```
/* MACROS IN SAS: */


/* PART-1:---> */


/* IT GIVE 'SUV' TYPE SUMMARY */

PROC SQL;
CREATE TABLE SUV_SUMM AS
SELECT TYPE,
       ORIGIN,
       COUNT(*) AS NO_OF_UNITS,
       SUM(MSRP) AS TOTAL_MSRP,
       SUM(INVOICE) AS TOTAL_INVOICE
       FROM SASHELP.CARS
       WHERE UPCASE(TYPE)="SUV"
       GROUP BY 1,2;
QUIT;

PROC PRINT DATA=WORK.SUV_SUMM;
RUN;

/* WHEN WE CHANGE THE 'SUV' TO OTHER TYPE,WE MANUALLY DO IT.
BUT TO AUTOMATE THIS THING WE USE MACROS. */



/* INTRODUCTION TO MACROS: */

/* MACROS ALLOWS US TO AVOID REPETITIVE SECTIONS OF CODE AND TO USE THEM AGAIN AND AGAIN WHEN NEEDED.
IT ALSO HELPS CREATE DYNAMIC VARIABLES WITHIN THE CODE THAT CAN TAKE DIFFERENT VALUES FOR DIFFERENT
        RUN INSTANCES OF THE SAME CODE. */


/* MOST IMPORTANT--> THING TO IDENTIFY THE VALUES WHICH NEEDS TO BE CHANGED. */

/* START WITH (MANDETORY)-->  %MACRO  AND THE NAME OF THE MACRO */
/* END WITH (MANDETORY)-->  %MEND */


%MACRO TYPE_SUMM_AUTO(CAR_TYPE);

PROC SQL;
CREATE TABLE &CAR_TYPE._SUMM AS
SELECT TYPE,
       ORIGIN,
       COUNT(*) AS NO_OF_UNITS,
       SUM(MSRP) AS TOTAL_MSRP,
       SUM(INVOICE) AS TOTAL_INVOICE
       FROM SASHELP.CARS
       WHERE UPCASE(TYPE)="&CAR_TYPE."
       GROUP BY 1,2;
QUIT;

PROC PRINT DATA=WORK.&CAR_TYPE._SUMM;
RUN;

%MEND;

/* PRFIX--> & AND SUFFIX--> . ===> THESE ARE USED TO MAKE THE PARAMETER DYNAMIC.
                              ===>  AND ALSO CALL ANY MACRO VARIABLE */
/* NOW WE CREATED THE MACRO BUT TO SHOW THE OUTPUT,WE WILL CALL THE MACRO WITH THE ARGUMENT. */

%TYPE_SUMM_AUTO(SUV);

%TYPE_SUMM_AUTO(TRUCK);



/* PART-2:--> */
/* MACRO VARIABLES: */
/* A MACRO VARIABLE IS USED TO STORE A VALUE WHICH ARE ALWAYS CHARACTERS
AND INCLUDES--> NAME, LETTERS, NUMBERS OR ANY TEXT */


/* HOW TO DEFINE A MACRO VARIABLE? */
/* %LET */

%LET FIRST_NUMBER=773;
%LET SECOND_NUMBER=876;

/* HOW TO SEE THE MACRO VARIABLE IS ASIGNED OR NOT? */
/* %PUT */

%PUT &FIRST_NUMBER.;

DATA TEST;
SUM=&FIRST_NUMBER.+ &SECOND_NUMBER.;
RUN;



/* TWO TYPES OF MACRO: */

/* 1. LOCAL MACRO: */
/* MACRO VARIABLES DEFINED INSIDE A MACRO.
 THESE CAN BE USED ONLY IN THE SAME MACRO WHERE IT HAS BEEN CREATED. */

%MACRO SAMPLE;              /* START THE MACRO */

%LET FIRST_NUMBER=70;          /* LOCAL MACRO VARIABLE1 */
%LET SECOND_NUMBER=30;         /* LOCAL MACRO VARIABLE2 */

DATA TEST1;
SUM=&FIRST_NUMBER.+ &SECOND_NUMBER.;
RUN;

PROC PRINT DATA= TEST1;
RUN;

%MEND;                          /* END THE MACRO */

%SAMPLE;                        /* CALL THE MACRO */


/* THESE MACRO VARIABLES ARE NOT USED ANY WHERE OUTSIDE THE MACRO. */



/* 2. GLOBAL MACRO: */
/* MACRO VARIABLES DEFINED OUTSIDE A MACRO.
  THESE CAN BE USED ANY WHERE IN THE SAS PROGRAM. */



%LET FIRST_NUMBER1=70;
%LET SECOND_NUMBER2=50;

%MACRO SAMPLE1;

DATA TEST2;
SUM=&FIRST_NUMBER1.+ &SECOND_NUMBER2.;
RUN;

PROC PRINT DATA= TEST2;
RUN;

%MEND;

%SAMPLE1;
```

```
/* THESE MACRO VARIABLES ARE USED ANY WHERE OUTSIDE THE MACRO. */

DATA TEST3;
SUM=&FIRST_NUMBER1.+ &SECOND_NUMBER2.;
RUN;

PROC PRINT DATA= TEST3;
RUN;




/* PART-3:--> */

/* DIFFERENT WAYS OF CREATING MACRO VARIABLES IN SAS: */
/* a. %LET */
/* b. MACRO PARAMETERS */
/* c. INTO CLAUSE IN PROC SQL */
/* d. CALL SYMPUT ROUTINE */



/* a. %LET: */
/* SYNTAX-->  %LET MACRO_VARIABLE_NAME = VALUE; */

%LET FIRST_NUMBER=773;
%LET SECOND_NUMBER=876;

DATA TEST;
SUM=&FIRST_NUMBER.+ &SECOND_NUMBER.;
RUN;



/* b. MACRO PARAMETERS: */
/* SYNTAX-->  */

/* %MACRO MACRO_NAME(INPUT=, IVAR=, OUTPUT=); */
/* ...STATEMENTS.. */
/* %MEND; */

%MACRO TYPE_SUMM_AUTO(CAR_TYPE);

PROC SQL;
CREATE TABLE &CAR_TYPE._SUMM AS
SELECT TYPE,
       ORIGIN,
       COUNT(*) AS NO_OF_UNITS,
       SUM(MSRP) AS TOTAL_MSRP,
       SUM(INVOICE) AS TOTAL_INVOICE
       FROM SASHELP.CARS
       WHERE UPCASE(TYPE)="&CAR_TYPE."
       GROUP BY 1,2;
QUIT;

PROC PRINT DATA=WORK.&CAR_TYPE._SUMM;
RUN;

%MEND;

%TYPE_SUMM_AUTO(SUV);


/* c. INTO CLAUSE IN PROC SQL: */

/* SYNTAX--> */
/* PROC SQL NOPRINT; */
/* SELECT VARIABLE_NAME INTO : MACRO_VARIABLE_NAME */
/* FROM TABLES; */
/* QUIT; */

PROC SQL NOPRINT;
```

```
SELECT NUMBER_OF_UNITS INTO : UNITS
FROM MYLIB.CARS_SUMM;
QUIT;

/* HOW TO PRINT ANY MACRO_VARIABLE */
%PUT &UNITS.;

DATA TESTL;
SET MYLIB.CARS;
WHERE NUMBER_OF_UNITS = &UNITS.;
RUN;



/* IF WE WANT TO MORE THAN ONE VALUES BE PRINTED */

/* IF YOU DO NOT WANT TO PRINT THEN USE 'NOPRINT'*/
PROC SQL NOPRINT;
SELECT NUMBER_OF_UNITS INTO : UNITS SEPARATED BY ","
FROM MYLIB.CARS_SUMM;
QUIT;

/* EG-1: */
DATA TESTL;
SET MYLIB.CARS;
WHERE NUMBER_OF_UNITS IN(94,11,10);
RUN;

/* EG-2 */
DATA TESTL;
SET MYLIB.CARS;
WHERE NUMBER_OF_UNITS IN(&UNITS.);
RUN;




/* d. CALL SYMPUT ROUTINE: */
/* SYNTAX--> */
/* CALL SYMPUT (MACRO_VARIABLE_NAME, VALUE); */

/* IF YOU DO NOT WANT TO PRINT THEN USE '_NULL_'*/
DATA _NULL_;
CALL SYMPUT ('UNITSS',94);
RUN;

%PUT &UNITSS.;

DATA TESTLL;
SET MYLIB.CARS;
WHERE NUMBER_OF_UNITS = &UNITSS.;
RUN;



DATA TET;
SET SASHELP.CARS;
RUN;

PROC SQL;
CREATE TABLE CAR_S AS
SELECT
    TYPE,
    ORIGIN,
    CYLINDERS,
    HORSEPOWER
    FROM SASHELP.CARS;

QUIT;
```

```
/* PART-4:--> */

/* IF , THEN DO: */

%MACRO AUTO_REPORT(CARS_TYPE);

%IF &CARS_TYPE. = "HYBRID" %THEN %DO;

PROC SQL;
    CREATE TABLE SPR_CATE AS
    SELECT TYPE,
            COUNT(*) AS NUMBER_OF_UNITS,
            AVG(CYLINDERS) AS AVG_NO_CYLINDERS
        FROM WORK.CAR_S
        WHERE UPCASE(TYPE) = &CARS_TYPE.
        GROUP BY 1;
QUIT;
PROC PRINT DATA=WORK.SPR_CATE;
RUN;
%END;


%ELSE %DO;
PROC SQL;
    CREATE TABLE NORMAL_CATE AS
    SELECT TYPE,
            COUNT(*) AS NUMBER_OF_UNITS,
            AVG(CYLINDERS) AS AVG_NO_CYLINDERS
        FROM WORK.CAR_S
        WHERE UPCASE(TYPE) NE &CARS_TYPE.
        GROUP BY 1;
QUIT;
PROC PRINT DATA=WORK.NORMAL_CATE;
RUN;
%END;

%MEND;

%AUTO_REPORT("SUV");

%AUTO_REPORT("HYBRID");






/* PART-5.1:---> */

/* MACRO FUNCTIONS IN SAS: */

/* %STR: */


/* THREE MAIN USAGES:- */

/* 1. HELPS IN PRINTS THE QUOTES WITH TEXTS BY PRECEDING A '%' SYMBOL. */

%LET WHAT = %STR(MITRA%'S BOOK);

%PUT &WHAT.;


/* 2. HELPS IN PRINTING THE SPECIAL CHARACTER LIKE : +,-,>,<,;,",LT,EQ,GT,LE,GE,LE,NE,AND,OR,NOT BLANK */

%LET X = %STR(PROC MEANS;RUN;) ;

%PUT &X.;
```

```
/* 3. HELPS IN RETAINING THE TRAILING AND LEADING BLANKS. */

%LET Y = %STR(   A   );

%PUT &Y.;




/* PART-5.2:---> */


/* OTHER MACRO FUNCTIONS IN SAS: */


/* %EVAL: */
/* HELPS IN PERFORMAING THE MATHEMATICAL AND LOGICAL OPERATIONS WITH MACRO VARIABLES. */

%LET a=10;
%LET b=3;

%LET Z=%EVAL(&a. * &b.);

%PUT &Z.;

/* %SYSFUNC: */
/* HELPS IN USING THE MANY OF THE USEFUL BASE SAS FUNCTIONS IN MACROS. */


%LET NAME= MITRABHANU;
%LET FIRST_NAME = %SYSFUNC(SUBSTR(&NAME.,1,5));
%LET FIRST_NAME_LENG = %SYSFUNC(LENGTH(&NAME.));

%PUT &FIRST_NAME. &FIRST_NAME_LENG.;



/* USAGE OF %: */

%LET NAME=MITRABHANU;

%LET FIRST_NAME_LENG = %LENGTH(&NAME.);
%LET FIRST_NAME = %SUBSTR(&NAME.,1,5);

%PUT &FIRST_NAME_LENG. &FIRST_NAME.;





/* PART-6.1:---> */

/* OPTIONS TO DEBUG SAS MACROS: */

/* a. MPRINT: */
/* IT SPECIFIES WHETHER SAS STATEMENTS GENERATED BY MACRO EXECUTION ARE TRACED FOR DEBUGGING. */
/* MPRINT TRANSLATES THE MACRO LANGUAGE TO REGULAR SAS LANGUAGE.
    IT DISPLAYS ALL THE SAS STATEMENTS OF RESOLVED MACRO CODE. */


/* EG-1: */

/* SS-1: */
---------------------------------------------------------------------------
%MACRO CHEC_FREQ(DATA,VAR,ORIGIN);

PROC FREQ DATA = &DATA.;
TITLE "THIS IS THE FRQUENCY OF &VAR. IN &ORIGIN.";
TABLES &VAR.;
WHERE ORIGIN="&ORIGIN.";
RUN;
```

```
%MEND;

%CHEC_FREQ(SASHELP.CARS,TYPE,USA)
%CHEC_FREQ(SASHELP.CARS,TYPE,Asia)



/* SS-2: */

%MACRO CHEC_FREQ(DATA,VAR,ORIGIN);

PROC FREQ DATA = &DATA.;
/* TITLE "THIS IS THE FRQUENCY OF &VAR. IN &ORIGIN."; */
TABLES &VAR.;
WHERE ORIGIN="&ORIGIN.";
RUN;

%MEND;

%CHEC_FREQ(SASHELP.CARS,TYPE,USA)
%CHEC_FREQ(SASHELP.CARS,TYPE,Asia);




/* FINAL: */
/* IT TELL ALL THE STEPS LINE BY LINE. */


OPTION MPRINT;
%MACRO CHEC_FREQ(DATA,VAR,ORIGIN);

PROC FREQ DATA = &DATA.;
/* TITLE "THIS IS THE FRQUENCY OF &VAR. IN &ORIGIN."; */
TABLES &VAR.;
WHERE ORIGIN="&ORIGIN.";
RUN;

%MEND;
```

```
1          OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
68
69         %CHEC_FREQ(SASHELP.CARS,TYPE,USA)
MPRINT(CHEC_FREQ):   PROC FREQ DATA = SASHELP.CARS;
MPRINT(CHEC_FREQ):   TABLES TYPE;
MPRINT(CHEC_FREQ):   WHERE ORIGIN="USA";
MPRINT(CHEC_FREQ):   RUN;

NOTE: There were 147 observations read from the data set SASHELP.CARS.
      WHERE ORIGIN='USA';
NOTE: PROCEDURE FREQ used (Total process time):
      real time           0.01 seconds
      user cpu time       0.01 seconds
      system cpu time     0.00 seconds
      memory              1474.65k
      OS Memory           20640.00k
      Timestamp           01/31/2024 08:26:31 AM
      Step Count                        29  Switch Count  3
      Page Faults                       0
      Page Reclaims                     758
      Page Swaps                        0
      Voluntary Context Switches        14
      Involuntary Context Switches      0
      Block Input Operations            0
      Block Output Operations           280


70         %CHEC_FREQ(SASHELP.CARS,TYPE,Asia)
MPRINT(CHEC_FREQ):   PROC FREQ DATA = SASHELP.CARS;
MPRINT(CHEC_FREQ):   TABLES TYPE;
MPRINT(CHEC_FREQ):   WHERE ORIGIN="Asia";
MPRINT(CHEC_FREQ):   RUN;

NOTE: There were 158 observations read from the data set SASHELP.CARS.
      WHERE ORIGIN='Asia';
NOTE: PROCEDURE FREQ used (Total process time):
      real time           0.01 seconds
      user cpu time       0.01 seconds
      system cpu time     0.00 seconds
      memory              733.03k
      OS Memory           21156.00k
      Timestamp           01/31/2024 08:26:31 AM
      Step Count                        30  Switch Count  3
      Page Faults                       0
      Page Reclaims                     121
      Page Swaps                        0
      Voluntary Context Switches        17
      Involuntary Context Switches      0
      Block Input Operations            0
      Block Output Operations           264


71
72         OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
82
```

```
/* PART-6.2:---> */

/* b. MLOGIC: */
/* IT CAUSES THE MACRO PROCESSOR TO TRACE ITS EXECUTION AND TO WRITE THE TRACE INFORMATION TO THE SAS LOG. */
/* IF MLOGIC IS IN EFFECT AND THE MACRO PROCESSOR ENCOUNTERS A MACRO INVOCATION,
        THE MACRO PROCESSOR DISPLAYS MESSAGES THAT IDENTIFY THE FOLLOWING:--> */

/*      --> THE BEGINING OF MACRO EXECUTION. */
/*      --> VALUES OF MACRO PARAMETERS AT INVOCATION. */
/*      --> EXECUTION OF EACH MACRO PROGRAM STATEMENT. */
/*      --> WHETHER EACH %IF CONDITION IS TRUE OR FALSE. */
/*      --> THE ENDING OF MACRO EXECUTION. */


/* EG-1: */


OPTION MLOGIC;
%MACRO TESTING(TYPE);

%IF %UPCASE(&TYPE.) = SUV %THEN %DO;
PROC FREQ DATA=SASHELP.CARS;
TABLES ORIGIN;
RUN;
%END;

%IF %UPCASE(&TYPE.) = SPORTS %THEN %DO;
PROC FREQ DATA=SASHELP.CARS;
TABLES CYLINDERS;
RUN;
%END;


%ELSE %DO;
PROC FREQ DATA=SASHELP.CARS;
TABLES MAKE;
RUN;
%END;

%MEND;

%TESTING(SPORTS);
```

```
1          OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
68
69         %TESTING(SPORTS);
MLOGIC(TESTING):  Beginning execution.
MLOGIC(TESTING):  Parameter TYPE has value SPORTS
MLOGIC(TESTING):  %IF condition %UPCASE(&TYPE.) = SUV is FALSE
MLOGIC(TESTING):  %IF condition %UPCASE(&TYPE.) = SPORTS is TRUE

NOTE: There were 428 observations read from the data set SASHELP.CARS.
NOTE: PROCEDURE FREQ used (Total process time):
      real time           0.01 seconds
      user cpu time       0.01 seconds
      system cpu time     0.00 seconds
      memory              1511.81k
      OS Memory           20896.00k
      Timestamp           01/31/2024 08:29:58 AM
      Step Count                        29  Switch Count  3
      Page Faults                       0
      Page Reclaims                     736
      Page Swaps                        0
      Voluntary Context Switches        19
      Involuntary Context Switches      0
      Block Input Operations            0
      Block Output Operations           272


MLOGIC(TESTING):  Ending execution.
70
71         OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
81
```

```
/* PART-6.3:---> */

/* c. SYMBOLGEN: */
/* IT DISPLAY THE RESULTS OF RESOLVING MACRO VARIABLE REFERENCES.
    THIS OPTION IS USEFUL FOR DUBUGGING. */

/* EG-1: */

OPTION SYMBOLGEN;
%MACRO CHECK_FRQ(TYPE, VAR);
PROC FREQ DATA=SASHELP.CARS;
TABLES &VAR.;
WHERE UPCASE(TYPE) = "&TYPE.";
RUN;
%MEND;

%CHECK_FRQ(SUV, ORIGIN);


/* NOTE:---> IF YOU ARE USING MACRO VARIABLE AS VALUE THEN ASIGN IT INTO DOUBLE QUOTES. */
```

```
1          OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
SYMBOLGEN:  Macro variable _SASWSTEMP_ resolves to /home/u63730693/.sasstudio/.images/67b3e709-f3e9-428e-84cd-9bd7a2b2d146
SYMBOLGEN:  Some characters in the above value which were subject to macro quoting have been unquoted for printing.
SYMBOLGEN:  Macro variable GRAPHINIT resolves to GOPTIONS RESET=ALL GSFNAME=_GSFNAME;
68
69         %CHECK_FRQ(SUV, ORIGIN);
SYMBOLGEN:  Macro variable VAR resolves to ORIGIN
SYMBOLGEN:  Macro variable TYPE resolves to SUV

NOTE: There were 60 observations read from the data set SASHELP.CARS.
      WHERE UPCASE(TYPE)='SUV';
NOTE: PROCEDURE FREQ used (Total process time):
      real time           0.01 seconds
      user cpu time       0.01 seconds
      system cpu time     0.00 seconds
      memory              1486.78k
      OS Memory           20896.00k
      Timestamp           01/31/2024 08:31:49 AM
      Step Count                        29  Switch Count  2
      Page Faults                       0
      Page Reclaims                     778
      Page Swaps                        0
      Voluntary Context Switches        10
      Involuntary Context Switches      0
      Block Input Operations            0
      Block Output Operations           272


70
71         OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
SYMBOLGEN:  Macro variable GRAPHTERM resolves to GOPTIONS NOACCESSIBLE;
81
```