```
#!/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
}
```

```

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

function pause {
  echo2 " SCRIPT PAUSED."
  read bleepo
  return
}

function cdate {
a=$(date +%Y%m%d%H)
return
}

# print compress date-time to expanded
function prdate {
typeset hours  datetime hours0  stringa  stringb stringc  stringd
datetime=${1}
hours0=$(echo ${datetime} | cut -c9-10 )
hours="${hours0} hundred hours"
stringa=$(echo ${datetime}| cut -c5-6 )
stringb=$(echo ${datetime}| cut -c7-8)
stringc=$(echo ${datetime}| cut -c1-4 )
string="${stringa}/${stringb}/${stringc} at ${hours}"
return
}

function myerror {

  case "${1}" in
     1)
        echo "User Error: Missing argument to option \"${2}\" !"
        exit 1
        ;;
     2)
       echo "Unknown option \"${2}\" given on command line!"
       exit 1
       ;;
     *) echo "Software error at $LINE , $LINENO"
       ;;
  esac
}

function displayall {
typeset lines tlines

  if [ ${schemeset} -eq 1 ]; then
    if [ "${scheme}" == "Name_Value" ]; then
      Sscheme ${rflag} -l -scheme Name_value ${1} |  grep -v '\ -row ' >
${temp2}
```

```
      lines=`wc -l ${temp2}| cut -f1 -d" "`
      (( lines = lines - 5 ))
      if [ "${preflag}" -eq 1 ]; then
          filepref2=$(echo ${1} | sed 's/\//\\\//g')
         tail -${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/^/${filepref2}:/g"
      else
         tail -${lines} ${temp2} | cut -c120- | sed 's/^$/xs1a/g' | tr '\012' ' ' | sed 's/xs1a/\n/g' | sed 's/\"  *\"/\=/g' |  sed 's/^  *//g'  |  tr -s '\012'
      fi
      echo " "
    elif [ "${scheme}" == "Dublin_Core" ]; then
    # Now the dublin core
      Sscheme ${rflag} -l -scheme 'Dublin_core' ${1} | grep -v '\ -row ' > ${temp2}
      lines=`wc -l ${temp2}| cut -f1 -d" "`
      (( lines = lines - 5 ))
      (( tlines = lines - 2 ))
      if [ "${preflag}" -eq 1 ]; then
      #  echo -e "${filepref}:\c";
        filepref2=$(echo ${1} | sed 's/\//\\\//g')
        tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/ *\"/=\"/g' | grep -v '""' |  sed "s/^/${filepref2}:/g"
      else

        tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/ *\"/=\"/g' | grep -v '""'
      fi
    #  echo " "
    fi
    return
  fi

#
# Display Everything because scheme was not set on command line
#
Sscheme ${rflag} -l -scheme Name_value ${1}  | grep -v '\ -row ' > ${temp2}
lines=`wc -l ${temp2}| cut -f1 -d" "`
(( lines = lines - 5 ))
if [ "${lines}" -ge "0" ]; then
  if [ "${preflag}" -eq 1 ]; then
    filepref2=$(echo ${1} | sed 's/\//\\\//g')
      tail -${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/^/${filepref2}:/g"
  else
     tail -${lines} ${temp2} | cut -c120- | sed 's/^$/xs1a/g' | tr '\012' ' ' | sed 's/xs1a/\n/g' | sed 's/\"  *\"/\=/g' |  sed 's/^  *//g'  |  tr -s '\012'
  fi
fi
# Now the dublin core

Sscheme ${rflag} -l -scheme 'Dublin_core' ${1} |  grep -v '\ -row ' > ${temp2}
lines=`wc -l ${temp2}| cut -f1 -d" "`
(( lines = lines - 5 ))
(( tlines = lines - 2 ))
```

```

if [ "${lines}" -ge "0" ]; then
      if [ "${preflag}" -eq 1 ]; then
        filepref2=$(echo ${1} | sed 's/\//\\\//g')
        tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/ *\"/=\"/g' | grep -v '""' | sed "s/^/${filepref2}:/g"
      else
       tail -${lines} ${temp2} | head -${tlines} | cut -c1-32,108- | sed 's/ *\"/=\"/g' | grep -v '""'
      fi
  #echo " "
fi
}

function preparekey {
typeset i  j   k  f  p1  p2
i=0; j=0; k=0; f=0; p1=0; p2=0
#       butill-0002.arl.hpc.mil>     Sscheme -l -scheme Comment_Scheme abutil*
#
#Sscheme  -w -val 'Admin.Retention_Period::90,Admin.Last_Review_Time::2011-04-25-13.48.05,Admin.HPCMP_Project_ID::HPCM092330SIS' test.txt
# You can see the Schemes with SgetS and the -x option lists all column names.
        if [ "${1}" == "" ]; then
                 cerr "missing operand in prepare object!"
        fi
       if [ ${search} -eq 0 ]; then
          echo ${*} |  sed 's/\;/\n/g' >> ${temp1}
       else
         echo ${*} | sed 's/\;/ AND\n/g' | sed 's/xs1b/ OR\n/g' >> ${temp1}
       fi
}

function findrow {

#echo "0,1,2=$0,$1,$2"
  export row=$(echo "($(Sscheme -l -scheme Name_Value ${1}  |  grep -v '\ -row ' | grep -n \"${2}\" | cut -f1 -d:)-6)/3" | bc)
  if [ "${row}" == "" ]; then
     echo "Soft error: row is null in findrow{} !"
  fi
  return
}

function askuser {
  typeset ent
  echo "
${1}
"
echo "Are you sure you want to do the above action?"
echo "Press Y or y, any other key means No"
read ent
if [ "${ent}" == "" ]; then
  ent=n
fi
if [ "${ent}" == "Y" -o "${ent}" == "y" ]; then
   export goyes=1
```

```
else
   export goyes=0
fi
echo " "
return
}

function maxrow {
#  echo "MAX1"
  export row=$(echo "(`Sscheme -l -scheme Name_Value ${1} | grep -v '\ -row ' |
wc -l`-6)/3" | bc)
#  echo "MAX2"
  return
}

#
# Search feature of Sdata
#

function searchit {
typeset i j k f p1 p2 srch addme
typeset -i lg lg2 setor setnot latch contand contand2
  lg=0
  lg2=0
  setor=0
  latch=0
  setnt=0
  contand=0
  contand2=0
  addme=""
  notl=""
  tn=""
  sq=$(echo -e '\047')
  srch="-policy \""

  if [ "${1}" == "" ]; then
    getout "missing operand in searchit!"
  fi
  which SgetD >/dev/null 2>&1
  if [ "$?" -ne "0" ]; then
    getout "*ERROR* Cannot locate SgetD command. Is the SRB module loaded?"
    echo "*ERROR $LINENO"
    myexit 1
  fi
  while read line ; do
          if [ ${contand} -ge 1 ]; then
               (( contand2=contand2+1 ))
          fi

#     echo "line=\"$line\""

       if [ "${latch}" -eq 1 -a  ${contand2} -lt 1 ]; then
         srch="${srch} ${addme}"
       fi
     if [ "${line:0:6}" == "SCHEME!" ]; then
          export scheme=${line:7}
          test ${verbose} -eq 1 && echo "New scheme detected and set to
```

```
\"${scheme}\" ..........."
            continue
     fi
# test for AND OR operator at end of line
     addme=" OR "
     lg=${#line}
     (( lg2=lg-4 ))
     k=${line:${lg2}:4}
     k=$( echo ${line}|awk '{print $NF}')
     case "${k}" in
      "AND")
#echo "IN CASE AND"
          (( contand=contand+1 ))
          latch=1
          addme=" AND "
          (( lg2=lg-4 ))
          line=${line:0:${lg2}}
       ;;
      "OR")
          latch=1
          addme=" OR "
          (( lg2=lg-3 ))
          line=${line:0:${lg2}}
# JCI new 6-19-2012
           if [ ${contand} -ge 1 ]; then
               while [ ${contand} -ge 1 ] ; do
                 srch="${srch})"
                 (( contand=contand-1 ))
               done
            fi
       ;;
       *)
       #   latch=1
       #   addme=" OR "
       #   (( lg2=lg2-1 ))
       #   line=${line:0:${lg2}}
# JCI 6-22-2012
          latch=1
          addme=" OR "
#          (( lg2=lg-3 ))
#          line=${line:0:${lg2}}
## JCI new 6-19-2012
#            if [ ${contand} -ge 1 ]; then
#               while [ ${contand} -ge 1 ] ; do
#                 srch="${srch})"
#                 (( contand=contand-1 ))
#               done
#            fi

       ;;
     esac
##
     echo ${line} | grep '=' 1>/dev/null 2>&1
# new JCI 6-19-2012
          if [ ${contand2} -ge 1 ]; then
#                (( contand2=contand2-1 ))
             srch="${srch} AND DATA_OBJECT.data_id IN (select
DATA_OBJECT.data_id where"
          fi
     if [ $? -eq 0 ]; then
    #    echo "2nd field present, this is a search=this thingy or search=NULL"
            p1=$(echo ${line} | cut -f1 -d=)
            p2=$(echo ${line} | cut -f2 -d=)
     #          echo "p1,p2=${p1},${p2}"
            stype=0
            tn=${p1:0:1}
            if [ "${tn}" == "!" ]; then
              notl="not "
              tn=${p1:1}
              p1=${tn}
            else
              notl=""
            fi

            echo ${p1} | egrep -i "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation Date$|^Type$|^Document ID$|^Rights$" >/dev/null 2>&1
            if [ $? -eq 0 ]; then
              test ${verbose} -eq 1 &&  echo "Dublin Core field detected, \"${p1}\", temporarily switching to Dublin_core scheme..."
              stype=1
              oldscheme=${scheme}
              scheme=Dublin_Core
            fi
            if [ "${p2}" == "" ]; then
            #  was delete but now keyword= should be interpreted as having the field contain null *******************************
            # SgetD -policy "(Name_Value.Name like color) AND (Name_value.value like red)"
              test ${verbose} -eq 1 && echo "Adding in search \"${p1}\" from \"${scheme}\" scheme"
              if [ ${stype} -eq 0 ]; then
              #
              # was  Sscheme ${rflag} -scheme Name_Value -d -row ${row} ${1} >/dev/null
              # SgetD -policy "(Name_Value.Name like color) AND (Name_value.value like red)"
                srch="${srch}(Name_Value.Name not like ${p1})"
                stype=0
              else
                # Sscheme ${rflag} -w -val "Dublin_Core.${p1}::" ${1} >/dev/null
                srch="${srch}(not like Dublin_core.${p1})"
              fi
            else
         # this is the search for this thingy in other words "field=this" search*****************************************
              test ${verbose} -eq 1 && echo "Changing/Inserting search ${notl}\"${p2}\" into \"${p1}\" field for \"${scheme}\" scheme for object \"${1}\""
              if [ ${stype} -eq 0 ]; then
               #   Sscheme ${rflag} -w -val "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}" >/dev/null
```

```
                 srch="${srch}(Name_Value.Name like ${p1}) AND (Name_value.value ${notl}like ${sq}${p2}${sq})"
              else
               #   Sscheme ${rflag} -w -val "Dublin_Core.${p1}::${p2}" "${1}" > /dev/null
               # SgetD -policy "(Dublin_Core.Title like Story*)"
                 # srch="${srch}(Dublin_Core.${p1} ${notl}like ${sq}${p2}${sq}*) "
                 srch="${srch}(Dublin_Core.${p1} ${notl}like ${sq}${p2}${sq}) "
              fi
            fi
     else  # if-then for equal sign present inside line or not ===============================================================================================
            p1=${line}
     #         echo "p1=${p1}"
            # was display ***************************************************************************************************
            # now for just showing if scheme has a field set******************************************************************
          stype=0
          echo ${p1} | egrep -i "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation Date$|^Type$|^Document ID$|^Rights$" >/dev/null
          if [ $? -eq 0 ]; then
             test ${verbose} -eq 1 &&  echo "Dublin Core field detected, \"${p1}\", temporarily switching to Dublin_core scheme..."
             stype=1
             oldscheme=${scheme}
             scheme=Dublin_Core
           fi
           test ${verbose} -eq 1 && echo "Search only for field is ${notl}present  \"${p1}\" field for \"${scheme}\" scheme "
           if [ ${stype} -eq 0 ]; then
            #   Sscheme ${rflag} -w -val "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}" >/dev/null
              # srch="${srch}(Name_Value.Name like ${p1}) AND (Name_value.value like ${p2})"
               srch="${srch}(Name_Value.Name ${notl}like ${p1})"
           else
              # srch="${srch} (Name_Value.Name like ${p1}) AND (Name_value.value like ${p2})"
              srch="${srch}(Dublin_core ${notl}like ${p1})"
           fi
           stype=0
#
# put in search string for searching a set field only here!
#

#
#
     fi  # if-then for equal sign present inside line or not =================================================================================================
# JCI new 6-19-2012
```

```
            if [ ${contand2} -ge 1 ]; then
              (( contand2=contand2-1 ))
                 srch="${srch})"
            fi
  done  < ${temp1}
#
  srch="${srch}\""

######################################################################
#
# Now pump search string into SgetD
#
test ${verbose} -eq 1 &&  echo "The search string is now set at ${rflag}
${srch} ${scollect}"
if [ "${explicitobjset}" -eq 0 ]; then
 :
#  eval SgetD ${rflag} ${srch}
   eval Sls ${rflag} ${srch}
else
 :
#  eval SgetD ${rflag} ${srch} ${scollect}
   eval Sls ${rflag} ${srch} ${scollect}
fi
myexit 0

#
#
#####################################################################

  return 0
}
function changeobject {
        typeset i j k f p1 p2
#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
# You can see the Schemes with SgetS and the -x option lists all column names.
        if [ "${1}" == "" ]; then
                 cerr "missing operand in changeobject!"
        fi
        exec 5<${temp1}
        which Sscheme >/dev/null 2>&1
        if [ $? -ne 0 ]; then
                getout "*ERROR* Cannot locate Sscheme command. Is the SRB
module loaded?"
        fi
#       echo ${2} |  sed 's/\;/\n/g' > ${temp1}
##      for i in $(cat ${temp1}); do
         while read -u5 line ; do
              #echo "line=$line"
              if [ "${line:0:6}" == "SCHEME!" ]; then
                  export scheme=${line:7}
                  test ${verbose} -eq 1 && echo "New scheme detected and set to
\"${scheme}\" ..........."
                  continue
              fi
              echo ${line} | grep '=' 1>/dev/null 2>&1
              if [ $? -eq 0 ]; then
              #    echo "2nd field present"
                  p1=$(echo ${line} | cut -f1 -d=)
                  p2=$(echo ${line} | cut -f2 -d=)
#                 echo "p1,p2=${p1},${p2}"
                  stype=0
                  echo ${p1} | egrep -i "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation Date$|^Type$|^Document ID$|^Rights$" >/dev/null 2>&1
                  if [ $? -eq 0 ]; then
                    test ${verbose} -eq 1 &&  echo "Dublin Core field detected, \"${p1}\", temporarily switching to Dublin_core scheme..."
                    stype=1
                    oldscheme=${scheme}
                    scheme=Dublin_Core
                  else
# if not dublin, then name_value, there could be more in the future so this may need to be changed
                    scheme=name_value
                  fi
                  if [ "${p2}" == "" ]; then
                  # delete ******************************************************************************************************
                     if [ "${enforced}" -eq 0 ]; then
                        echo "The \"-d\" switch was NOT specified on the command line. Ignoring request to delete ${p1} from ${1} !"
                        continue
                     fi
                     if [ "${ask}" -eq "1" ]; then
                         askuser "Delete \"${p1}\" from \"${scheme}\" scheme for object \"${1}\""
                         if [ "${goyes}" -eq "0" ]; then
                           continue
                         fi
                     fi
                    test ${verbose} -eq 1 && echo "Deleting \"${p1}\" from \"${scheme}\" scheme for object \"${1}\""
                    if [ ${stype} -eq 0 ]; then
                    #
                     findrow "${1}" "${p1}"
                     if [ ${row} -lt 0 ]; then
                         echo "*ERROR* There is no field named ${p1} for object ${1}! Ignoring delete..."
                         continue
                     fi
                       [ ${DEBUG2} -eq 1 ] && echo "The delete would look like this ---==>>> Sscheme -scheme Name_Value -d -row ${row} ${1}"
                      Sscheme ${rflag} -scheme Name_Value -d -row ${row} ${1} >/dev/null
                    else
                    #  Sscheme ${rflag} -w -val "Dublin_Core.${p1}::${p2} ''" ${1}   >/dev/null
                      [ ${DEBUG2} -eq 1 ] && echo  Sscheme ${rflag} -w -val "Dublin_Core.${p1}::" ${1}
                      Sscheme ${rflag} -w -val "Dublin_Core.${p1}::" ${1} >/dev/null
                    fi
                  else
                # change/insert ************************************************************************************************
## Kludge for removing double-double quotes
#                    p4=$(echo ${p2} | sed 's/\"\"/\"/g')
#                    p2=${p4}
                       if [ "${ask}" -eq "1" ]; then
                          askuser "Changing/inserting \"${p2}\" into \"${p1}\" field for \"${scheme}\" scheme for object \"${1}\""
                         if [ "${goyes}" -eq "0" ]; then
                           continue
                         fi
                       fi
                    test ${verbose} -eq 1 && echo "Changing/Inserting \"${p2}\" into \"${p1}\" field for \"${scheme}\" scheme for object \"${1}\""
                    if [ ${stype} -eq 0 ]; then
                  # Sscheme -w -val 'Name_Value.Name[0]::Mw0,Name_Value.Value[0]::zeroth' abutil.txt
                      findrow "${1}" "${p1}"
 #                    echo "row=${row}"
                      if [ ${row} -lt 0 ]; then
                      # Name not there, so place at end
                        maxrow "${1}" "${p1}"
                        if [ ${row} -gt 20 ]; then
                           echo "There is no more room in Name-Value table to place ${p1} for object \"${1}\"! Ignoring insert request!"
                           continue
                        fi
                      fi
                      [ ${DEBUG2} -eq 1 ] && echo Sscheme ${rflag} -w -val "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}"
                      Sscheme ${rflag} -w -val "${scheme}.Name[${row}]::${p1},Name_Value.Value[${row}]::${p2}" "${1}" >/dev/null
                    else
                  # Sscheme -w -val 'Dublin_Core.title::Bizarre Rituals of the West Phillians' abutil.txt
                       [ ${DEBUG2} -eq 1 ] && echo Sscheme ${rflag} -w -val "Dublin_Core.${p1}::${p2}" "${1}"
                      Sscheme ${rflag} -w -val "Dublin_Core.${p1}::${p2}" "${1}"     > /dev/null
                    fi
                  fi
                  continue
              else
                  p1=${line}
#                 echo "p1=${p1}"
               # display ********************************************************************************************************
                stype=0
               # echo "preflag=$preflag"
                echo ${p1} | egrep -i "^Title$|^Creator$|^Subject$|^Description$|^Publisher$|^Contributor$|^Creation Date$|^Type$|^Document ID$|^Rights$" >/dev/null
                if [ $? -eq 0 ]; then
                   test ${verbose} -eq 1 &&  echo "Dublin Core field detected, \"${p1}\", temporarily switching to Dublin_core scheme..."
                   stype=1
                   oldscheme=${scheme}
                   scheme=Dublin_Core
                 fi
                test ${verbose} -eq 1 && echo "Display item inside \"${p1}\" field for \"${scheme}\" scheme in object \"${1}\""
              fi
              if [ "${stype}" -eq "1" ]; then
              # display Dublin field
               if [ "${preflag}" -eq 1 ]; then
                  filepref2=$(echo ${1} | sed 's/\//\\\//g')
                  [ ${DEBUG2} -eq 1 ] && echo "Sscheme -l -scheme Dublin_Core ${1} "
                 Sscheme -l -scheme Dublin_Core ${1}  |  grep -v '\ -row ' | grep ${p1} | egrep -o '\".*\"$' | sed "s/^/${filepref2}:/g"
                 #Sscheme -l -scheme Dublin_Core ${filepref2} | grep ${p1} | egrep -o '\".*\"$' | sed "s/^/${filepref2}:/g"
                else
                  [ ${DEBUG2} -eq 1 ] && echo "Sscheme -l -scheme Dublin_Core ${1}"
                  Sscheme -l -scheme Dublin_Core ${1}  |  grep -v '\ -row ' | grep ${p1} | egrep -o '\".*\"$'
                fi
              # turn off dublin
                 stype=0
                 scheme=${oldscheme}
              else
              # display field inside user scheme or other scheme
              # Sscheme -l -scheme name_value abutil.txt | grep -A1 'string\[16\]' |  grep test
                 [ ${DEBUG2} -eq 1 ] && echo  "Sscheme -l -scheme name_value ${1}"
                Sscheme -l -scheme name_value ${1} | grep -v '\ -row ' | grep 'string\[16\]' |  grep ${p1} >/dev/null 2>&1
                if [ $? -eq 0 ]; then
               # match!
                  if [ "${preflag}" -eq 1 ]; then
                    filepref2=$(echo ${1} | sed 's/\//\\\//g')
                    [ ${DEBUG2} -eq 1 ] && echo  "Sscheme -l -scheme name_value ${1}"
                    Sscheme -l -scheme name_value ${1} | grep -v '\ -row ' | grep -A1 'string\[16\]' | grep -A1 ${p1} | tail -1 | egrep -o '\".*\"' | sed "s/^/${filepref2}:/g"
                   # echo "YULP"
                   # read n
                  else
                    [ ${DEBUG2} -eq 1 ] && echo  "Sscheme -l -scheme name_value ${1}"
                     Sscheme -l -scheme name_value ${1} | grep -v '\ -row ' | grep -A1 'string\[16\]' | grep -A1 ${p1} | tail -1 | egrep -o '\".*\"'
                  fi
                else
               # no match
                 test ${verbose} -eq 1 && echo "Field \"${p1}\" was NOT FOUND
```

```
for object \"${1}\" !"
                fi
              fi
         done
#          done  < ${temp1}
#          done  <&5
         return
}

function displayhelp {

      echo '

      Sdata - Set/modify/delete/show/search metadata on objects within a SLM collection

      SYNOPSIS

      Sdata {-R} {-o|--object} object_name {[-p|--project]PROJECT} {-c <collection>} {keyword{= {value}} { keyword{={value}} ...} {OPTIONS}

      DESCRIPTION
      Sdata allows one to display, set, change or delete keyword-value pairs or the project in the Storage Resource Broker (SRB) metadata.
      When the "-S" or "--search" option is provide, Sdata will search for files containing metadata (see Sdata In Search Mode below)

      OPTIONS
      -c, --collection  set the collection to operate
      -d, --delete Enforce deletions for "keyword=" keywords
      -f, --force  ignore nonexistent files, never prompt
      -h,--help  display this help and exit
      -i, --interactive  prompt before setting metadata
      -o, --object  SRB object or objects
      -p, --project  set the project code for object
      -R, --recursive  operate on SRB object metadata contents recursively
      -s,--scheme choose scheme to display/select
      -S,--search run Sdata in search mode to find files that match a metadata line arguments
      -v, --verbose  explain what is being done
      --version  output version information and exit
      -x, --xml insert/parse xml

      keyword{={value}}

      For each object_name, the {keyword{={value}}|..} will perform actions:

      "keyword" is NOT provided,  all metadata associated with object_name is displayed to standard output
      "keyword" is provided,  metadata associated with keyword is displayed to standard output  "keyword=" is provided,  metadata associated with keyword is DELETED
      "keyword=value" is provided,  metadata associated with "keyword" is inserted/changed to "value"

      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.

      EXAMPLES :

      Sdata -o MyObj "Creator=John Doe"  This command will set the Creator attribute for the object MyObj in the current collection to "John Doe".

      Sdata -o MyObj  This command will display all metadata to standard output for the object MyObj in the current collection.

      Sdata -d -o MyObj "Creator="  This command will DELETE the Creator attribute value for the object MyObj in the current collection to "John Doe".

       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.

       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.

       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.

'
echo '

       Sdata In Search Mode:
       ==========================================================================
       (NB: All other command line flags MUST come before the "-S" flag! )

       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

       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

       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

       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

        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


       '
       myexit 0
```

```
}

function xtest {
 typeset f
   echo "----------------------====================================vvvvvvvvvvvvvvvvvvvvvvvvvvvvv"
   if [ "${2}" != "" ]; then
     echo " ******************* Test Type: ${2} "
   fi
   echo "Testing \"${1}\"  ......"
   echo " "
#   eval "${1} --verbose"
   eval "${1}"
#   if [ $? -ne 0 ]; then
#      echo "----====>> PROBLEM WITH \"${1}\" !!!"
#      return 1
#   fi
   echo "-------------===========================^^^^^^^^^^^^^^^^^^^^^"
   echo " "
   }

function gotest {
 typeset f
echo "

    *******************************************
   ********** Running Internal Tests ***********
    *******************************************

   "

   f=abutil.txt.$$
   echo "This is a test of the Sdata command" > ${WORKDIR}/${f}
   echo "You are currently in SRB path:"
   Spwd
   if [ $? -ne 0 ]; then
      echo "** SEVERE ERROR! Cannot \"Spwd\""
      exit 1
    fi
    echo " "
   Sput ${WORKDIR}/${f} .
   if [ $? -ne 0 ]; then
      echo "** SEVERE ERROR! Cannot \"Sput ${f} .\""
      exit 1
    fi
xtest "Sdata ${f} theory=MP2 subject='Physical Organic Chemistry'" "Inserting fields"
xtest "Sdata ${f}"  "Display All Special Metadata Fields"
xtest "Sdata ${f} --scheme Name_Value" "Display Only Metadata Associated with Name_Value Scheme"
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"
xtest "Sdata ${f} " "Examine the output to verify previous command has run correctly."
xtest "Sdata ${f} PROGRAM AUTHOR theory"  "Display AUTHOR and Theory fields"
xtest "Sdata ${f} ARTHUR" "Cannot display field which does not exist"
xtest "Sdata ${f} -d  MW=" "Delete Mw field"
xtest "Sdata ${f}"  "Examine the output to verify previous command has run
correctly."
xtest "Sdata ${f} -d AUTHOR MW=393.4343 EXCELFILENAME='H2SO4-H2O_3.xls' theory
AUTHOR= ISOMER="  "Modify/Insert/Delete/Display various fields"
xtest "Sdata ${f}"  "Examine the output to verify previous command has run
correctly."

echo "

Testing is Finished.

"
}

#
# Settings
#

export DEBUG2=0
# Cores per node
CORESPERNODE=${BC_CORES_PER_NODE:-8}
# memory available to user in GB
MEMPERNODE=${BC_MEM_PER_NODE:-17}
export ask=0
object=
export project=
export projectset=0
collection=
first=0
export verbose=0
export row=0
export enforced=0
 export schemeset=0
#

#
# Start of script
#

if [ -z "${USER}" ]; then
   getout "USER env variable not set!!!"
fi

if [ -z "${WORKDIR}" ]; then
#   getout "WORKDIR env variable not set!!!"
   export WORKDIR=/usr/var/tmp/${USER}
fi

#out=${WORKDIR}/${USER}
out=${WORKDIR}
temp1=${out}/sdata.$$.temp1.out
cerr "Cannot create temp1"
temp2=${out}/sdata.$$.temp2.out
cerr "Cannot create temp2"
export recur=0
export verbose=0
export scheme=Name_Value
export rflag=""
export project=xxxx
export projectset=0
export search=0
export objectset=0
export explicitobjset=0
export filepref=""
export preflag=0


if [ ! -d ${out} ];  then
   mkdir -p ${out}
   cerr "Cannot mkdir -p ${out}"
fi

if [ -z  "${SAMPLES_HOME}" ]; then
#   getout "SAMPLES_HOME directory is not set!!!"
   export SAMPLES_HOME=/usr/cta/SCR
fi

project=$(cat /etc/passwd | egrep -e "^${USER}:" | cut -f2 -d\( | cut -f1 -d\))
#echo "Project = <${project}> "
#if [ -z ${project} ]; then
#       getout "Could not obtain users project id from passwd!!"
#fi

while [ "$#" -ge 1 ]; do
  case "${1}" in
      -d|--delete|--DELETE|-delete)
          export enforced=1
           shift
       ;;
      -h|--help|-H|--HELP|--Help|-help|-HELP|-Help)
          displayhelp
          shift
           exit 0
       ;;
       -t|--test)
           #echo "Test 1, value=${2}"
           shift
           gotest
           exit 0
           ;;
       --debug2)
           export DEBUG2=1
           shift
           ;;
        -o|--obj*|--Obj*)
           if [ "${2}" == "" ]; then
                   myerror 1 ${1}
           fi
           object=${2}
           export objectset=1
           export explicitobjset=1
           first=1
```

```
            shift ; shift
            ;;
        -x|--xml|--XML)
            if [ "${2}" == "" ]; then
                    myerror 1 ${1}
            fi
            myxml=$(2)
            echo "XML to parse: ${2}"
            shift ; shift
            ;;
         -p|--proj*|--PROJ*|--Proj*|-proj*)
            if [ "${2}" == "" ]; then
                    myerror 1 ${1}
            fi
            export project=${2}
            export projectset=1
            shift ; shift
           ;;
         -c|--collect*|--Collect*|-collect*)
            if [ "${2}" == "" ]; then
                    myerror 1 ${1}
            fi
            collection=${2}
            object=${2}
            export objectset=1
            export explicitobjset=1
            first=1
            shift ; shift
           ;;
         -s|--scheme|--Scheme|--SCHEME|-scheme)
            if [ "${2}" == "" ]; then
                    myerror 1 ${1}
            fi
#           if [ "${2}" == "Name_Value" -o "${2}" == "name_value" -o "${2}" -o
"${2}" == "Name_value"
            export scheme=${2}
            export schemeset=1
            echo "SCHEME!${2}" >> ${temp1}
            shift ; shift
           ;;
           -v|--verbose|--Verbose)
               export verbose=1
               shift
           ;;
           --version|--Version|-version)
             echo "


             Sdata Beta RCS ID: $Revision: 1.26 $ @ $Date: 2012/07/02 14:54:52 $


              Copyright (c) 2011 Lockheed-Martin Company. All Rights Reserved.

              This material may be reproduced by or for the U.S. Government
              pursuant to the copyright license under the clause at
              DoD FAR SUP 252.227-7014 (clause date).

             "
               shift
               exit 0
           ;;
            -R|--recursive|--Recursive|-recursive)
               export recur=1
               export rflag=""
               shift
           ;;
           -r)
               export recur=2
               export rflag="-R"
               shift
           ;;
            -i|--interactive)
               export ask=1
               shift
           ;;
            -S|--search|--Search|--SEARCH|-search)
                export search=1
                shift
                # temp9=$( echo ${*} | sed 's/ * AND  */\;/g' | sed 's/ * OR  *//g')
                temp9=$( echo ${*} | sed 's/ * AND  */\;/g' | sed 's/ * OR  */xs1b/g')
                set -- ${temp9}
                #echo "@=${@}"
                #read fkfkfk
           ;;
             *)
                if [ "${1:0:1}" == "-" ]; then
                  myerror 2 ${1}
                fi
                if [ "${first}" -eq "0" -a "${search}" -eq "0" ]; then
                # Must be object since we are first
                   first=1
                   object=${1}
                   export objectset=1
                   shift
                else
                 # must be keyword
                   (( first=first+1 ))
                   # echo "Before prepare key $*"
                   # echo "1=$1"
                   #read jdjdj
                   preparekey ${1}
                   shift
                 #  echo "After prepare key $*"
                fi
            ;;
   esac
 done

 #
 # MAIN
 # """"
 #
```

```

#
# Now process fields into object
#

if [ ${first} -eq 0 ]; then
   getout "Missing object/collection name on Sdata command line!"
fi

if [ ${search} -eq 0 ]; then
  if [ ${first} -le 1 -a ${recur} -le 0  -a "${projectset}" -ne "1" ]; then
#   getout "No keyword operations were specified to operate on ${object}"
# If no keywords, then assume user wants to display all schemes for ${object}
     displayall ${object}
     myexit 0
  fi
  if [ "${projectset}" -eq 1 -a  ${recur} -eq 0 ]; then
    if [ "${#project}" -ge "9" -a  "${#project}" -le "13" ]; then
    Sscheme ${rflag} -w -val "Admin.HPCMP_Project_ID::${project}"  ${object}

# new JCI 6-22-2012
      myexit 0
    else
      getout "Invalid Project ID entered ${project} . Please correct. "
    fi
  fi

  if [ ${recur} -eq 0 ]; then
    changeobject ${object}
    myexit 0
  fi
  if [ ${recur} -eq 1 ]; then
     export preflag=0
    if [ ${first} -le 1 ]; then
#   getout "No keyword operations were specified to operate on ${object}"
# If no keywords, then assume user wants to display all schemes for ${object}
      for i in $(Sls ${object}| awk '{print $1}'); do
        if [ ${preflag} -eq 0 ]; then
           export filepref=${i}
           export preflag=1
        else
          i=${filepref}/${i}
        fi
        # echo -e "OUTLOOP:::: ${i} "
          displayall ${i}
      done
       myexit 0
    fi
    if [ "${projectset}" -eq 1 ]; then
      if [ "${#project}" -ge "9" -a  "${#project}" -le "13" ]; then
        for i in $(Sls ${object}| awk '{print $1}'); do
          Sscheme ${rflag} -w -val "Admin.HPCMP_Project_ID::${project}"  ${i}
        done
# new JCI 6-22-2012
        myexit 0
      else
        getout "Invalid Project ID entered: ${project} "
      fi
    fi
#   echo "YULP" ; read djdjdj
    for i in $(Sls ${object}| awk '{print $1}'); do
     #echo "Read in \"${i}\""
     if [ ${preflag} -eq 0 ]; then
         export filepref=${i}
         export preflag=1
     else
         i=${filepref}/${i}
     fi
      changeobject ${i}
    done
    myexit 0
  fi
  if [ ${recur} -eq 2 ]; then
    if [ "${projectset}" -eq 1 ]; then
      for i in $(Sls ${object}| awk '{print $1}'); do
        Sscheme ${rflag} -w -val "Admin.HPCMP_Project_ID::${project}"  ${i}
      done
    fi
    changeobject ${object}
    myexit 0
  fi
else
# when in search mode there is no object !
  if [ "${explicitobjset}" -eq 0 ]; then
   export scollect=""
    if [ "${objectset}" -eq "1" ]; then
       preparekey ${object}
    else
      object="xs1a"
    fi
  else
    export scollect=${object}
    object="xs1a"
  fi
  if [ $recur -ge 0 ]; then
     rflag="-R"
  fi
  searchit 1
#  getout "Should not be here at $LINENO !!"
  :
fi

myexit 0

```