ASSESSMENT 5

QUESTION 1

Merge Multiple Data Sets

Import Credit Ratings

The file CREDRATING.DAT contains the credit ratings of the customers, previously imported from CREDIT.DAT.

The data are separated by a space and organised as follows:

1. ID Number
2. Rating

Self Assessment Question

1. Import these data into a SAS data set called RATINGS, using the most appropriate method.
2. Perform appropriate validation on the imported variables.
3. Convert the text of the ratings categories to suitably chosen numerical values.
4. Create an output format for the numerical variable, which will detect erroneous and missing values and apply the format to the variable.
5. Revalidate the variable.
6. Drop any unnecessary variables.

When you have finished the task, paste your code into the box below.

ANSWER 1

```
/* 1.)importing the SAS data set called RATINGS*/
filename pwd 'C:\Users\Folashikemi\OneDrive - De Montfort
University\P2586104 (vfiler1.lec-admin.dmu.ac.ukHome4)\Documents\IMAT5168
Analytical Programming Mark\Lab 5';
data RATINGS;
            infile pwd (psa05-credrating.dat) missover
;
            input
                        ID_Number
                        Rating $
;
            label
                        ID_Number = 'ID Number'
                        Rating =    'Ratings'
;
run;
ods exclude enginehost;
/* 2.)Perform appropriate validation on the imported variables*/
proc contents
      data=RATINGS
      varnum
      ;
run;
ods select all;
proc freq data =RATINGS;
table _all_; /*_all_ is for all variables*/
run;
```

```
proc means data = RATINGS n nmiss min max;
var   ID_Number
;
run;
proc print data =RATINGS
label
noobs;
run;


data RATINGS;
set RATINGS;
/*3.)converting the text values to numerical values using if statement.
This will create a new variable called Rate which will validate missing
values or error in the raw data*/
if Rating = 'bad' then rate = 0;
if Rating = 'good' then rate = 1;
/* to check for missing values or error in the data*/
if Rating = ''    then rate = .;
run;
/*observe a new variable rate, was created*/
proc print data =RATINGS
label
noobs;
run;

/*4.)set output format which will detect erroneous and missing values*/
proc format;
      value
            rate
                  0='bad'
                  1='good'
                  other =.
;
run;
data RATINGS;
set RATINGS;
format
rate rate.;
run;

proc print data = RATINGS;
format rate;
run;
 ods exclude enginehost;
 /*5.)revalidate the data */
proc contents
      data=RATINGS
      varnum
      ;
run;
ods select all;
proc freq data =RATINGS;
      table _all_;
run;

proc means data = RATINGS n nmiss min max;
var   ID_Number

;
run;
```

```
/* difficulty understanding proc compare/but its meant to confirm that the
values in rate is the same as ratings*/
proc compare
 base = RATINGS
 compare = RATINGS;
 var rate rating;
run;
proc print data =RATINGS
label
noobs;
run;
/* 6.) Drop unneccessary variables*/
data RATINGS;
set RATINGS;
drop Rating;
rename rate = Rating;
run;
proc print data =RATINGS
label
noobs;
run;
```

QUESTION 2

Merge Multiple Data Sets

Import Date of Rating

The file CREDDATES.CSV contains a record of dates when the credit ratings of customers, previously imported from CREDIT.DAT were assessed.

Self Assessment Question

1. Import these data into a SAS data set called DATES.
2. Perform appropriate validation on each of the imported variables.
3. Convert the dates into SAS dates, using a suitable function.
4. Drop the variables that are no longer required.

When you have completed the task, copy your code into the box below

ANSWER 2

```
/* 1.)Importing CSV data set called DATES*/
filename pwd 'C:\Users\Folashikemi\OneDrive - De Montfort
University\P2586104 (vfiler1.lec-admin.dmu.ac.ukHome4)\Documents\IMAT5168
Analytical Programming Mark\Lab 5';
data DATES;
      infile
            pwd (psa05-creddates.csv) dsd firstobs=2 missover;
      input
                  ID_Number
                  day_of_rating
                  month_of_rating
                  year_of_rating
;
      label
                  ID_Number = 'ID Number'
```

```
                    day_of_rating = 'Day of Rating'
                    month_of_rating = 'Month of Rating'
                    year_of_rating = 'Month of Rating'
;
run;
      ods exclude enginehost;
/*2.)validating the imported variables*/
proc contents data = DATES
      varnum;
run;
      ods select all;
proc freq data = DATES;
      table
                    day_of_rating
                    month_of_rating
                    year_of_rating
;
run;

proc means data= DATES n nmiss min max;
      var   ID_Number
;
run;
/*3.) convert date to SAS DATE using CATX function.
      Note: the suggested function in the link provided won't work with
large data sets*/
data CR_DATES (keep= ID_Number Ratings_date);
 set DATES;
      Ratings_date = input
(catx('/',day_of_rating,month_of_rating,year_of_rating), ddmmyy10.);
 informat
      Ratings_date ddmmyy10.;
format
      Ratings_date ddmmyy10.;
run;

proc print
data = CR_DATES
label
;
run;
/* 4.) no unneccassary variable to drop*/
```

QUESTION 3

Merge Multiple Data Sets

Previously you wrote code to import CREDIT.DAT into a SAS Data Set. If you have not done so already, complete this task and name the data set: CUSTOMERS.

Self Assessment Question

1. Merge the data sets: CUSTOMERS, RATINGS and DATES into a new data set called: CREDIT, sorting the source data where appropriate.
2. Label the new variables appropriately.
3. Set output formats where appropriate.

When you have completed the task, paste all the code you used in the box below.

Hint: Read about the data step's merge statement as well as the sort procedure.

ANSWER 3

```
/* 1.)sort all data sets by ID_Number using proc sort*/
proc sort data=CR_RATINGS;
              by ID_Number;
run;
proc sort data=CR_DATES;
              by ID_Number;
run;
proc sort data=CUSTOMER;
              by ID_Number;
run;
/*3.) set out put format for Rating*/
proc format;
      invalue $Rating
                  'good'=1
                  'bad'=0
                  other = .
;
run;
/* 1.) Merge CUSTOMER CR_RATINGS CR_DATE*/
data CREDIT;
merge CUSTOMER CR_RATINGS CR_DATES;
by ID_Number ;
label Ratings_date='Ratings Date'; /*2.)Set new Label for Ratings Date*/
informat Rating $Rating.;
run;

proc print data=CREDIT label;
format Rating;
run;
```

QUESTION 4

Merge Multiple Data Sets

Validation

There are many ways of merging data sets.

Self Assessment Question

Describe and implement a strategy using data steps and the validation procedures you have used so far, for checking that the merged data sets are correct.

Hint: You should consider the following questions:

1. What is the significance of the ID number?
2. What is a correct result?
3. What new data sets might help compare the merged and the original data?

4. How might you combine these data sets so that you can compare variables for the same customer to check if they are equal?

When you have completed the task, paste the code into the box below.

ANSWER 4

```
/*created a new data set to for mismatched record in one-to-one merge
Note: In the output only CR_DATES have '0' because to represent  missing
values*/
data MCREDIT /*Merged Credit = MCREDIT*/;
set CREDIT;
merge CUSTOMER(in=CUSTOMERX) CR_RATINGS(in=CR_RATINGSX)
CR_DATES(in=CR_DATESX);
by ID_Number;
fromCUSTOMER = CUSTOMERX;
fromCR_RATINGS = CR_RATINGSX;
fromCR_DATES = CR_DATESX;
run;
/* Cross-Tabular Frequency Table*/
proc freq data = MCREDIT;
tables fromCUSTOMER*fromCR_RATINGS*fromCR_DATES;
run;

proc print data=MCREDIT;
run;
```

QUESTION 5

Merge Multiple Data Sets

Validation

Is there a SAS procedure that performs this function?

Self Assessment Question

Compare using the SAS procedure with performing the task using the data step and validation procedures.

Hint: perform the same task as the previous question using the SAS procedure you found. What are the advantages and disadvantages compared with using the data step.

ANSWER 5

1. Multiple datasets. The DATA Step is a true workhorse. It allows you to create multiple datasets in one swift step. PROC SQL requires several SELECT clauses to create multiple datasets.
2. Reading text. The ability of SAS to read virtually any data (text files, etc.) is outstanding. Use the DATA Step when you want to read text files. PROC SQL cannot read text files.
3. Control. The SQL optimizer has a mind of its own. For example if you want to join three tables, small, medium and large, it might consider that the large table is indexed so it easily joins it with the small table and then takes the intermediate table and joins with the mid-sized table. If you like control over the how the join is done, then go to the DATA Step.
PROC SQL
4. Joining tables. PROC SQL is flexible while joining multiple tables that don't have key variables in common. The data step requires you to have common named key columns before you attempt a merge.
5. Querying dictionary tables. When you query a dictionary table, SAS launches a discovery process which can end up searching libraries, open tables and execute views. It is often more efficient to use PROC SQL which optimizes the query before the discovery process is launched.