SAS Data sets for Q1 to Q8: WORK.Employee,WORK.Manager

Q1. Executing the following SAS macro codes:

```
%let DA=Employee;
%let var1=emp_id;
%let var2=dep_id;
%let t=List of Employees;
%let sa=60000;
```

(1) Using the command %put to print the values of above macro variables in the SAS LOG WINDOW.
(2) What is the difference between outcomes of the following SAS codes? (a) %put t: "&t" (b) %put t: '&t'

Q2. Following Q1, compare and executing the following two SAS programs (SAS codes 1 and 2)
(1) Do they get the same results? Why?
(2) Which program is better? Why?
(3) If you want to print the different titles for tables 'Employee' and 'Manager', for example, 'List of Employees' and 'List of Managers', how do you improve the SAS macro codes 1?

```
/*******SAS codes 1*****************/
Proc print data=&DA;
  Var &var1 &var2 salary;
  Where salary>&sa;
  Title "&t";
Run;
%let DA=Manager;
Proc print data=&DA ;
  Var &var1 &var2 salary;
  Where salary>&sa;
  Title "&t";
Run;
```

```
/*******SAS codes 2*****************/
Proc print data=Employee;
  Var emp_id dep_id salary;
  Where salary>60000;
Title "List of Employees";
Run;
Proc print data=Manager;
  Var emp_id dep_id salary;
  Where salary>60000;
  Title "List of Employees";
Run;
```

(1) Do they get the same results? Why?
(2) Which program is better? Why?

Q3. If you have defined the following macro variables by running the following SAS codes:

```
%let memid=E12;
%let mdepid=D1;
```

Now for the employee emp_id='E12' and 'dep_id='D1', you need to print (use %put) the value of variable 'tenure' in the table 'WORK.Employee' and the value of variable 'start_since' in the table 'WORK.Manager'. Use 'Call Symput' to realize this.

Q4. If you have defined the following macro variables by running the following SAS codes:

```
%let mt=1;
%let ms=0.03;
%let mo=employee;
%let newt=newemployee;
```

(1) Now you need to create a new SAS table work.newemployee (using above macro variable 'newt'). The contents of the data set are first copied from the table 'WORK.Employee' and then the variable 'tenure' is added 1 (year) (using above macro variable 'mt'), the variable 'salary' is added 3% (using above macro variable 'ms').
(2) Defining a macro program `%macro getnew(a,b,c,d).` Where the parameters are matched with the macro variables 'mt',ms','mo' and 'newt' correspondingly. Finally run the program by

```
%getnew(&mt.,&ms.,&mo.,&newt.);
```

Q5. You need to generate 5 SAS data sets: 'WORK.employee_1', 'WORK.newemployee_2',.. 'WORK.newemployee_5'. Where the table 'WORK.newemployee_1' is copied from 'WORK.employee'. Then the variable 'tenure' is added 1 (year), and the variable 'salary' is added 3%. Following this procedure, the table'WORK.newemployee_2' is copied from ''WORK.newemployee_1', and then the variable 'tenure' is added 1 (year), and the variable 'salary' is added 3%.....Write a SAS macro program to realize the function described above with the form:

```
%macro createntables(n,salary_incr_per);
```

Then run macro using `%createntables(5,0.03)`

Q6. Following Q5, create a macro program:

```
%macro changebyyear(startyear,endyear,out);
```

Where the program is explained by the following example; if you run the program using the following SAS codes:

```
%changebyyear(2,4,diff24);
```

Then a new SAS table 'diff24' is created by merging the tables WORK.newemployee_2' and WORK.newemployee_4' in terms of the variable 'emp_id'. This resulting table contains three columns (a) emp_id (2) salaeary_increase (3) tenure increase.

Q7. Following Q6, create a macro program:

```
%macro printdata(D);
```

Where the program is explained by the following example; if you run the program using the following SAS codes:

```
%printdata(diff24);
```

Then the table 'diff24' is printed with the title 'difference after 2 years' (Note, you have no information for 'startyear','endyear' in the parameter list, so try to extract them from the parameter 'D').

Q8. Following Q5, assuming you already generated the data set 'newemployee_1' to 'newemployee_5'. Now you want to calculate the net income by adjusting the variable 'salary' in each table. The following SAS data set 'WORK. Adj_rate' can be used for the calculation:

	year	tax_rate	bonus_rate
1	1	0.23	0.11
2	2	0.25	0.13
3	3	0.25	0.1
4	4	0.27	0.12
5	5	0.3	0.15

For instance, the net income of the table 'newemployee_1' can be calculated in this way: 'income=salary*(1-0.23+0.11);' and the net income of the table 'newemployee_2' can be calculated in this way: 'income=salary*(1-0.25+0.13);'… Write a SAS macro program to realize this:

```
%macro adjust(adjtable);

%adjust(Adj_rate);
```

You then use the macro above to calculate (and obtain) the net income of each table. Finally print out all adjusted tables using the title 'Table 1: Total Rate Adjusted: 0.12' (e.g. 0.23-0.11=0.12 for the table 'newemployee_1').

SAS Data sets for Q1 to Q8: 'WORK.Sas_code'and 'WORK.Coffee_new'

Q9. Checking the data set 'work.Sas_code' and 'work.Coffee_new' and define the following SAS macro program:

```
%macro createnew(agelist, inputd, outputd);
```

Where the program is explained by the following example; if you run the program using the following SAS codes:

```
%let agelist=%str(30,40);
%let inputd=coffee_new;
%let outputd=coffee_result;
%createnew(&agelist., &inputd., &outputd.);
```

Then the program will be doing the following jobs:

(a) Extracting the SAS codes stored in the column 'code' of data set 'work.Sas_code' (check the table) i.e. send

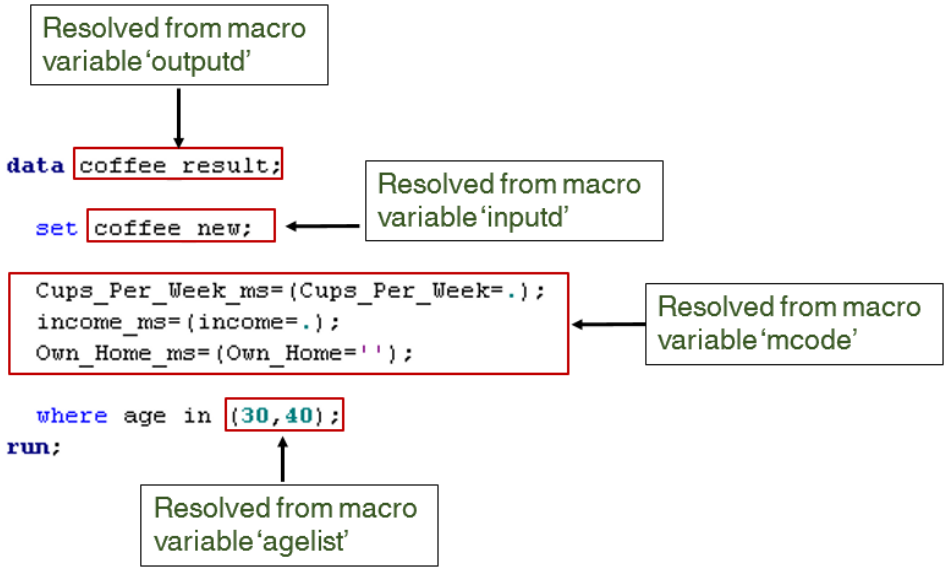
```
Cups_Per_Week_ms=(Cups_Per_Week=.);

income_ms=(income=.);

Own_Home_ms=(Own_Home='');
```

Into a macro variable 'mcode' (hint: use 'PROC SQL')

(b) Applying the macro variable 'mcode' created above and example's macro variable parameters 'agelist', 'inputd' and 'outputd', you should get the following resolved SAS codes:



Please explain why the mask function '%str' (for %str(30,40)) is used here.