```
#!/usr/bin/env python
try:
        import os, glob, sys, getopt
        import tempfile
        from SrbRegisterUtil import execMe,execSMe,PyIsSinit,CheckPyVersion
except:
        raise "Check you local Python Version. A minimum Python version required"
""" A script to query Metadata inside SRB """


SCHEME_NAME = "Name_Value"

def usage():
        print """SrbQueryUtil.py [--explore | --coll | --plot] <options>
  There are three exclusive modes available
    --explore [--name] | "key1" "key2" "key3" ... "keyN"
      List metadata names and possible values from existing SRB collections

    --coll <argument(s)>
      List collections that match all criterias provided in one or more
      key::value pairs

      "key1::value1" "key2::value2" "key3::value3" ... "keyN::valueN"   : Input
        parameters restraints

    --plot <arglist>
      Create 2 dimensional plot file from existing SRB collections that match
      all criterias provided in one or more key::value pair

      arglist:
        --xaxis <param>       : X-axis from Readme.xml. Ascending (program
                                      will strip non numeric characters)
         --yaxis <param>       Y-axis from Readme.xml
         --output <filename> : ASCII filename to store output
         --collection              : [Optional] Base SRB collection. If blank,
                                      current Collection is used

         "key1::value1" "key2::value2" "key3::value3" ... "keyN::valueN"    :
                                      Input parameters restraints

  Generic options
    -h | --help        : This help text
    --version          : Print the script version

  Examples

  Explore mode examples
    To list valid metadata keys from existing SRB collections:
      SrbQueryUtil.py --explore --name

    To list valid metadata values (usually gotten from SrbQueryUtil.py
    --explore --name) with key "PenMaterial", "StudyName", and
    "PenStrModelPara":
      SrbQueryUtil.py --explore "PenMaterial" "StudyName" "PenStrModelPara"

  Collection mode example
    To list collection with StudyName is "WHA RHA study":
```

```
58       SrbQueryUtil.py --coll "StudyName::WHA RHA study"
59
60   Plot mode example
61     To create a 2 dimensional chart with x axis being StrikingVel
62     and y axis being PenDepth,output file is myoutput.123, base
63     collection name is /home/margom.nirvana, with the following
64     restraints "PenMaterial::93W-5Ni-2Fe" "StudyName::WHA
65     RHA study" "PenStrModelPara::Weerasooriya" :
66
67     SrbQueryUtil.py --plot --xaxis StrikingVel --yaxis PenDepth
68       --output myoutput.123 --collection  /home/margom.nirvana
69       "PenMaterial::93W-5Ni-2Fe" "StudyName::WHA RHA study"
70       "PenStrModelPara::Weerasooriya"
71 """
72 # end function
73
74
75 def version():
76         print """$Header:
   /cvs_repository/customers/HPCMP/testbed/NavyPilot/SrbQueryUtil.py,v 1.4
   2013/04/11 17:18:50 martin Exp $"""
77 # end function
78
79 def printOneIndent(string):
80         print '\t' + string
81 #end function
82
83 def parseSschemeOutput(rawInput):
84         """ Parse output (with multiple lines) """
85         arrOneLine = rawInput.split('\n')
86
87         for strLine in arrOneLine:
88                 if len(strLine) > 0:
89                         arrElem = strLine.split()
90                         printOneIndent( (" ".join(arrElem[5:])).replace("\"","""))
91 #end of for
92
93
94
95 def plotMode():
96         xaxis           = ''
97         yaxis           = ''
98         output          = '' #output file
99         arrKeyValue     = ''
100        collection      =       ''
101
102        try:
103                opts,args = getopt.getopt(sys.argv[1:], "",
   ['plot','collection=','xaxis=', 'yaxis=','output='])
104        except getopt.GetoptError, err:
105                print str(err)
106                usage()
107                sys.exit(2)
108
109        for arg,val in opts:
110                if arg == "--collection":
111                        collection = val
```

```
112                 elif arg == "--xaxis":
113                         xaxis = val
114                 elif arg == "--yaxis":
115                         yaxis = val
116                 elif arg == "--output":
117                         output = val
118                 elif arg == "--plot":
119                         pass
120                 else:
121                         usage()
122                         sys.exit()
123
124         #end for
125         try:
126                 fileOut = open(output, 'w')
127         except IOError:
128                 print "Unable to open file {0}".format(output)
129                 sys.exit()
130         #end try
131
132         arrKeyValue = args
133         fileOut.write("# Query Criteria : {0}\n".format(args))
134
135
136         if xaxis == '' or yaxis == '' or output == '' or arrKeyValue == '':
137                 print "Missing mandatory option xaxis, yaxis, output, or key
    value"
138                 usage()
139                 sys.exit()
140
141         fileOut.write("# x axis : {0}        y axis : {1}\n".format(xaxis,yaxis))
142
143         if PyIsSinit():
144                 print "Sinit is OK..."
145                 status = 0
146         else:
147                 print "No SRB session detected. Have you run Sinit yet?"
148                 sys.exit(2)
149
150
151         if len(arrKeyValue) > 0 :
152
153                 ## Try to unpack key_value
154
155                 subCmd = "SgetD -R -policy \""
156                 count  = 0
157                 for term in arrKeyValue:
158                         if count > 0:
159                                 subCmd = subCmd + " AND "
160
161
162                         innerKey = term.split('::')[0]
163                         innerValue = term.split('::')[1]
164
165                         subCmd = subCmd + " DATA_OBJECT.data_id IN (select
    DATA_OBJECT.data_id where Name_Value.Name = '{0}' AND Name_Value.Value like
    '{1}')".format(innerKey,innerValue)
```

```
166
167                         count = count + 1
168
169                 #end for
170
171                 subCmd = subCmd + "\""  #end double quote before the collection
    name
172                 if collection != '':
173                         subCmd = subCmd + " " + collection #append collection
    name, if any
174
175                 print subCmd
176                 (rc,out) = execSMe(subCmd)
177                 if rc != 0:
178                         print "SQL query error. Perhaps an input parameter is
    incorrectly typed"
179                         rc = -1
180
181                 print out
182
183         #end if
184
185         #Phase 2: Get the output from SgetD -R and extract X and Y axis from SRB
    MCAT
186
187         xaxisVal = ''
188         yaxisVal = ''
189         if rc == 0:
190                 arrOutput = out.split('\n')
191                 arrOutput = arrOutput[2:] #I filter the first two lines. They
    are header information from SgetD
192                 for strEntry in arrOutput:
193                         arrEntry = strEntry.split()
194                         try:
195                                 strAbsCol = arrEntry[0] + "/" + arrEntry[1]
196                         except:
197                                 continue
198
199                         try:
200                                 if arrEntry[2] != 'collection':
201                                         print "This entry {0} is not a
    collection, skipping".format(strAbsCol)
202                                         rc = -1
203                                         continue
204                         except:
205                                 rc = -1
206
207                         if rc == 0:
208                                 #WARNING: Parsing sixth column from Sscheme
    output. There is a dependency to SRB 2012 R3 Sscheme output column. If Sscheme
    output format changes, this command has to be revised
209                                 subCmd = "Sscheme -l -scheme {0} {1} | egrep -1
    '{2}|{3}'  ".format(SCHEME_NAME, strAbsCol, xaxis,yaxis)
210                                 (rc,out) = execSMe(subCmd)
211                                 if rc != 0:
212                                         print "Unable to find datapoint for
    collection {0}. Skipping".format(strAbsCol)
```

```
213                                 else:
214                                         arrVal= out.split('\n')
215                                         for index, strLine in enumerate(arrVal):
216                                                 arrStrLine = strLine.split()
217                                                 if len(arrStrLine)>5:
218
219                                                         try:
220                                                                 if
    arrStrLine[5].replace('\"','') == xaxis:
221                                                                         xaxisVal
    = arrVal[index+1].split()[5].replace('\"','')
222                                                                 elif
    arrStrLine[5].replace('\"', '') == yaxis:
223                                                                         yaxisVal
    = arrVal[index+1].split()[5].replace('\"','')
224                                                                 #end if
225                                                         except Exception:
226                                                                 print "Unable to
    parse Sscheme output. Skipping"
227                                                 #end if len
228                                         #end for
229                                         if xaxisVal != '' and yaxisVal != '':
230                                                 fileOut.write("{0}
    {1}\n".format(xaxisVal,yaxisVal))
231                                                 xaxisVal = ''
232                                                 yaxisVal = ''
233
234                                 #end if rc != 0
235                         #end else
236                 #end for
237         #end if
238
239         fileOut.write("\n") # End it with a new line character. Some computers
    have problem reading files without endline character
240         fileOut.close()
241         if rc == 0:
242                 print "Success. Please review output file at :
    {0}".format(output)
243         else:
244                 print "Error detected. Please review error message, fix, and try
    again"
245 #end function
246
247 def collMode():
248         arrKeyValue     = ''
249         try:
250                 opts,args = getopt.getopt(sys.argv[1:], "", ['coll'])
251         except getopt.GetoptError, err:
252                 print str(err)
253                 usage()
254                 sys.exit(2)
255
256         arrKeyValue = args
257
258         if len(arrKeyValue) > 0 :
259                 if PyIsSinit():
260                         print "Sinit is OK..."
```

```
261                         status = 0
262                 else:
263                         print "No SRB session detected. Have you run Sinit yet?"
264                         sys.exit(2)
265
266                 ## Try to unpack key_value
267
268                 subCmd = "SgetD -R -policy \""
269                 count  = 0
270                 for term in arrKeyValue:
271                         if count > 0:
272                                 subCmd = subCmd + " AND "
273
274
275                         innerKey = term.split('::')[0]
276                         innerValue = term.split('::')[1]
277
278                         # This specifies the value as a non-exact match. TODO: Add flag for exact match.
279                         subCmd = subCmd + \
280                                 " DATA_OBJECT.data_id IN (select DATA_OBJECT.data_id where Name_Value.Name = '{0}' AND Name_Value.Value like '{1}')".format(innerKey,innerValue)
281
282                         count = count + 1
283                 #end for
284
285                 subCmd = subCmd + "\""  #end double quote before the collection name
286
287                 #print subCmd
288                 (rc,out) = execSMe(subCmd)
289                 if rc != 0:
290                         print "SQL query error. Perhaps an input parameter is incorrectly typed"
291                         rc = -1
292
293         #end if
294         else:
295                 print "At least one key::value pair must be specified.\n"
296                 usage()
297                 sys.exit()
298
299         #Phase 2: Get the output from SgetD -R and extract collection, name, and value into columns
300
301         if rc == 0:
302                 arrOutput = out.split('\n')
303                 arrOutput = arrOutput[2:] #I filter the first two lines. They are header information from SgetD
304
305                 matchingCollections = 0
306                 for strEntry in arrOutput:
307                         arrEntry = strEntry.split()
308                         try:
309                                 # Absolute path to collection
310                                 strAbsCol = arrEntry[0] + "/" + arrEntry[1]
```

```
311                                     #print "strAbsCol: ", strAbscol
312                             except:
313                                     continue
314
315                             try:
316                                     if arrEntry[2] != 'collection':
317                                             print "This entry {0} is not a
    collection, skipping".format(strAbsCol)
318                                             rc = -1
319                                             continue
320                                     else:
321                                             matchingCollections =
    matchingCollections + 1
322                             except:
323                                     rc = -1
324
325
326                             if rc == 0:
327                                     print strAbsCol
328                                     #WARNING: Parsing sixth column from Sscheme
    output. There is a dependency to SRB 2012 R3 Sscheme output column. If Sscheme
    output format changes, this command has to be revised
329                                     subCmd = "Sscheme -l -scheme {0} {1} | egrep -w
    'Name|Value' | grep string".format(SCHEME_NAME, strAbsCol)
330                                     (rc,out) = execSMe(subCmd)
331
332                                     if rc != 0:
333                                             print "Unable to find name/value pairs
    for collection {0}. Skipping".format(strAbsCol)
334                                     else:
335                                             arrVal= out.split('\n')
336
337                                             # Loop, incrementing index by 2
338                                             #for index, strLine in [(2*i,l) for i,l
    in enumerate(arrVal)]:
339                                             for index in range(0,len(arrVal),2):
340                                                     try:
341                                                             strNameLine =
    arrVal[index]
342                                                             arrStrNameLine =
    strNameLine.split()
343                                                             arrNameQuoteSplit =
    strNameLine.split('"')
344
345                                                             valIndex = index + 1
346                                                             strValLine =
    arrVal[valIndex]
347                                                             arrStrValLine =
    strValLine.split()
348                                                             arrValQuoteSplit =
    strValLine.split('"')
349                                                     except Exception:
350                                                             break
351
352                                                     name = ''
353                                                     val = ''
354                                                     if len(arrStrNameLine)>5:
```

```
355                                                             try:
356                                                                     if
    arrStrNameLine[0] == 'Name':
357                                                                             name =
    arrNameQuoteSplit[1]
358                                                                     #end if
359                                                             except Exception:
360                                                                     print "Unable to
    parse Sscheme output. Skipping"
361                                                                     break
362                                                     #end if len
363                                                     if len(arrStrValLine)>5:
364                                                             try:
365                                                                     if
    arrStrValLine[0] == 'Value':
366                                                                             val =
    arrValQuoteSplit[1]
367                                                                     #end if
368                                                             except Exception:
369                                                                     print "Unable to
    parse Sscheme output. Skipping"
370                                                                     break
371                                                     #end if len
372                                                     print
    "\t{0}::{1}".format(name,val)
373
374                                             #end for
375
376                                     #end if rc != 0
377                             #end else
378                     #end for
379             #end if
380
381             print "{0} matching collections found".format(matchingCollections)
382
383
384             if rc == 0:
385                     print "Success."
386             else:
387                     print rc
388                     print "Error detected. Please review error message, fix, and try
    again"
389 #end function
390
391 def exploreMode():
392             name_mode       =0
393             path                    = ''
394             rc                              = 0
395             out                     = ''
396
397             try:
398                     opts,args = getopt.getopt(sys.argv[1:], "",  ['explore','name'])
399             except getopt.GetoptError, err:
400                     print str(err)
401                     usage()
402                     sys.exit(2)
403
```

```
404         for arg,val in opts:
405                 if arg == "--name":
406                         name_mode = 1
407                 elif arg =="--explore":
408                         pass
409                 else:
410                         usage()
411                         sys.exit()
412
413         #end for
414
415         if PyIsSinit():
416                 print "Sinit is OK..."
417                 status = 0
418         else:
419                 print "No SRB session detected. Have you run Sinit/Sshell yet?"
420                 sys.exit(2)
421
422
423         if len(args) > 0  or name_mode == 1:
424
425                 # Try to unpack key_value
426
427                 #Save overhead here by running Sscheme -l once and parse the
    output multiple times with help of temporary files
428
429                 subCmd = "Sscheme -l -scheme {0}  | egrep -w 'Name|Value' | grep
    'string'".format(SCHEME_NAME)
430                 (rc,out) = execSMe(subCmd)
431
432                 if (rc == 0):
433
434                         fd, path = tempfile.mkstemp()
435                         os.write(fd,out)
436                         os.close(fd)
437                 else:
438                         print "Failed to read Metadata Schema from MCAT. Exiting"
439                         rc = -1
440                 #end if
441
442
443                 if (rc == 0):
444                         if (name_mode == 1):
445                                 print "Name mode"
446                                 subCmd = "cat {0} | grep Name | sort |
    uniq".format(path)
447                                 (rc,out) = execSMe(subCmd)
448                                 if (rc != 0):
449                                         print "Error getting Name listing"
450                                 else:
451                                         parseSschemeOutput(out)
452
453                         #end name mode
454                         else:
455                                 print "Value mode"
456                                 for strMetaName in args:
457                                         print "Valid Metadata Values for key
```

```
    {0}".format(strMetaName)
458
459                                         subCmd = "cat {0} | grep -A 1 -i {1} |
    grep -w Value | sort | uniq".format(path,strMetaName)
460                                         (rc,out) = execSMe(subCmd)
461
462                                         if (rc != 0):
463                                                 print "Error getting values for
    parameter {0}".format(strMetaName)
464                                         else:
465                                                 parseSschemeOutput(out)
466                                         #end if
467
468                                 #end of for loop
469                         #end if Value mode
470                 #end if
471
472         if (rc == 0):
473                 print "Success"
474         else:
475                 print "Error detected. Please review error message, fix, and try
    again"
476
477         try:
478                 os.remove(path) #cleanup
479         except:
480                 pass
481
482 #end function
483
484 def main():
485
486         if not CheckPyVersion():
487                 sys.exit(1)
488
489         currMode = ''
490         try:
491                 opts,args = getopt.getopt(sys.argv[1:], "hv", ['help',
    'version','name','coll','plot','explore','collection=','xaxis=','yaxis=','output=
    '])
492         except getopt.GetoptError, err:
493                 print str(err)
494                 print
495                 usage()
496                 sys.exit(2)
497
498         for arg,val in opts:
499                 if arg == "-h" or arg == "--help":
500                         usage()
501                         sys.exit()
502                 elif arg == "--version" or arg == "-v":
503                         version()
504                         sys.exit()
505                 elif arg == "--coll":
506                         if currMode == '':
507                                 currMode = 'COLL'
508                         else:
```

```
509                                     print "Coll mode is exclusive, please remove
    plot or explore mode."
510                                     print
511                                     usage()
512                                     sys.exit()
513                 elif arg =="--plot":
514                         if currMode == '':
515                                     currMode = 'PLOT'
516                         else:
517                                     print "Plot mode is exclusive, please remove
    coll or explore mode."
518                                     print
519                                     usage()
520                                     sys.exit()
521                 elif arg == "--explore":
522                         if currMode == '':
523                                     currMode = 'EXPLORE'
524                         else:
525                                     print "Explore mode is exclusive, please remove
    coll or plot mode."
526                                     print
527                                     usage()
528                                     sys.exit()
529
530         #end for
531
532         print """
533 SrbQueryUtil.py script
534 ------------"""
535         if currMode == 'COLL':
536                 collMode();
537         elif currMode == 'PLOT':
538                 plotMode();
539         elif currMode == 'EXPLORE':
540                 exploreMode();
541         else:
542                 print "Unrecognized Mode"
543                 usage()
544                 sys.exit()
545
546
547
548
549
550
551 #end of function
552
553 if __name__=="__main__":
554         main()
555
556
```