```
/*******************\
/*Lesson 3 ***********|
/*******************/
/* Exploring data */
/*1*/
/*
   proc contents used to view the description of the data
   SAS has several proceduces that can help you quickly
   and easily explore your data:
   1. print     --> creates list of all rows and columns
   2. Means     --> calculates simple summary od the statistics
                    for numeric columns in the data:
                      - frequency count
                      - mean
                      - standard deviation
                      - minmum
                      - maximum
   3. univariatr --> generates summmary statistics more detail
                     statistics related to distribution:
                      - 5 lowest and highest extreme values and
                        observation numbers
   4. FREQ       --> creates freqeuncy table for each colunm in
                     the input table, the result content a unique
                     value for each table:
                       - frequency
                       - percent
                       - cumulative frequency
                       - cumulative percent

*/
/*****************************************************************/

/*demo*/
libname pg1 base "/home/u64168505/EPG1V2/data";
/*1*/
/* print first 10 observation */
proc print data=PG1.STORM_SUMMARY (obs=10);
run;
/*2*/
/* print first 10 observation in a specific columns */
proc print data=PG1.STORM_SUMMARY (obs=10);
     var Season Name Basin MaxWindMPH MinPressure StartDate EndDate;
run;
/*3*/
/* calculate summary statistic only used in numeric columns */
/* the different between the the frequency count between this two
   column indicate the missing values*/
proc means data=PG1.STORM_SUMMARY;
     var MaxWindMPH MinPressure;
run;
/*4*/
/* examine extreme values */
proc univariate data=PG1.STORM_SUMMARY;
     var MaxWindMPH MinPressure;
```

```
run;
/*5*/
/* list unique values and frequencies */
proc freq data=PG1.STORM_SUMMARY;
     tables Basin Type Season;
run;
/*****************************************************************/

/*practices*/
libname pg1 base "/home/u64168505/EPG1V2/data";

/*1*/
/* list first 20 rows */
proc print data=pg1.np_summary (obs=20);
run;

/*2*/
/* list first 20 rows of specific columns*/
proc print data=pg1.np_summary (obs=20);
    var Reg Type ParkName DayVisits TentCampers RVCampers;
run;

      /* Do you observe any possible inconsistencies in the data?*/
       * yes, in the tupe column there are PRE and PRESERVE;

/*3*/
/* calculate summary statistics */
proc means data=pg1.np_summary;
    var DayVisits TentCampers RVCampers;
run;

       /* What is the minimum value for tent campers?
          Is that value unexpected?*/
        * 0 value is not unexpected,
          it's possible that a park had zero tent campers;

/*4*/
/* examine extreme values */
proc univariate data=pg1.np_summary;
    var DayVisits TentCampers RVCampers;
run;

        /* Are there negative values for any of the columns?
         * No;

/*5*/
/* list unique values and frequency counts */
proc freq data=pg1.np_summary;
    tables Reg Type;
run;

       /* Are there any lowercase codes?*/
        * There are no lowercase codes;
       /* Are there any codes that occur only once in the table?*/
        * NC, NPRE, and RIVERWAYS occur once in the table;
```

```
/*************************************************************************/


/* Filtering Rows */
/*2*/
/*
   Making a condition using where.
   Where statement can be used:
    - proc print
    - proc means
    - proc freq
    - proc unvariate
    - ...

   Where followed by one or more expression.
   expression test the value of a column against a condition

   Basic Operators:                                          *;
         = or EQ                                              *;
         ^= or ~= or NE                                        *;
         > or GT                                              *;
         < or LT                                              *;
         >= or GE                                             *;
         <= or LE

   character is in quotation column="..."
   special format:
      - "ddmmmyyyy"d
      - "01JAN2015"d

   combing different formats:
     - column="..." and column="..."
     - column="..." or column="..."
     - column="..." in (..,...,...)
     - column="..." not in (..,...,...)
     - column="..." between ... and ...
     - column LIKE "..." --> % for any number of characters
                         --> _ is for single character

    for missing values:
    where column=. or column=" "
    other option:
     - where column is missing
     - where column is not missing
     - where column is null

*/
/*************************************************************************/

/*demo*/
/*1*/
PROC PRINT PG1.STORM_SUMMARY;

/*2*/
proc print data= PG1.STORM_SUMMARY ;
```

```
    where MaxWindMPH >= 156;
run;

/*3*/
*a;
proc print data= PG1.STORM_SUMMARY ;
    where Basin='WP' ;
run;
*b;
proc print data= PG1.STORM_SUMMARY ;
    where Basin in ('SI','NI') ;
run;
*c;
proc print data= PG1.STORM_SUMMARY ;
    where StartDate >= "01jan2010"d;
run;
*d;
proc print data= PG1.STORM_SUMMARY ;
    where Type = 'TS' and Hem_EW ='W' ;
run;
*e;
proc print data= PG1.STORM_SUMMARY ;
    where MaxWindMPH > 156 or MinPressure <920 ;
run;

/*4*/
/*e*/
/*
In the final WHERE statement, are missing values included for MinPressure?
How can you exclude missing values?
** Yes, the null values is treated like the smallest possible value
** by adding that
           0 < MinPressure <920
*/
/*************************************************************************/

/*activity*/

/*1*/
proc print data=pg1.storm_summary(obs=50);
    where MinPressure is missing;
run;

proc print data=pg1.storm_summary(obs=50);
    where MinPressure = .;
run;

proc print data=pg1.storm_summary(obs=50);
    where Type is not missing;
run;

proc print data=pg1.storm_summary(obs=50);
     where Type ne " ";
run;
```

```sas
proc print data=pg1.storm_summary(obs=50);
    where MaxWindMPH between 150 and 155;
run;

proc print data=pg1.storm_summary(obs=50);
    where Basin like "_I";
run;
/*2*/
proc print data=pg1.storm_summary(obs=50);
   where name like "Z%";
run;
/* How many storms are included in the results?
*24 observation;
*/
/*****************************************************************/

/* Macro variable */
/*3*/
/*
   A macro variable store a test string
   The macro variable is design to make your programs reusable and dynamic.

    Syntax:
            %LET macrovar=value;
             Usage:
             WHERE numvar=&macrovar;
             WHERE charvar="&macrovar";
             WHERE datevar="&macrovar"d;

*/
/*****************************************************************/

/*demo*/
/*1*/
proc print data=pg1.storm_summary;
    where MaxWindMPH>=156 and Basin="NA" and StartDate>="01JAN2000"d;
    var Basin Name StartDate EndDate MaxWindMPH;
run;

proc means data=pg1.storm_summary;
    where MaxWindMPH>=156 and Basin="NA" and StartDate>="01JAN2000"d;
    var MaxWindMPH MinPressure;
run;
/*2 - 3*/
%let WindSpeed=156;
%let BasinCode=NA;
%let Date=01JAN2000;

proc print data=pg1.storm_summary;
    where MaxWindMPH>=&WindSpeed and Basin="&BasinCode" and StartDate>="&Date"d;
    var Basin Name StartDate EndDate MaxWindMPH;
run;

proc means data=pg1.storm_summary;
    where MaxWindMPH>=&WindSpeed and Basin="&BasinCode" and StartDate>="&Date"d;
```

```
    var MaxWindMPH MinPressure;
run;
/*4*/
%let WindSpeed=100;
%let BasinCode=SP;
%let Date=01JAN2010;

proc print data=pg1.storm_summary;
    where MaxWindMPH>=&WindSpeed and Basin="&BasinCode" and StartDate>="&Date"d;
    var Basin Name StartDate EndDate MaxWindMPH;
run;

proc means data=pg1.storm_summary;
    where MaxWindMPH>=&WindSpeed and Basin="&BasinCode" and StartDate>="&Date"d;
    var MaxWindMPH MinPressure;
run;
/*************************************************************************/

/*activity*/
/*1*/
%let BasinCode=SP;

proc means data=pg1.storm_summary;
    where Basin="&BasinCode";
    var MaxWindMPH MinPressure;
run;

proc freq data=pg1.storm_summary;
    where Basin='&BasinCode';
    tables Type;
run;
/*2*/
/* Which procedure did not produce a report?
* proc freq did not produce;
* NOTE: No observations were selected from data set PG1.STORM_SUMMARY;
/* What is different about the WHERE statement in that step?
* The difference in the WHERE statement in that step is that single quotation
  marks were used around the macro variable &BasinCode
  rather than double quotation marks.
  Double quotation marks must be used around macro variables.;
/*************************************************************************/

/*practice*/
/*1*/
proc print data=pg1.np_summary;
    var Type ParkName;
     WHERE ParkName like'%Preserve%';
run;
/*2*/
/*Which codes are used for Preserves?
* pre, preserve and npre;
*/
/*************************************************************************/

/*formatting column*/
```

```
/*4*/
/*
  To control how values appear in your reports
  - format column formattoapply
    formattoapply:
    <$> --> character
    <W> --> total width including decimals and special characters
     .  --> required delimiter
    <d> --> number of decimal places

    Common formats:

     - dollar10.2 -> $12,345.67
     - dollar10.  -> $12,346
     - comma8.1   -> 9,876.5
     - date7.     -> 01JAN17
     - date9.     -> 01JAN2017
     - mmddyy10.  -> 12/31/2017
     - ddmmyy8.   -> 31/12/17
     - monname.   -> january
     - weekdate.  -> Monday,January15,2018

*/
/*****************************************************************************/

/*demo*/
/*1*/
PROC PRINT data=PG1.STORM_DAMAGE;
run;
/*2*/
proc print data=pg1.storm_damage;
    format Date mmddyy8. cost dollar14. ;
run;

proc print data=pg1.storm_damage;
    format Date mmddyy10. cost dollar16. ;
run;
/*What happens to the formatted values?
from 08/25/05     and  $161300000000
to   08/25/2005   and  $161,300,000,000

/*****************************************************************************/

/*activity*/
/*1*/
proc print data=pg1.storm_summary(obs=20);
    format Lat Lon 4. StartDate EndDate date9.;
run;
/*2*/
proc print data=pg1.storm_summary(obs=20);
    format Lat Lon 4. StartDate EndDate date7.;
run;
/* How does the display of StartDate and EndDate change?
 * DATE7. displays a 2-digit year
*/
```

```
/*3*/
proc print data=pg1.storm_summary(obs=20);
    format Lat Lon 4. StartDate EndDate date11.;
run;
/* How does the display of StartDate and EndDate change?
 * DATE11. displays a 4-digit year and adds dashes
*/
/*4*/
proc freq data=pg1.storm_summary order=freq;
    tables StartDate;
run;
/*5*/
proc freq data=pg1.storm_summary order=freq;
    tables StartDate;
    FORMAT StartDate monname.;
run;
/*************************************************************************************/

/*Sorting Data and Remove duplicates*/
/*5*/
/* sorting the data is helpful
    - to improve visual arrangement of the data
    - to examine the high or low values
    - to identify and remove duplicate rows
    - to prepare data for certain data processing steps

   proc sort --> to sort one or more character or numeric colmns
   DATA=input-table
   <OUT=output-table>; --> optional if not the input table order will change
    BY <DESCENDING> col-name (s); --> specify one or more columns to be sort row
    default --> ascending order

   Remove duplicate:

     PROC SORT DATA=input-table <OUT=output-table>
         NODUPKEY <DUPOUT=output-table>;
         BY _ALL_;          ----->for duplicated row
         BY <DESCENDING> col-name (s);    ------->for duplicated value
     RUN;
     NODUPKEY --> to keep only the first one


*/
/*************************************************************************************/

/*activity*/
proc sort data=pg1.storm_summary out=pg1.STORM_SORT;
    where basin in ("NA" "na") ;
    by descending MaxWindMPH;
run;

/*************************************************************************************/

/*demo*/
*Step 1;
```

```
proc sort data=pg1.storm_detail out=storm_clean
    nodupkey dupout=STORM_DUPS;
    by _all_ ;
run;

*Step 2;
proc sort data=pg1.storm_detail out=min_pressure;
    where Pressure is not missing and Name is not missing;
    by descending Season Basin Name Pressure;
run;

*Step 3;
proc sort data=min_pressure nodupkey;
    by descending Season Basin Name  ;
run;
/*************************************************************************************/

/*activity*/

proc sort data= pg1.np_summary out=np_sort;
    where Type="NP";
    by Reg descending DayVisits;
run;

/*************************************************************************************/
```