Using automatic macros;

```
footnote "dataset created on &sysdate &systime";

title "OS &sysscp using release &sysver";

data work.fish;

set sashelp.fish (obs=50);

run;

proc print data=fish;

run;
```

User defined macros;

```
%put _all_;     (each macro variable name is written to SAS log with auto or global label)

%put The value of countries is : &country;     (writes only to SAS log the resolved macro reference)

options symbolgen;     (specify if you want to see the value of macro variable reference)

%let country = 'Asian';

%let conti = Asia;

title "Vehicles from &country countries";

data cars;

set sashelp.cars;

where origin = "&conti";     (must be specified in double quotes else macro variable reference is not resolved)

run;

proc print data = cars;

run;
```

%str function;

```
%let prog =
%str (data newset;
    x=1;
    y=2;
    run;);

&prog
proc print;
run;
```

%let text = %str (New Chap%'s Example);

Or {%let text = %bquote (New Chap's Example); can also be used}

```
%let prog =
%str (data newset;
    x=1;
    y=2;
    run;);

title "&text";
&prog
proc print;
run;
```

%nrstr function;

```
%let texts = %nrstr (Nov&Dec);
%let prog =
%str (data newset;
    x=1;
    y=2;
    run;);

title "&text";
footnote "&texts";
&prog
proc print;
run;
```

%upcase function;

```
%let dvsn = ne;
data baseball;
set sashelp.baseball;
where div = "%upcase (&dvsn)";
run;

proc print data = baseball;
run;
```

%sysfunc function;

```
title "report created on %sysfunc (today(), worddate.)";
proc print data = sashelp.fish;
where weight > 200;
run;
```

Macro in a data step;

```
data bpltable;
set newlib.bpltable;
cl + 1;
if points >=75 then quali + 1;
if quali < cl then do;
  %let comp = european football;
 end;
else do;
  %let comp = domestic football;
 end;
run;

proc print data = bpltable;
footnote "&comp for top 5";
run;
```

modified code;

```
data epltable;
set newlib.bpltable;
cl + 1;
if points >= 75 then quali + 1;
if quali < cl then do;
  call symput ('comp','european football');
 end;
else do;
  call symput ('comp','domestic football');
 end;
run;

proc print data = epltable;
footnote "&comp for top 5";
run;
```

(compare the above two codes for understanding the difference)

Defining a macro;

options mcompilenote = noautocall; *(issues a note to log after macro compilation) (values none & all can also be used instead of noautocall)*

%macro **prntlast**;

proc print data = &syslast; *(auto macro variable which stores the name of the last data set created)*

title "most recent dataset created : &syslast";

run;

%mend;

%prntlast *(called a macro call, semicolon not required)*

```
data main;
input id$ var1 var2 var3;
cards;
a 1 2 3
b 4 5 6
c 7 8 9
;
run;

%macro calculations (x);
 &x._diff = var3 - var2 - var1;
 &x._sum = var1 + var2 + var3;
 &x._squared = (var1 + var2 + var3) ** 2;
```

```
  &x._cube = (var1 + var2 + var3) ** 3;
%mend calculations;

data new_main;
set main;
%calculations (var4);
run;
```

Positional parameters;

```
%macro printdsn (dsn, vars);
        proc print data = &dsn;
        var &vars;
        title "output of %upcase(&dsn) data set";
        run;
%mend;

%printdsn (sashelp.cars, make origin)     (substitute values instead of dsn and vars)
%printdsn (sashelp.fish, species height weight width)
```

Keyword parameters;

```
%macro printdsn (dsn = sashelp.gas, vars = fuel nox);
        proc print data = &dsn;
        var &vars;
        title "output of %upcase(&dsn) data set";
        run;
%mend;
%printdsn     {can also use %printdsn ()}
```

%printdsn (dsn = sashelp.failure, vars = cause process) *(to invoke the printdsn macro with different data set name and variables)*

Mixed parameter lists;

```
%macro printdsn (dsn, vars = date);
        proc print data = &dsn;
        var &vars;
        title "output of %upcase(&dsn) data set";
        run;
%mend;
```

```
%printdsn (sashelp.citiday, vars = date)
%printdsn (sashelp.citiwk)
```
(default values of parameter vars used)

%Global statement;

```
%macro printdsn;
%global dsn vars;
%let dsn = sashelp.cars;
%let vars = make model msrp;
        proc print data = &dsn;
        var &vars;
        title "contents of &dsn";
        run;
%mend printdsn;
%printdsn
```

%Local statement;

```
%let dsn = sashelp.air;

%macro localmacro;
        %local dsn;
        %let dsn = sashelp.gas;
        %put value of local dataset is &dsn;
%mend;

%localmacro    (without this the log wont display the %put message inside the macro)
%put value of global dataset is &dsn;
```

%if - %then statement;

```
%macro lastset;
        %if &syslast ne _null_ %then %do;
        proc print data = _last_ (obs=25);
        title "last data set created is %lowcase(&syslast)";
        run;
        %end;
%mend;
%lastset
%macro cars (country, baseprice = 20000, highprice = 35000);
        %let baseprice = &baseprice;
        %let highprice = &highprice;
                proc freq data = sashelp.cars;
                where msrp between &baseprice and &highprice;
                table make / nocum;
                title "MSRP between &baseprice and &highprice";
        %if &country= %then %do;
        title2 "for all origins";
        %end;
                %else %do;
                title2 "for cars originating in &country";
                where also origin = "&country";
                %end;
%mend;
%cars (Asia)
```

Storing and accessing SAS macros;

%include statement;

```
%macro printlast;
        proc print data = &syslast;
        title "Last Dataset Created : &syslast";
        run;
%mend;
```
(save this code as a file in a folder)

```
%include '/folders/myfolders/Macros/printlast.sas' /source2;
```
(invoking the macro)(source2 displays the code in the log)

```
%printlast
```

Using the autocall facility to access the macros;

```
options mautosource sasautos = ('/folders/myfolders/macros' , sasautos);
```
(works without fileref sasautos)

```
%printlast
```

Stored compiled macros;

```
options mstored sasmstore = newlib;

%macro sortlast (sortby) /store;

        proc sort data = &syslast out = sorted;

        by &sortby;

        run;

%mend;

options mstored sasmstore = newlib;

%sortlast (msrp)
```

Proc catalog;

```
proc catalog cat = newlib.sasmacr;

contents;

run;
```

%copy;

```
options mstored sasmstore = newlib;

 %macro freq (that) /store source;

        proc freq data = &syslast;

        table &that;

        run;

 %mend;

%copy freq / source;
```

(copy the source code to the sas log)

Creating a macro in proc sql;

```
proc sql noprint;
select sum(fee) format=dollar10. into :totalfee
from sasuser.all;
quit;
%let totalfee=&totalfee;
proc means data=sasuser.all sum maxdec=0;
class course_title;
var fee;
title "Grand Total for All Courses Is &totalfee";
run;
```

Symbol tables;

```
%let dsn=sasuser.courses;
%macro printdsn;
%local dsn;
%let dsn=sasuser.register;
%put The value of DSN inside Printdsn is &dsn;
%mend;
%printdsn
%put The value of DSN outside Printdsn is &dsn;
```

```
%macro printdsn;
%global dsn vars;      (can also be used outside of a macro definition)
%let dsn=sasuser.courses;
%let vars=course_title course_code days;
proc print data=&dsn;
var &vars;
title "Listing of &dsn data set";
run;
%mend;
%printdsn
```

Multiple local symbol tables;

```
%macro outer;
%local variX;      (only remains till the execution of this macro)
%let variX=one;
%inner
%mend outer;
%macro inner;
%local variY;      (only remains till the execution of this macro)
%let variY=&variX;
%mend inner;
%let variX=zero;     (goes to global symbol table)
%outer
```