22-2012
304           latch=1
305           addme=" OR "
306 #          (( lg2=lg-3 ))
307 #          line=${line:0:${lg2}}
308 ## JCI new 6-19-2012
309 #            if [ ${contand} -ge 1 ]; then
310 #               while [ ${contand} -ge 1 ] ; do
311 #                 srch="${srch})"
312 #                 (( contand=contand-1 ))
313 #               done
314 #            fi
315
316         ;;
317       esac
318 ##
319       echo ${line} | grep '=' 1>/dev/null 2>&1
320 # new JCI 6-19-2012
321           if [ ${contand2} -ge 1 ]; then
322 #              (( contand2=contand2-1 ))
323             srch="${srch} AND DATA_OBJECT.data_id IN (select
```

```
    DATA_OBJECT.data_id where"
324           fi
325      if [ $? -eq 0 ]; then
326     #    echo "2nd field present, this is a search=this thingy or search=NULL"
327             p1=$(echo ${line} | cut -f1 -d=)
328             p2=$(echo ${line} | cut -f2 -d=)
329      #          echo "p1,p2=${p1},${p2}"
330             stype=0
331             tn=${p1:0:1}
332             if [ "${tn}" == "!" ]; then
333               notl="not "
334               tn=${p1:1}
335               p1=${tn}
336             else
337               notl=""
338             fi
339
340             echo ${p1} | egrep -i
    "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation
    Date$|^Type$|^Document ID$|^Rights$" >/dev/null 2>&1
341             if [ $? -eq 0 ]; then
342               test ${verbose} -eq 1 &&  echo "Dublin Core field detected,
    \"${p1}\", temporarily switching to Dublin_core scheme..."
343               stype=1
344               oldscheme=${scheme}
345               scheme=Dublin_Core
346             fi
347             if [ "${p2}" == "" ]; then
348             #  was delete but now keyword= should be interpreted as having the
    field contain null ******************************
349             # SgetD -policy "(Name_Value.Name like color) AND (Name_value.value
    like red)"
350               test ${verbose} -eq 1 && echo "Adding in search \"${p1}\" from
    \"${scheme}\" scheme"
351               if [ ${stype} -eq 0 ]; then
352               #
353               # was  Sscheme ${rflag} -scheme Name_Value -d -row ${row} ${1}
    >/dev/null
354               # SgetD -policy "(Name_Value.Name like color) AND
    (Name_value.value like red)"
355                 srch="${srch}(Name_Value.Name not like ${p1})"
356                 stype=0
357               else
358                 # Sscheme ${rflag} -w -val "Dublin_Core.${p1}::" ${1}
    >/dev/null
359                 srch="${srch}(not like Dublin_core.${p1})"
360               fi
361             else
362          # this is the search for this thingy in other words "field=this"
    search*****************************************
363               test ${verbose} -eq 1 && echo "Changing/Inserting search
    ${notl}\"${p2}\" into \"${p1}\" field for \"${scheme}\" scheme for object
    \"${1}\""
364               if [ ${stype} -eq 0 ]; then
365                #   Sscheme ${rflag} -w -val
    "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}"
    >/dev/null
```

```
366                 srch="${srch}(Name_Value.Name like ${p1}) AND
    (Name_value.value ${notl}like ${sq}${p2}${sq})"
367               else
368                #   Sscheme ${rflag} -w -val "Dublin_Core.${p1}::${p2}" "${1}"
      > /dev/null
369                # SgetD -policy "(Dublin_Core.Title like Story*)"
370                  # srch="${srch}(Dublin_Core.${p1} ${notl}like
    ${sq}${p2}${sq}*) "
371                 srch="${srch}(Dublin_Core.${p1} ${notl}like ${sq}${p2}${sq}) "
372               fi
373             fi
374     else  # if-then for equal sign present inside line or not
    ================================================================================
    ===============
375             p1=${line}
376     #          echo "p1=${p1}"
377             # was display
    ********************************************************************************
    ******************
378             # now for just showing if scheme has a field
    set*************************************************************
379           stype=0
380           echo ${p1} | egrep -i
    "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation
    Date$|^Type$|^Document ID$|^Rights$" >/dev/null
381           if [ $? -eq 0 ]; then
382              test ${verbose} -eq 1 &&  echo "Dublin Core field detected,
    \"${p1}\", temporarily switching to Dublin_core scheme..."
383               stype=1
384               oldscheme=${scheme}
385               scheme=Dublin_Core
386             fi
387             test ${verbose} -eq 1 && echo "Search only for field is
    ${notl}present  \"${p1}\" field for \"${scheme}\" scheme "
388             if [ ${stype} -eq 0 ]; then
389              #   Sscheme ${rflag} -w -val
    "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}"
    >/dev/null
390                # srch="${srch}(Name_Value.Name like ${p1}) AND (Name_value.value
    like ${p2})"
391                 srch="${srch}(Name_Value.Name ${notl}like ${p1})"
392             else
393                # srch="${srch} (Name_Value.Name like ${p1}) AND
    (Name_value.value like ${p2})"
394                 srch="${srch}(Dublin_core ${notl}like ${p1})"
395             fi
396             stype=0
397 #
398 # put in search string for searching a set field only here!
399 #
400
401 #
402 #
403     fi  # if-then for equal sign present inside line or not
    ================================================================================
    ================
404 # JCI new 6-19-2012
```

```
405             if [ ${contand2} -ge 1 ]; then
406               (( contand2=contand2-1 ))
407                  srch="${srch})"
408             fi
409   done  < ${temp1}
410 #
411   srch="${srch}\""
412
413 ##########################################################################
414 #
415 # Now pump search string into SgetD
416 #
417 test ${verbose} -eq 1 &&  echo "The search string is now set at ${rflag}
    ${srch} ${scollect}"
418 if [ "${explicitobjset}" -eq 0 ]; then
419  :
420 #  eval SgetD ${rflag} ${srch}
421    eval Sls ${rflag} ${srch}
422 else
423  :
424 #  eval SgetD ${rflag} ${srch} ${scollect}
425    eval Sls ${rflag} ${srch} ${scollect}
426 fi
427 myexit 0
428
429 #
430 #
431 ##########################################################################
432
433   return 0
434 }
435 function changeobject {
436         typeset i j k f p1 p2
437 #Sscheme  -w -val
    'Admin.Retention_Period::90,Admin.Last_Review_Time::2011-04-25-13.48.05,Admin.HP
    CMP_Project_ID::HPCMO92330SIS' test.txt
438 # You can see the Schemes with SgetS and the -x option lists all column names.
439         if [ "${1}" == "" ]; then
440                 cerr "missing operand in changeobject!"
441         fi
442         exec 5<${temp1}
443         which Sscheme >/dev/null 2>&1
444         if [ $? -ne 0 ]; then
445                 getout "*ERROR* Cannot locate Sscheme command. Is the SRB
    module loaded?"
446         fi
447 #       echo ${2} |  sed 's/\;/\n/g' > ${temp1}
448 ##      for i in $(cat ${temp1}); do
449          while read -u5 line ; do
450               #echo "line=$line"
451               if [ "${line:0:6}" == "SCHEME!" ]; then
452                   export scheme=${line:7}
453                   test ${verbose} -eq 1 && echo "New scheme detected and set to
    \"${scheme}\" ..........."
454                   continue
455               fi
456               echo ${line} | grep '=' 1>/dev/null 2>&1
```

```
457                 if [ $? -eq 0 ]; then
458                 #    echo "2nd field present"
459                     p1=$(echo ${line} | cut -f1 -d=)
460                     p2=$(echo ${line} | cut -f2 -d=)
461 #                   echo "p1,p2=${p1},${p2}"
462                     stype=0
463                     echo ${p1} | egrep -i
    "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation
    Date$|^Type$|^Document ID$|^Rights$" >/dev/null 2>&1
464                     if [ $? -eq 0 ]; then
465                       test ${verbose} -eq 1 &&  echo "Dublin Core field detected,
    \"${p1}\", temporarily switching to Dublin_core scheme..."
466                       stype=1
467                       oldscheme=${scheme}
468                       scheme=Dublin_Core
469                     else
470 # if not dublin, then name_value, there could be more in the future so this may
    need to be changed
471                       scheme=name_value
472                     fi
473                     if [ "${p2}" == "" ]; then
474                     # delete
    ************************************************************************************
    **********************
475                       if [ "${enforced}" -eq 0 ]; then
476                          echo "The \"-d\" switch was NOT specified on the
    command line. Ignoring request to delete ${p1} from ${1} !"
477                          continue
478                       fi
479                       if [ "${ask}" -eq "1" ]; then
480                           askuser "Delete \"${p1}\" from \"${scheme}\" scheme
    for object \"${1}\""
481                           if [ "${goyes}" -eq "0" ]; then
482                             continue
483                           fi
484                       fi
485                     test ${verbose} -eq 1 && echo "Deleting \"${p1}\" from
    \"${scheme}\" scheme for object \"${1}\""
486                     if [ ${stype} -eq 0 ]; then
487                     #
488                       findrow "${1}" "${p1}"
489                       if [ ${row} -lt 0 ]; then
490                           echo "*ERROR* There is no field named ${p1} for object
    ${1}! Ignoring delete..."
491                           continue
492                       fi
493                         [ ${DEBUG2} -eq 1 ] && echo "The delete would look like
    this ---==>>> Sscheme -scheme Name_Value -d -row ${row} ${1}"
494                        Sscheme ${rflag} -scheme Name_Value -d -row ${row} ${1}
    >/dev/null
495                     else
496                     #  Sscheme ${rflag} -w -val "Dublin_Core.${p1}::${p2} ''"
    ${1}   >/dev/null
497                       [ ${DEBUG2} -eq 1 ] && echo  Sscheme ${rflag} -w -val
    "Dublin_Core.${p1}::" ${1}
498                       Sscheme ${rflag} -w -val "Dublin_Core.${p1}::" ${1}
    >/dev/null
```

```
499                     fi
500                   else
501                 # change/insert
    ********************************************************************************
    ***************
502 ## Kludge for removing double-double quotes
503 #                     p4=$(echo ${p2} | sed 's/\"\"/\"/g')
504 #                     p2=${p4}
505                       if [ "${ask}" -eq "1" ]; then
506                          askuser "Changing/inserting \"${p2}\" into \"${p1}\"
    field for \"${scheme}\" scheme for object \"${1}\""
507                        if [ "${goyes}" -eq "0" ]; then
508                          continue
509                        fi
510                       fi
511                     test ${verbose} -eq 1 && echo "Changing/Inserting \"${p2}\"
    into \"${p1}\" field for \"${scheme}\" scheme for object \"${1}\""
512                     if [ ${stype} -eq 0 ]; then
513                   # Sscheme -w -val
    'Name_Value.Name[0]::Mw0,Name_Value.Value[0]::zeroth' abutil.txt
514                       findrow "${1}" "${p1}"
515  #                    echo "row=${row}"
516                       if [ ${row} -lt 0 ]; then
517                       # Name not there, so place at end
518                         maxrow "${1}" "${p1}"
519                         if [ ${row} -gt 20 ]; then
520                            echo "There is no more room in Name-Value table to
    place ${p1} for object \"${1}\"! Ignoring insert request!"
521                            continue
522                         fi
523                       fi
524                       [ ${DEBUG2} -eq 1 ] && echo Sscheme ${rflag} -w -val
    "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}"
525                       Sscheme ${rflag} -w -val
    "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}"
    >/dev/null
526                     else
527                   # Sscheme -w -val 'Dublin_Core.title::Bizarre Rituals of the
    West Phillians' abutil.txt
528                        [ ${DEBUG2} -eq 1 ] && echo Sscheme ${rflag} -w -val
    "Dublin_Core.${p1}::${p2}" "${1}"
529                       Sscheme ${rflag} -w -val "Dublin_Core.${p1}::${p2}"
    "${1}"     > /dev/null
530                     fi
531                   fi
532                   continue
533               else
534                   p1=${line}
535 #                 echo "p1=${p1}"
536                # display
    ********************************************************************************
    ************************
537                 stype=0
538               # echo "preflag=$preflag"
539                echo ${p1} | egrep -i
    "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation
    Date$|^Type$|^Document ID$|^Rights$" >/dev/null
```

```
540                 if [ $? -eq 0 ]; then
541                    test ${verbose} -eq 1 &&  echo "Dublin Core field detected,
    \"${p1}\", temporarily switching to Dublin_core scheme..."
542                     stype=1
543                     oldscheme=${scheme}
544                     scheme=Dublin_Core
545                  fi
546                 test ${verbose} -eq 1 && echo "Display item inside \"${p1}\"
    field for \"${scheme}\" scheme in object \"${1}\""
547               fi
548               if [ "${stype}" -eq "1" ]; then
549               # display Dublin field
550                if [ "${preflag}" -eq 1 ]; then
551                   filepref2=$(echo ${1} | sed 's/\//\\\//g')
552                   [ ${DEBUG2} -eq 1 ] && echo "Sscheme -l -scheme Dublin_Core
    ${1} "
553                    Sscheme -l -scheme Dublin_Core ${1}  |  grep -v '\ -row ' |
    grep ${p1} | egrep -o '\".*\"$' | sed "s/^/${filepref2}:/g"
554                   #Sscheme -l -scheme Dublin_Core ${filepref2} | grep ${p1} |
    egrep -o '\".*\"$' | sed "s/^/${filepref2}:/g"
555                 else
556                   [ ${DEBUG2} -eq 1 ] && echo "Sscheme -l -scheme Dublin_Core
    ${1}"
557                    Sscheme -l -scheme Dublin_Core ${1}  |  grep -v '\ -row ' |
    grep ${p1} | egrep -o '\".*\"$'
558                 fi
559               # turn off dublin
560                  stype=0
561                  scheme=${oldscheme}
562               else
563               # display field inside user scheme or other scheme
564               # Sscheme -l -scheme name_value abutil.txt | grep -A1
    'string\[16\]' |  grep test
565                 [ ${DEBUG2} -eq 1 ] && echo  "Sscheme -l -scheme name_value
    ${1}"
566                  Sscheme -l -scheme name_value ${1} | grep -v '\ -row ' | grep
    'string\[16\]' |  grep ${p1} >/dev/null 2>&1
567                  if [ $? -eq 0 ]; then
568                 # match!
569                   if [ "${preflag}" -eq 1 ]; then
570                     filepref2=$(echo ${1} | sed 's/\//\\\//g')
571                     [ ${DEBUG2} -eq 1 ] && echo  "Sscheme -l -scheme name_value
    ${1}"
572                      Sscheme -l -scheme name_value ${1} | grep -v '\ -row ' |
    grep -A1 'string\[16\]' | grep -A1 ${p1} | tail -1 | egrep -o '\".*\"' | sed
    "s/^/${filepref2}:/g"
573                    # echo "YULP"
574                    # read n
575                   else
576                     [ ${DEBUG2} -eq 1 ] && echo  "Sscheme -l -scheme name_value
    ${1}"
577                      Sscheme -l -scheme name_value ${1} | grep -v '\ -row ' |
    grep -A1 'string\[16\]' | grep -A1 ${p1} | tail -1 | egrep -o '\".*\"'
578                   fi
579                 else
580                 # no match
581                   test ${verbose} -eq 1 && echo "Field \"${p1}\" was NOT FOUND
```

```
    for object \"${1}\" !"
582                 fi
583             fi
584         done
585 #         done  < ${temp1}
586 #         done  <&5
587         return
588 }
589
590 function displayhelp {
591
592       echo '
593
594       Sdata - Set/modify/delete/show/search metadata on objects within a SLM
    collection
595
596       SYNOPSIS
597
598       Sdata {-R} {-o|--object} object_name {[-p|--project]PROJECT} {-c
    <collection>} {keyword{= {value}} { keyword{={value}} ...} {OPTIONS}
599
600       DESCRIPTION
601       Sdata allows one to display, set, change or delete keyword-value pairs or
    the project in the Storage Resource Broker (SRB) metadata.
602       When the "-S" or "--search" option is provide, Sdata will search for
    files containing metadata (see Sdata In Search Mode below)
603
604       OPTIONS
605       -c, --collection  set the collection to operate
606       -d, --delete Enforce deletions for "keyword=" keywords
607       -f, --force  ignore nonexistent files, never prompt
608       -h,--help  display this help and exit
609       -i, --interactive  prompt before setting metadata
610       -o, --object  SRB object or objects
611       -p, --project  set the project code for object
612       -R, --recursive  operate on SRB object metadata contents recursively
613       -s,--scheme choose scheme to display/select
614       -S,--search run Sdata in search mode to find files that match a metadata
    line arguments
615       -v, --verbose  explain what is being done
616       --version  output version information and exit
617       -x, --xml insert/parse xml
618
619       keyword{={value}}
620
621       For each object_name, the {keyword{={value}}|..} will perform actions:
622
623       "keyword" is NOT provided,  all metadata associated with object_name is
    displayed to standard output
624       "keyword" is provided,  metadata associated with keyword is displayed to
    standard output  "keyword=" is provided,  metadata associated with keyword is
    DELETED
625       "keyword=value" is provided,  metadata associated with "keyword" is
    inserted/changed to "value"
626
627       For the Title, Creator, Subject, Description, Publisher, Contributor,
    Creation Date, Type, Document ID, and Rights names metadata values will be
```

```
    stored in the "Dublin Core" scheme. The values for all other names will be
    stored in the "Name Value" scheme. At the current time the "Name Value" scheme
    is limited to 20 name value pairs.
628
629       EXAMPLES :
630
631       Sdata -o MyObj "Creator=John Doe"  This command will set the Creator
    attribute for the object MyObj in the current collection to "John Doe".
632
633       Sdata -o MyObj  This command will display all metadata to standard output
    for the object MyObj in the current collection.
634
635       Sdata -d -o MyObj "Creator="  This command will DELETE the Creator
    attribute value for the object MyObj in the current collection to "John Doe".
636
637        Sdata -o MyObj "Creator=John Doe" "Description=A model of some type"
    Type=Input   This command will set the Creator attribute to "John Doe", the
    Description attribute to "A model of some type", and the Type attribute to
    "Input" for the object MyObj in the current collection.
638
639        Sdata -R -c user/ModelA/Input "Description=A model of some type"
    Type=Input   This command will set the Description attribute to "A model of
    some type", and the Type attribute to "Input" for all objects recursively in
    the user/ModelA/Input.
640
641        eval Sdata -o MyObj `Sdata -o MyObj_2` (watch the backticks!)  This
    command will copy  all the user metadata values of MyObj_2 to MyObj.
642
643 '
644 echo '
645
646        Sdata In Search Mode:
647        ========================================================================
    ==
648        (NB: All other command line flags MUST come before the "-S" flag! )
649
650        Sdata -o Model_1 -S color=red  This will locate all files inside the
    Model_1 collection that contain metadata which the color is set to red
651
652        Sdata -o Model_1 -S color=red OR size=large  This will locate all files
    inside the Model_1 collection that contain metadata that has color set to red
    OR size is set to large
653
654        Sdata -o Model_1 -S color=red AND size=large  This will locate all files
    inside the Model_1 collection that contain metadata that has color set to red
    AND size is set to large
655
656        Sdata -o Model_1 -S color=red AND \!size=large  This will locate all
    files inside the Model_1 collection that contain metadata that has color set to
    red AND size is NOT set to large
657
658         Sdata -o Model_1 -S \!color=red  This will locate all files inside the
    Model_1 collection that contain metadata which the color is NOT set to red
659
660
661        '
662        myexit 0
```

```
663 }
664
665 function xtest {
666  typeset f
667    echo
    "--------------------==================================vvvvvvvvvvvvvvvvvvvvvv
    vvvvvvvv"
668    if [ "${2}" != "" ]; then
669      echo " ******************** Test Type: ${2} "
670    fi
671    echo "Testing \"${1}\"  ......"
672    echo " "
673 #   eval "${1} --verbose"
674    eval "${1}"
675 #   if [ $? -ne 0 ]; then
676 #      echo "----====>> PROBLEM WITH \"${1}\" !!!"
677 #      return 1
678 #   fi
679    echo "-------------==========================^^^^^^^^^^^^^^^^^^^^^^^^"
680    echo " "
681    }
682
683 function gotest {
684  typeset f
685 echo "
686
687     ****************************************
688    ********** Running Internal Tests ***********
689     ****************************************
690
691    "
692
693    f=abutil.txt.$$
694    echo "This is a test of the Sdata command" > ${WORKDIR}/${f}
695    echo "You are currently in SRB path:"
696    Spwd
697    if [ $? -ne 0 ]; then
698       echo "** SEVERE ERROR! Cannot \"Spwd\""
699       exit 1
700     fi
701     echo " "
702    Sput ${WORKDIR}/${f} .
703    if [ $? -ne 0 ]; then
704       echo "** SEVERE ERROR! Cannot \"Sput ${f} .\""
705       exit 1
706     fi
707 xtest "Sdata ${f} theory=MP2 subject='Physical Organic Chemistry'" "Inserting
    fields"
708 xtest "Sdata ${f}"  "Display All Special Metadata Fields"
709 xtest "Sdata ${f} --scheme Name_Value" "Display Only Metadata Associated with
    Name_Value Scheme"
710 xtest "Sdata ${f} theory=B3LYP AUTHOR='Willard Gibbs' DECRIPTION='H2SO4-H2O
    Phase diagrams' PROGRAM='Gaussian09 RevB' MW=95.43333 ISOMER='Lowest energy'  "
     "Modify previous records and insert new records"
711 xtest "Sdata ${f} " "Examine the output to verify previous command has run
    correctly."
712 xtest "Sdata ${f} PROGRAM AUTHOR theory"  "Display AUTHOR and Theory fields"
```

```
713 xtest "Sdata ${f} ARTHUR" "Cannot display field which does not exist"
714 xtest "Sdata ${f} -d  MW=" "Delete Mw field"
715 xtest "Sdata ${f}"  "Examine the output to verify previous command has run
    correctly."
716 xtest "Sdata ${f} -d AUTHOR MW=393.4343 EXCELFILENAME='H2SO4-H2O_3.xls' theory
    AUTHOR= ISOMER="  "Modify/Insert/Delete/Display various fields"
717 xtest "Sdata ${f}"  "Examine the output to verify previous command has run
    correctly."
718
719 echo "
720
721 Testing is Finished.
722
723 "
724 }
725
726 #
727 # Settings
728 #
729
730 export DEBUG2=0
731 # Cores per node
732 CORESPERNODE=${BC_CORES_PER_NODE:-8}
733 # memory available to user in GB
734 MEMPERNODE=${BC_MEM_PER_NODE:-17}
735 export ask=0
736 object=
737 export project=
738 export projectset=0
739 collection=
740 first=0
741 export verbose=0
742 export row=0
743 export enforced=0
744  export schemeset=0
745 #
746
747 #
748 # Start of script
749 #
750
751 if [ -z "${USER}" ]; then
752    getout "USER env variable not set!!!"
753 fi
754
755 if [ -z "${WORKDIR}" ]; then
756 #   getout "WORKDIR env variable not set!!!"
757    export WORKDIR=/usr/var/tmp/${USER}
758 fi
759
760 #out=${WORKDIR}/${USER}
761 out=${WORKDIR}
762 temp1=${out}/sdata.$$.temp1.out
763 cerr "Cannot create temp1"
764 temp2=${out}/sdata.$$.temp2.out
765 cerr "Cannot create temp2"
766 export recur=0
```

```
767 export verbose=0
768 export scheme=Name_Value
769 export rflag=""
770 export project=xxxx
771 export projectset=0
772 export search=0
773 export objectset=0
774 export explicitobjset=0
775 export filepref=""
776 export preflag=0
777
778
779 if [ ! -d ${out} ];  then
780    mkdir -p ${out}
781    cerr "Cannot mkdir -p ${out}"
782 fi
783
784 if [ -z  "${SAMPLES_HOME}" ]; then
785 #   getout "SAMPLES_HOME directory is not set!!!"
786    export SAMPLES_HOME=/usr/cta/SCR
787 fi
788
789 project=$(cat /etc/passwd | egrep -e "^${USER}:" | cut -f2 -d\( | cut -f1 -d\))
790 #echo "Project = <${project}> "
791 #if [ -z ${project} ]; then
792 #       getout "Could not obtain users project id from passwd!!"
793 #fi
794
795 while [ "$#" -ge 1 ]; do
796   case "${1}" in
797       -d|--delete|--DELETE|-delete)
798           export enforced=1
799            shift
800        ;;
801       -h|--help|-H|--HELP|--Help|-help|-HELP|-Help)
802           displayhelp
803           shift
804            exit 0
805        ;;
806        -t|--test)
807            #echo "Test 1, value=${2}"
808            shift
809            gotest
810            exit 0
811            ;;
812        --debug2)
813            export DEBUG2=1
814            shift
815            ;;
816         -o|--obj*|--Obj*)
817            if [ "${2}" == "" ]; then
818                    myerror 1 ${1}
819            fi
820            object=${2}
821            export objectset=1
822            export explicitobjset=1
823            first=1
```

```
824             shift ; shift
825             ;;
826         -x|--xml|--XML)
827             if [ "${2}" == "" ]; then
828                     myerror 1 ${1}
829             fi
830             myxml=$(2)
831             echo "XML to parse: ${2}"
832             shift ; shift
833             ;;
834          -p|--proj*|--PROJ*|--Proj*|-proj*)
835             if [ "${2}" == "" ]; then
836                     myerror 1 ${1}
837             fi
838             export project=${2}
839             export projectset=1
840             shift ; shift
841            ;;
842          -c|--collect*|--Collect*|-collect*)
843             if [ "${2}" == "" ]; then
844                     myerror 1 ${1}
845             fi
846             collection=${2}
847             object=${2}
848             export objectset=1
849             export explicitobjset=1
850             first=1
851             shift ; shift
852            ;;
853          -s|--scheme|--Scheme|--SCHEME|-scheme)
854             if [ "${2}" == "" ]; then
855                     myerror 1 ${1}
856             fi
857 #           if [ "${2}" == "Name_Value" -o "${2}" == "name_value" -o "${2}" -o
    "${2}" == "Name_value"
858             export scheme=${2}
859             export schemeset=1
860             echo "SCHEME!${2}" >> ${temp1}
861             shift ; shift
862            ;;
863            -v|--verbose|--Verbose)
864                export verbose=1
865                shift
866            ;;
867            --version|--Version|-version)
868              echo "
869
870
871              Sdata Beta RCS ID: $Revision: 1.26 $ @ $Date: 2012/07/02 14:54:52 $

872
873               Copyright (c) 2011 Lockheed-Martin Company. All Rights Reserved.
874
875               This material may be reproduced by or for the U.S. Government
876               pursuant to the copyright license under the clause at
877               DoD FAR SUP 252.227-7014 (clause date).
878
```

```
879             "
880                shift
881                exit 0
882           ;;
883           -R|--recursive|--Recursive|-recursive)
884                export recur=1
885                export rflag=""
886                shift
887           ;;
888           -r)
889                export recur=2
890                export rflag="-R"
891                shift
892           ;;
893           -i|--interactive)
894                export ask=1
895                shift
896           ;;
897           -S|--search|--Search|--SEARCH|-search)
898                export search=1
899                shift
900                # temp9=$( echo ${*} | sed 's/ * AND  */\;/g' | sed 's/ * OR  */
    /g')
901                temp9=$( echo ${*} | sed 's/ * AND  */\;/g' | sed 's/ * OR
    */xs1b/g')
902                set -- ${temp9}
903                #echo "@=${@}"
904                #read fkfkfk
905           ;;
906             *)
907                if [ "${1:0:1}" == "-" ]; then
908                  myerror 2 ${1}
909                fi
910                if [ "${first}" -eq "0" -a "${search}" -eq "0" ]; then
911                # Must be object since we are first
912                   first=1
913                   object=${1}
914                   export objectset=1
915                   shift
916                else
917                 # must be keyword
918                   (( first=first+1 ))
919                   # echo "Before prepare key $*"
920                   # echo "1=$1"
921                   #read jdjdj
922                   preparekey ${1}
923                   shift
924                 #  echo "After prepare key $*"
925                fi
926            ;;
927     esac
928   done
929
930   #
931   # MAIN
932   # """"
933   #
```

```
934
935 #
936 # Now process fields into object
937 #
938
939 if [ ${first} -eq 0 ]; then
940    getout "Missing object/collection name on Sdata command line!"
941 fi
942
943 if [ ${search} -eq 0 ]; then
944   if [ ${first} -le 1 -a ${recur} -le 0  -a "${projectset}" -ne "1" ]; then
945 #   getout "No keyword operations were specified to operate on ${object}"
946 # If no keywords, then assume user wants to display all schemes for ${object}
947      displayall ${object}
948      myexit 0
949   fi
950   if [ "${projectset}" -eq 1 -a  ${recur} -eq 0 ]; then
951     if [ "${#project}" -ge "9" -a  "${#project}" -le "13" ]; then
952     Sscheme ${rflag} -w -val "Admin.HPCMP_Project_ID::${project}"  ${object}

953 # new JCI 6-22-2012
954       myexit 0
955     else
956       getout "Invalid Project ID entered ${project} . Please correct. "
957     fi
958   fi
959
960   if [ ${recur} -eq 0 ]; then
961     changeobject ${object}
962     myexit 0
963   fi
964   if [ ${recur} -eq 1 ]; then
965      export preflag=0
966     if [ ${first} -le 1 ]; then
967 #   getout "No keyword operations were specified to operate on ${object}"
968 # If no keywords, then assume user wants to display all schemes for ${object}
969       for i in $(Sls ${object}| awk '{print $1}'); do
970         if [ ${preflag} -eq 0 ]; then
971            export filepref=${i}
972            export preflag=1
973         else
974           i=${filepref}/${i}
975         fi
976         # echo -e "OUTLOOP:::: ${i} "
977           displayall ${i}
978       done
979        myexit 0
980     fi
981     if [ "${projectset}" -eq 1 ]; then
982       if [ "${#project}" -ge "9" -a  "${#project}" -le "13" ]; then
983         for i in $(Sls ${object}| awk '{print $1}'); do
984           Sscheme ${rflag} -w -val "Admin.HPCMP_Project_ID::${project}"  ${i}
985         done
986 # new JCI 6-22-2012
987         myexit 0
988       else
989         getout "Invalid Project ID entered: ${project} "
```

```
 990       fi
 991     fi
 992 #   echo "YULP" ; read djdjdj
 993     for i in $(Sls ${object}| awk '{print $1}'); do
 994      #echo "Read in \"${i}\""
 995      if [ ${preflag} -eq 0 ]; then
 996          export filepref=${i}
 997          export preflag=1
 998      else
 999          i=${filepref}/${i}
1000      fi
1001       changeobject ${i}
1002     done
1003     myexit 0
1004   fi
1005   if [ ${recur} -eq 2 ]; then
1006     if [ "${projectset}" -eq 1 ]; then
1007       for i in $(Sls ${object}| awk '{print $1}'); do
1008         Sscheme ${rflag} -w -val "Admin.HPCMP_Project_ID::${project}"  ${i}
1009       done
1010     fi
1011     changeobject ${object}
1012     myexit 0
1013   fi
1014 else
1015 # when in search mode there is no object !
1016   if [ "${explicitobjset}" -eq 0 ]; then
1017    export scollect=""
1018     if [ "${objectset}" -eq "1" ]; then
1019        preparekey ${object}
1020     else
1021       object="xs1a"
1022     fi
1023   else
1024     export scollect=${object}
1025     object="xs1a"
1026   fi
1027   if [ $recur -ge 0 ]; then
1028      rflag="-R"
1029   fi
1030   searchit 1
1031 #  getout "Should not be here at $LINENO !!"
1032   :
1033 fi
1034
1035 myexit 0
1036
```