Lua by Tasks

Contents

[Basic 1](#_Toc448928041)

[Iterate 1](#_Toc448928042)

[Lua as SAS macro replacement: list processing + read SAS dataset 3](#_Toc448928043)

[Call Proc FCMP functions in Lua 5](#_Toc448928044)

[Submit SAS Code 5](#_Toc448928045)

[Read sas dataset 6](#_Toc448928046)

http://blog.sina.com.cn/s/blog\_72ef7bea0102vzi7.html

## Basic

**proc** **lua**;

submit;

--this is a Line Comment

--http://support.sas.com/resources/papers/proceedings15/SAS1561-**2015.**pdf

--**0.**When PROC LUA initializes the Lua state, it creates a special global Lua table called sas

print(sas) -->table: 000000000D8F8F80

--**1.** hello world

print("hello world")

--[[this is a Block Comment

**2.** declare variables

--]]

local pi = **3.14**

local guest = 'Lua'

print(pi); print(guest)

print(pi **..** " " **..** guest)

--**3.** table as array

local shoppinglist = {'milk', 'flour', 'eggs', 'sugar'}

print(shoppinglist)

--like SAS DATA step arrays, arrays in Lua start at index **1** by default

local drink = shoppinglist[**1**]

print(drink)

endsubmit;

**run**;

## Iterate

**proc** **lua**;

submit;

local shoppinglist = {'milk', 'flour', 'eggs', 'sugar'}

--**4.** iterate

--[[

The ipairs function returns two values with each iteration:

the current index into the array (assigned to variable i in the preceding example) and

the value of the item at that index in the array (assigned to variable item)**.**

--]]

for i, item in ipairs(shoppinglist) do

print(i, item)

end

--**5.** table as dictionary (hash)

local band = {

vocals='Robert Plant',

guitar='Jimmy Page',

bass='John Paul Jones',

drummer='John Bonham'

}

--[[

**6.** d pairs (similar to ipairs) that iterates through all the key/value combinations in the dictionary

--]]

for key, value in pairs(band) do

print(key, value)

end

endsubmit;

**run**;

/\*pairs vs ipairs\*/

**proc** **lua**;

submit;

t = {fred='one',[**0**]=**1**; **10**,**20**,**30**,**40**}

for k,v in pairs(t) do

print(k,v)

end

for i,v in ipairs(t) do

print(i,v)

end

endsubmit;

**run**;

## Lua as SAS macro replacement: list processing + read SAS dataset

/\*

https://www.youtube.com/watch?v=7G5Mb--iTc8

\*/

**data** work.mytables;

length in out by $**32**;

in = "sashelp.cars";

out = 'work.cars';

by ="make model";

output;

in = "sashelp.pricedata";

out = 'work.pricedata';

by ="date";

output;

in = "sashelp.class";

out = 'work.class';

by ="age height";

output;

**run**;

**proc** **lua**;

submit;

function sort\_b(ds)

local dsid =sas.open(ds)

while sas.next(dsid) do

local data = sas.get\_value(dsid,'in')

local out = sas.get\_value(dsid,'out')

local by = sas.get\_value(dsid,'by')

sas.submit[[

proc sort data=@data@ out=@out@;

by @by@;

**run**;

]]

end

sas.close(dsid)

end

sort\_b("work.mytables")

endsubmit;

**quit**;

**%macro** sortt(ds);

%local dsid;

%let dsid = %sysfunc(open(&ds));

%do %while (not %sysfunc(fetch(&dsid)));

%let in = %sysfunc(getvarc(&dsid,%sysfunc(varnum(&dsid,in))));

%let out = %sysfunc(getvarc(&dsid,%sysfunc(varnum(&dsid,out))));

%let by = %sysfunc(getvarc(&dsid,%sysfunc(varnum(&dsid,by))));

%put in = &in;

%put out = &out;

%put by = &by;

proc sort data = &in out = &out;

by &by;

run;

%end;

**%mend**;

%***sortt***(mytables)

**data** \_null\_;

set mytables end = eof;

call execute('proc sort data = '|| in || " " ||"out = " || out || " ; ");

call execute('by ' || " " || by || " ;");

call execute ('run;');

**run**;

**data** \_null\_;

set mytables end = eof;

call execute(catx(' ','proc sort data = ',in ,' out = ', out, ';'));

call execute(catx(' ','by ', by, ';'));

call execute ('run;');

**run**;

**data** \_null\_;

set mytables end = eof;

call execute('proc sort data = '|| in || " " || "out = " || out || " ; " ||'by ' || " " || by || " ;" ||'run;' );

**run**;

**data** \_null\_;

set mytables end = eof;

call execute(catx(' ','proc sort data = ',in,"out = ",out, " ; " ,'by ',by , " ;" ,'run;' ));

**run**;

## Call Proc FCMP functions in Lua

**proc** **fcmp** outlib=work.foo.foo;

function sumx(x[\*]);

sum = **0**;

do i = **1** to dim(x);

sum = sum + x[i];

end;

return (sum);

endsub;

**run**;

options cmplib=work.foo;

**proc** **lua**;

submit;

local array = {**1**,**2**,**3**,**4**}

local sum = sas.sumx(array)

print(sum)

endsubmit;

**run**;

## Submit SAS Code

**proc** **lua**;

submit;

sas.submit('proc print data=sashelp.iris(obs=2);run;')

sas.submit[[

proc print data=sashelp.class(obs=**2**);

**run**;

]]

local ds ="sashelp.aarfm(obs=2)"

sas.submit[[

proc print data=@ds@;

**run**;

]]

sas.submit([[

proc print data=@indat@;

**run**;

]], {indat = 'sashelp.heart(obs=2)'})

endsubmit;

**run**;

## Read sas dataset

**proc** **lua**;

submit;

local dsid = sas.open('sashelp.class')

for row in sas.rows(dsid) do

print(row.name,row.age)

end

sas.close(dsid)

endsubmit;

**run**;

**proc** **lua**;

submit;

local dsid = sas.open('sashelp.class')

while sas.next(dsid) do

print(sas.get\_value(dsid,"name"),sas.get\_value(dsid,'age'))

end

sas.close(dsid)

endsubmit;

**run**;

**proc** **lua**;

submit;

local dsid = sas.open("sashelp.class")

local nvars = sas.nvars(dsid)

while sas.next(dsid) do

for i=**1**,nvars do

print(sas.get\_value(dsid,i))

end

end

sas.close(dsid)

endsubmit;

**run**;

**proc** **lua**;

submit;

local dsid = sas.open("sashelp.class")

for var in sas.vars(dsid) do

print("var=", table.tostring(var))

end

sas.close(dsid)

endsubmit;

**run**;

**proc** **lua**;

submit;

local dsid = sas.open("sashelp.class")

-- get info for variable 'name'

print("name=", table.tostring(sas.varinfo(dsid,"name")))

sas.close(dsid)

endsubmit;

**run**;