accessCode=688-327-941 locale=en-US

1 -

data x;

input v1 $ v2 8.7;

cards;

Hello 123.032

Hihig 213.04

;

run;

P.S: the semicolne should be under the data def not at the row of the data,example we cannot here put ; near 213.04

2-

data data\_1;

infile "/home/u38430500/external-csv/other\_del\_data.csv" dsd ; -> here dsd mean that by default it's , as sperator and it's used to treat 2 consecutive , as missing

input name $ gender $ age weight ;

run;

data data\_1;

infile "/home/u38430500/external-csv/other\_del\_data.txt" dsd dlm=':'; -> dlm is the deliminter, we can also used on a txt file

input name $ gender $ age weight ;

run;

3-

Tim M14510/21/1978

Sara 13009/20/1964

Mike M18011/23/1965

LauraF13011/06/1980

Sean M16704/07/2000

data data\_1;

infile "/home/u38430500/external-csv/data\_unwell formated.txt" dsd dlm=':'; -> here we are saying to sas that for each line, take name from char(1) to char(5), gender char(6), weight char(7) to char (9) and s on

input name $ 1-5 gender $ 6 weight 7-9 dob $ 10-19 ;

run;

4 -

@ point to the column position

5 -

@ sholud allway be before the variable

/ is to go to the next row

#3 go to row 3

we should alway precise the value size;

P.S: mmddyy10. is a datatype of type date and size 10

P.S: 3.2 means we are dealing with a number of size 3 with decimal 2 decimal

data data\_1;

infile "/home/u38430500/external-csv/data\_unwell formated.txt" dsd dlm=':';

input @1 name $5. @6 gender $1. @7 weight 2. @10 dob mmddyy10. ; -> here name value will start from col1 (@1) and take 5 space ($5)

format dob mmddyy.

run;

6-

libname mylib '/home/u38430500/mydata'; here we are creating a lib when a dataset is bound to this lib it will stores its info inside the folder that comes with this lib

7-

proc import datafile='/home/u38430500/external-csv/excel\_data.xlsx' dbms=xlsx

out=data\_excel replace;

getnames=no;

sheet='score'; ->here we are getting a specefic sh

range="score$A1:b6"; -> here we getting a sheet with specefic range

run;

8 -

proc import datafile='/home/u38430500/external-csv/DATA\_blanks.txt' dbms=dlm

out=data\_txt replace;

guessingrows=2;

delimiter=' ';

getnames=no; -> here to auto set the name if not present sas will consider the first data row as the cols name

run;

proc import datafile='/home/u38430500/external-csv/DATA\_commas.csv' dbms=csv

out=data\_csv replace;

getnames=no;

delimiter=',';

run;

9-

using sum and mean as function is not as doing by hands, function take care of missing values

10 -

if x NE . -> means if col x is not missing then do something

11 -

if a string lenght pass the one specified in the lenght it will be truncated;

12 -

-> here split means we are putting the ;label in 2 lignes with reference to \*

proc print data=work.scoredata1 label split='\*';

label score1='Math \* Test' score2 = 'Science \* Test' score3 = 'Language \* Test' Avg ='Average \* Score';

run;

proc print data=work.scoredata1 ;

run;

13 -

while formatting we need to set the date size correctly, $ , . are counted , so we need to set size to take into account those chars

14 -

proc format;

value $GenderAsChar 'm'='Male' 'f'='Female' other = 'Unkown';

run;

proc print data=work.scoredata1 label ;

format avg dollar7.3 gender $GenderAsChar.;

run;

15 -

we use the lib statement to store the format

proc format lib=mywork;

value $GenderAsChar 'm'='Male' 'f'='Female' other = 'Unkown'; -> here we need to specify the variable type before

run;

proc format lib=mywork fmtlib; ->to print the formats we have in this library, the variable will remain the same but the display will be different

run;

option fmtsearch = (work mywork library); -> this ligne is to tell sas from where to fetch stored formats

16 -

the lenght of a variable is determined by the first reference of this varaible for the exception if it was preceded by lenght or input statment

17 -

if we have an all numeric string sas will try to converted to numeric if it's not possible it will produce missing

so we need to use the input and the put to make conversion

18 -

input function is used to convert from string to numeric, here we need to specify the variable and it's informat

the Put function is the inverse of input

if we have str \*5 -> if str is full numric the result will be a number otherwise it will be empty

score\_ac=input(raw\_score,7.) \*5; -> here the conversion will work rwa\_score is a full numeric so format 7 will converted

19 -

in scan function we can use many deliminetre, the scan can use them singly or a combo of successive ones

20 -

substr(x,i,j) represent a portion of x from i to i+j

substr(x,i,) represent a portion of x from i to the lsat char of x

substr(x,i,j) = str ->we are modifying value of x by replacing char from i to i+j with str

21-

if the lenght of a variable depasse the value of the specify length it will be naturly truncated

22 -

index(x,str) to find the position of str in x,

23 -tranwrd(x,y,z) replace y with z in x

24 - int(x) is to take the integer portion of the value

25 -

fullDob=mdy(month,day,year);

FIX\_DATE=MDY(12,31,1994);

26 - when putting a $ with the input format in the input statement we are telling SAS to treat a variable as string with a specefic format

$MMDDYY10. -> SAS will assume that the varaible is number and need to be translated to a string

MMDDYY10. -> the date is an string but transformed into numeric

26 - fullDob is number

syear=year(fullDob);

smonth=month(fullDob);

sday=day(fullDob);

sqtr=qtr(fullDOb);

sweekday=weekday(fullDOb);

x=intck('day', start\_date, today()); ->is like date difference, day for difference in day, month for difference in month year for difference in years

P.S this function can take date as string or number

27 -

YRDIF(date1,date2,format='30/360',ACT/ACT or ..)

DATDIF same as YRDIF

28 -

data work.x;

Intrest=0.03;

Base=2150.35;

Total=Base;

do year=5 to 1 by -1; -> here we are decreasing , looping in reverse order

Total=(Intrest+1)\*(Total+1500\*12);

output;

end;

format Total euro13.2;

run;

29 -

data work.x;

Intrest=0.01;

Total=2150.35;

keep Year;

do until(Total gt 100000); -> Loop until the condition is true, when it's true stop

Total=(Intrest+1)\*(Total+1500\*12);

Year+1;

end;

run;

data work.x;

Intrest=0.01;

Total=2150.35;

keep Year;

do while(Total lt 100000);-> Loop until the is false, as long as it's true continue;P.S: if the first iteration result a false the loop will stop

Total=(Intrest+1)\*(Total+1500\*12);

Year+1;

end;

run;

30 -

data work.x;

Intrest=0.01;

Total=2150.35;

keep Year;

do Year=1 to 3 until(Total gt 100000); -> here we iterate until total > 100000 however if the number of iteration reach 3 it will stop the loop even if total didnt reach the goal

Total=(Intrest+1)\*(Total+1500\*12);

output;

end;

run;

31 -

dim(array) -> to get the lenghts

Array scores [\*] array with unknow size that represent all the elements

Array scores [3] \_numeric\_; -> P.S: an array with variables from dataset will modify the variablesof this dataset if the content of the array is modified

Array xyz [3] \_temporary\_ (1 2 3); -> init with fix value, \_temporary\_ is say that this array will not included it's varaibles in dataset

Array scores [3] score:; -> is a notation to say that the array all varaibles prefixed with scoer, P.S: the keyword of is only needed when calling a function

32 -

proc sort data=y out=z dupout=zz nodupkey; -> nodupkey is to eleminate any dup after sorting, nodupkey is to extract everything that have been elimenated to a dataset

by descending gender ;

run;

33-

proc print obs='test' data=y obs='test'; -. putting an observation name,the name of observation column

var name score:;

Id name; -> here name is considerd to be the obs , simply we put id and the vras names

where score1 in (88,95,67);

run;

34 -

proc print obs='test' data=z obs='test';

var score: gender; -> the vars to show

id name;

sum score:; -> creating a row for displaying a sum for some variable score1 to score3

run;

35 -

proc print obs='test' data=z obs='test' sumlabel='Sum of Scores' grand\_label='Total of all classes';

var score: gender;

id name;

sum score:;

by gender; since we have a by statement the sum will function as sum pergroup and for all groups, the by is way to say how we are grouping

run;

36 -

proc means maxdec=2 n mean median max min; maxdec the number of decimals , n mean median max min are the ops to view

var score1-score3;

by class ; by or class are doing the same thing

run;

37 -

the \* mean show gender as variation class

proc freq ;

tables gender\*class/missing nocol norow nopercent nofreq; -> nocol mean dont display col pct,norow -> no display rowpct nofreq -> dont display the frequncy, noprecent dont show percentage

run;

p.s: missing will show freq of missing value

38 -

proc univariate data=z;

var score1-score3 avg ;

class class;

run;

39-

proc export data=y(where=(avg>80)) dbms=csv

outfile='/home/u38430500/export/rapport2.csv' replace;

delimiter='@';

run;

proc export data=y(where=(avg>80)) dbms=xlsx

outfile='/home/u38430500/export/rapport2.csv' replace;

sheet='test';

run;

40 -

proc sql ;

create table scoredata3 as select \* from scoredata1 where averagescore>80; -> create table scoredate3 will actualy create a dataset scoredate3, and the from scoredate1 is actualy taking data from scoredata1 dataset

quit;

41 -

ods pdf file='/home/u38430500/export/test-1.pdf' style=Daisy;

ods rtf file='/home/u38430500/export/test-1.rtf';

ods powerpoint file='/home/u38430500/export/test-1.ppt';

proc print ....

ods pdf close;

ods rtf close;

ods powerpoint close;

or we can use ods \_all\_ close;

42 -

ods excel file="/home/u38430500/export/export.xlsx"

options(

sheet\_interval="bygroup" -> here we are specifying that we need to divide our excel to multiple sheets with tables that results from the bygroup

sheet\_label="Score" -> prefix the sheet name

embedded\_titles="yes" -> title option will be included in excel

embed\_titles\_once="yes"

);

ods noproctitle; -> supress the proc mean

proc sort data=scoredata0;

by gender;

run;

proc print data=scoredata0;

Id name;

by gender;

title 'Test';

where score1 ne 60;

run;

ods \_all\_ close;

43 -

%let sales=sale:; -> here we are defining varaibles name, P.S it's not a string

%let exportName = '/home/u38430500/export/export.xlsx'; -> here we are defining s a string, single quote are required to define a string

ods excel file="&exportName" style=Ocean -> here to reference a macro varaible as a part of a string we use "&macrovaraiblename"

proc means data=score1 maxdec=3 mean n max min median;

var &sales; -> here simply we are saying that vars of mean proc will be from sales Macro variable

by state;

run;

44 -

here we can use those macro by simply invoking: %dataset or dataset1 %print -> %macroname

example: data %dataset; means data score1

example: set %dataset1; means set scoredata0

example %print; we are calling everything inside print macro

%macro dataset;

score1

%mend dataset;

%macro dataset1;

scoredata0

%mend dataset1;

%macro print;

proc print data=%dataset;

run;

%mend;

45-

%macro print(vars,Id);

proc print data=scoredata0;

var &vars; -> the params are considered macro variables and not macros so we refere to them by the end

id &id;

run;

%mend;

%print(year saleCA1,year);

46 -

to refer to a macro varaible we use & to refer to a macro we use %

47 -

inner -> get everything in common between the 2 tables

LEFT JOIN:

It joins two or more tables, returns all records from the left table, and matching rows from the right-hand table.

RIGHT JOIN:

It is used to join two or more tables, returns all records from the right table, and matching rows from the left-hand table.

proc sql;

create table data3

as select \* from data1 inner join data2 on

data1.stu\_id=data2.stu\_id;

quit;

48 -

PUTLOG 'ERROR: testing ' \_ERROR\_ \_N\_ stu\_id= math=; -> here we are litteraly telling sas to throw a error by putting the error Label in the string

PUTLOG ' testing ' \_ERROR\_ \_N\_ stu\_id= math=;-> we are simply logging

ERROR '???' \_ERROR\_ \_N\_ stu\_id= math=; -> here we are saying that there is a error , not throwing one P.S: the \_N\_ specify the iteration index, \_ERROR\_ the flag, it's 0 when no error occured, and 1 when there is error or when put ERROR instead of PUTLOG

49 -

here we are tranforming 3 rows of the name and transform it to one row with score1 score2 score3 , before trans each ligne have one score

data mylib.oneper;

set mylib.manyper;

by stu\_id name ;

Array score [3] 3.;

retain score1-score3; -> we need the retain because each data ietration will reset the score array,

if first.name then

call missing(of score1-score3);

score[stime]=all\_score;

if last.name then

output;

keep name stu\_id score1-score3;

run;

50 -

here we are fixing stu\_id and name , so for each couple stu\_id and name we have 3 rows,

example:

x y 1 2 3 -> the by statement contains x y , the var contains 1 2 3

the 1 2 3 will be 1

2

3

so the final data set will be : x y 1

x y 2

y y 3

proc transpose data=mylib.oneper out=mylib.x( rename=(col1=score \_name\_=score\_type) drop=\_label\_ where=(score ne .));

by stu\_id name;

var score1-score3 ;

id anything; it mean that that a rotated value should specified in the columnname = anything

run;

51-

1- line 1 of a will be extended by line 1 of b, line 2 of a with ligne 2 of b and so on.

2- if a have a vraible called v1 and b have the same variable, the v1 in b will be taken into account since it will the last data set in the order;

3- the number of lines in results will the number of line of the smallest

data x;

set a;

setb;

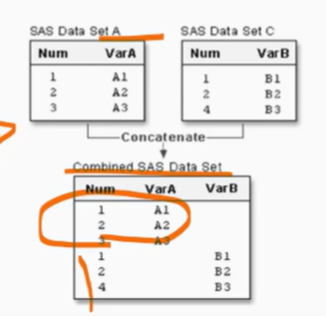
run;

52 – if num in a have a length != then num in c , we take the length of num a, and the same for label,informat and…

Data x;

Set a c;

Run; here num on both side should be with the same type otherwise it’s an error



53-

Proc append base=a;

Data=b force;

Run;

Here we are adding b under a; so basically and b should have the same variable definition,

In case not the results will have variables definition same as a (from name point of view, size and so on)

So in case of type mismatch with variables with the same name, the values in b will be replaced by missing, in case b have varaibles that are not in a they will discarded, in case of length miss math with varaibles of the same name the varaibles in b will be truncated to whats in a