

# HEC MONTRÉAL

## Statistical Analysis and Inference

SAS Laboratory

INTRODUCTION TO SAS

# SAS Lab: outline

- 1) save all the necessary files from ZoneCours into a work directory.
- 2) SAS software presentation through examples using the SAS code file `MATH60619A_SAS_intro.sas`.
- 3) Exercises :
  - The PDF document `MATH60619A_SASexercises` contains exercises to be completed throughout the lab session.
  - The files `elnino.sas7bdat` and `aapl.csv` contain the data for the exercises.
  - Solutions for the exercises will be posted after the lab session.

# Introduction to SAS

- Part I : Familiarisation with SAS and management of data files
- Part II : The DATA and PROC statements
- Part III : ODS
- Part IV : Graphs with SAS
- Part V : Consulting the SAS help

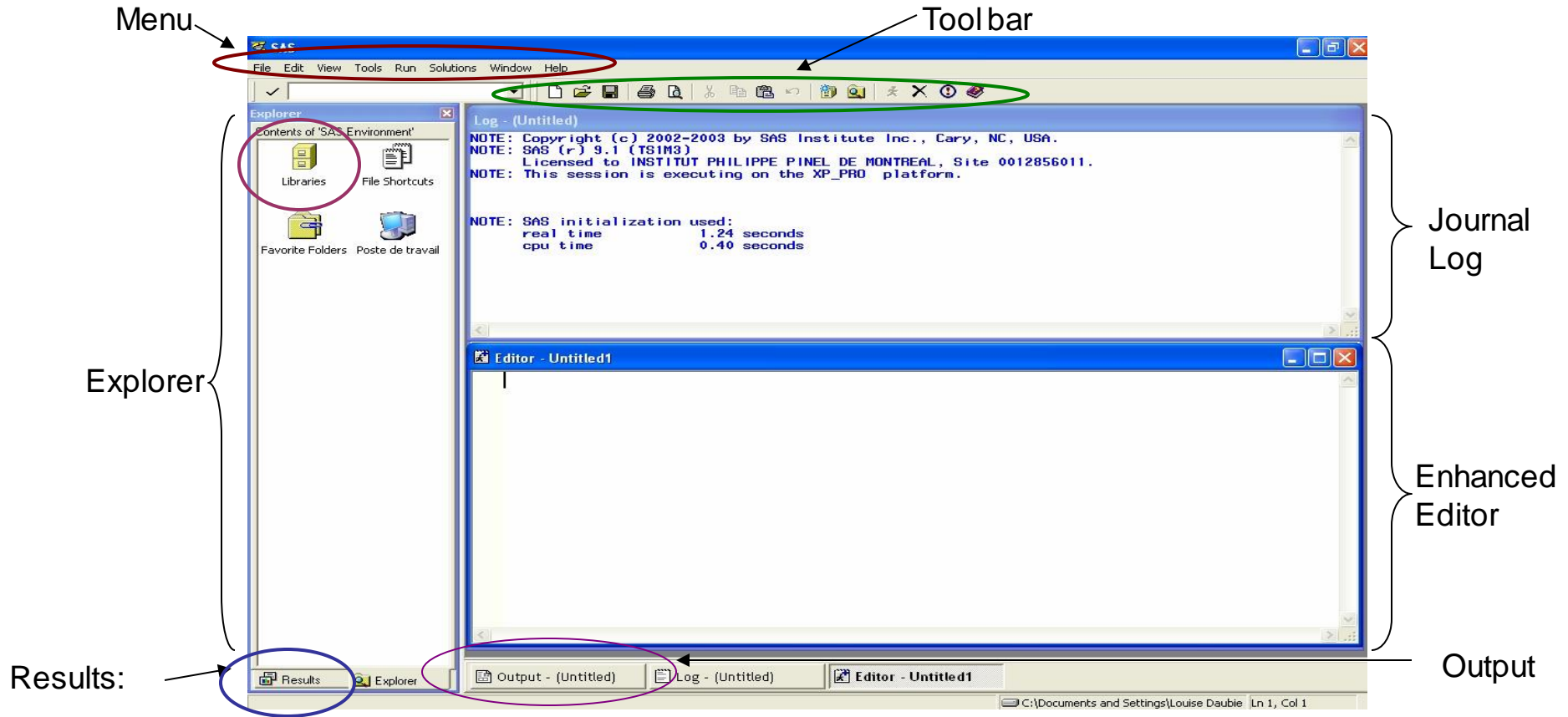
# Familiarisation with SAS

## Part I



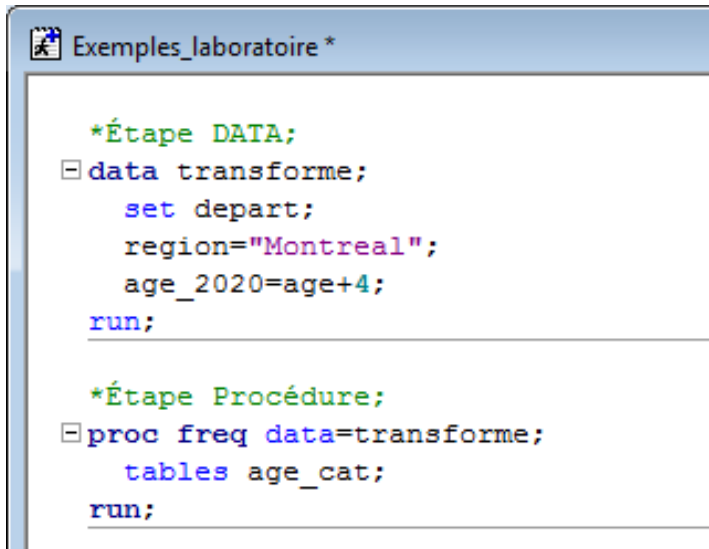
HEC MONTRÉAL

# Overview



# Editor

Window in which we type the code / program to be executed in SAS



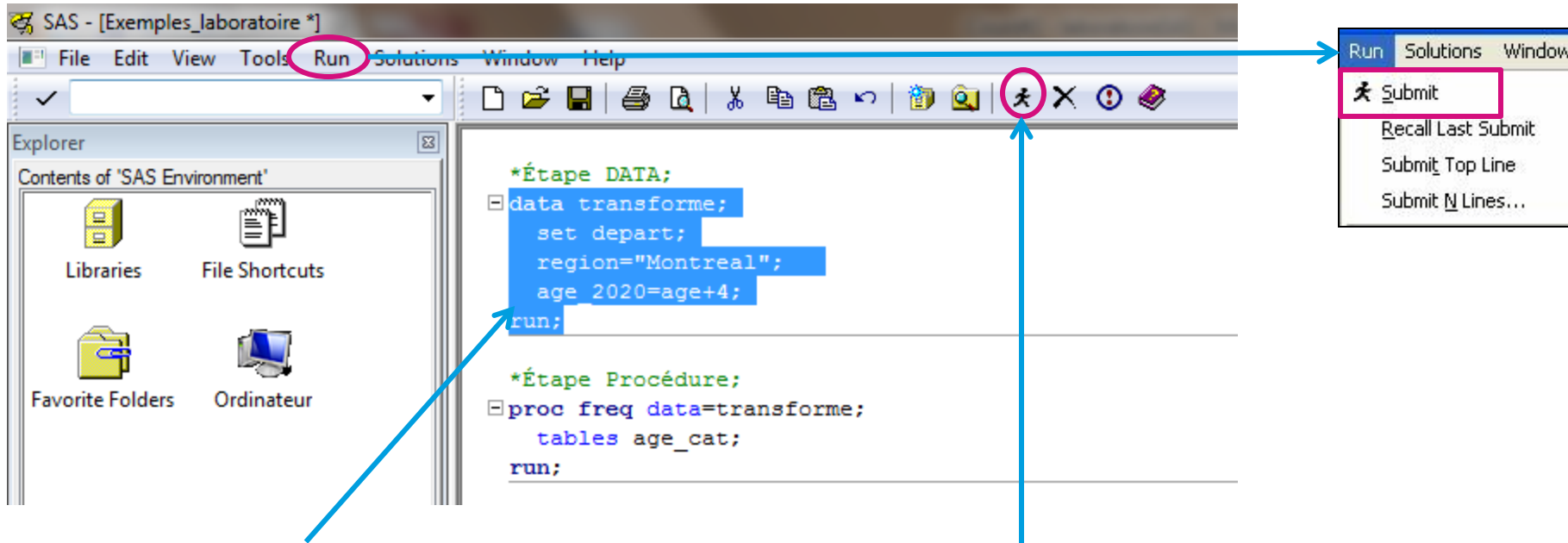
```
*Étape DATA;  
data transforme;  
    set depart;  
    region="Montreal";  
    age_2020=age+4;  
run;  
  
*Étape Procédure;  
proc freq data=transforme;  
    tables age_cat;  
run;
```

Colors : provide helpful indications

- **Green**: comments
- **Red**: incorrect
- **Blue (bold)**: key word marking the beginning and end of an instruction
- **Blue** : key word within an instruction
- **Purple**: chain of characters (must be between quotations marks or apostrophes)
- **Turquoise**: numbers
- **Black**: variables, tables, libraries,...

# Editor

## Executing a program



1. If we don't want to run the entire program in the editor, simply select the portion of code you want to execute.

2. Click on the «Submit» button (or use the shortcut Fn+F3)

HEC MONTRÉAL

# SAS syntax – generalities

In SAS, there are two main types of instructions:

## DATA commands

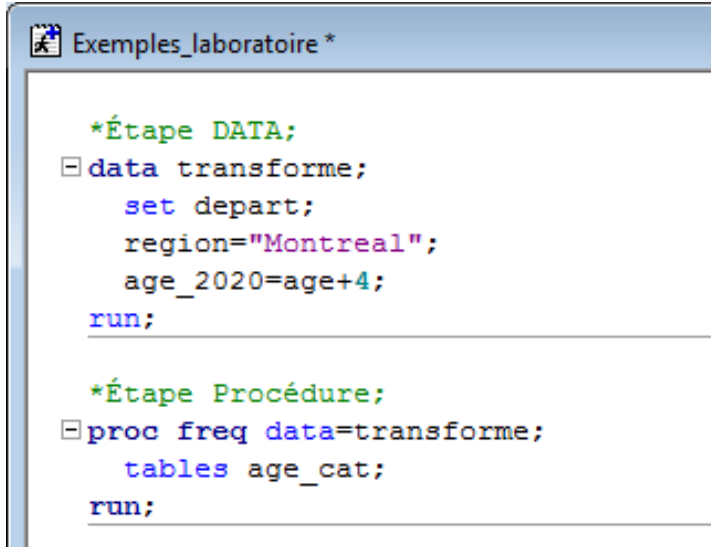
- Allows to manipulate data
- Ex: create tables, create/modify/delete variables, merge tables, etc.

## PROC (procedures) commands

- Allows to process/handle data
  - Ex: produce descriptive statistics, print the data, fit statistical models, etc.
- 
- There are also other types of global instructions that allow to define the options that will remain active for all steps in a program (ex: `OPTIONS` and `TITLE`).



# SAS instructions - generalities



```
*Étape DATA;  
data transforme;  
    set depart;  
    region="Montreal";  
    age_2020=age+4;  
run;  
  
*Étape Procédure;  
proc freq data=transforme;  
    tables age_cat;  
run;
```

- A SAS command starts with either a DATA or PROC instruction.
- A SAS command ends with a RUN instruction.
- A SAS command consists of several statements, each of which starts with a keywords (ex: SET, VAR, etc.) and ends with a semi-colon ;

- SAS does not distinguish between upper and lower case letters.
- Character chains must be encompassed by quotations or apostrophes.
- Comments start with /\* and end with \*/  
*or start with \* and end with a semi-colon ;*

# Log (Journal)

Contains history of executed code

3 types of messages :

- Note

```
NOTE: The data set WORK.TABLE1 has 3 observations and 3 variables.  
NOTE: DATA statement used (Total process time):  
      real time           0.01 seconds  
      cpu time            0.00 seconds
```

- Warning

```
WARNING: The data set WORK.TABLE may be incomplete. When this step was stopped there were 0  
         observations and 2 variables.  
WARNING: Data set WORK.TABLE was not replaced because this step was stopped.
```

- Error (the program was interrupted due to an error)

```
76  
ERROR 85-322: Expecting a format name.  
ERROR 76-322: Syntax error, statement will be ignored.
```

# Output (Results)

Contains the results from the executed code

The SAS System

The FREQ Procedure

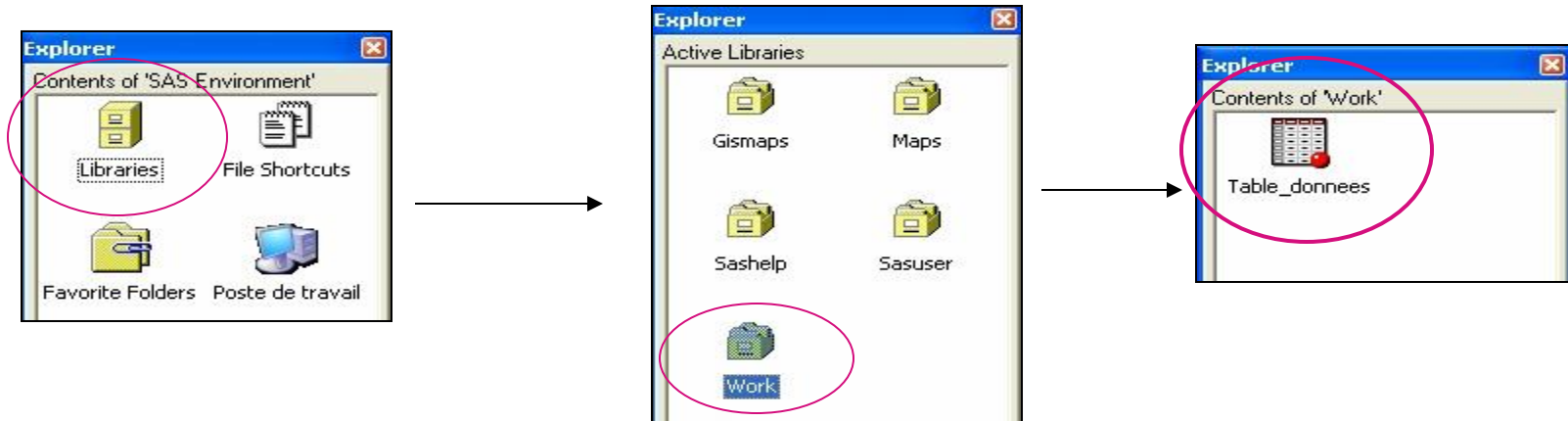
age_cat	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	1	25.00	1	25.00
2	2	50.00	3	75.00
3	1	25.00	4	100.00

Terminé

C:\Users\Julie Meloche

# Libraries

In the Explorer window



## Libraries


- Where data files are stored

## WORK library

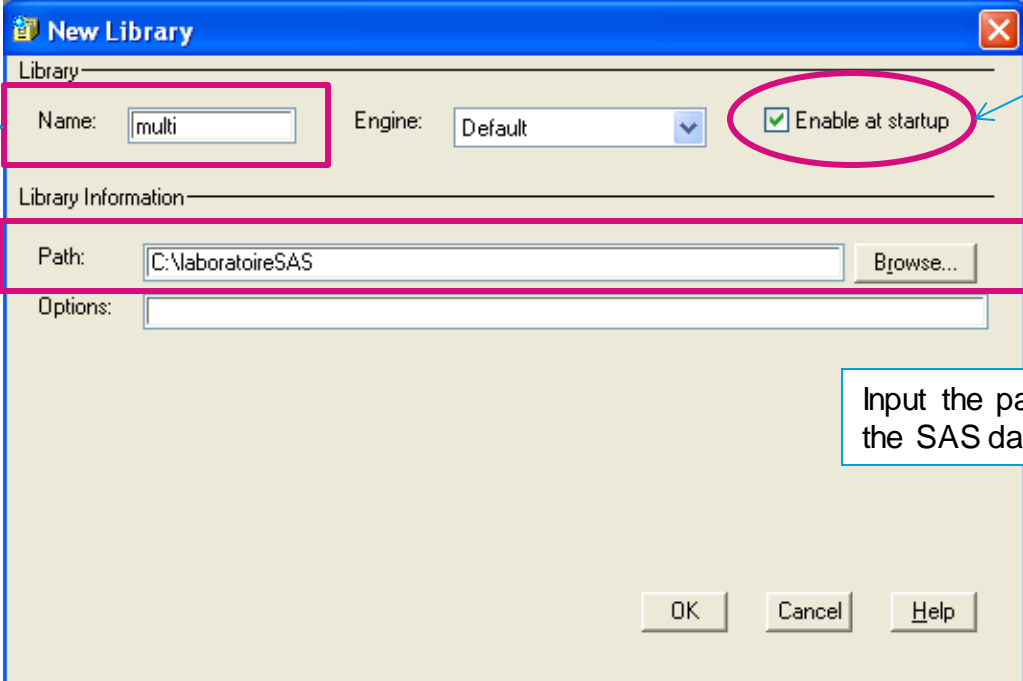
- Default library where data files are stored
- Default library where temporary files are stored

# Libraries

## Creating a library

- Click on the *New Library* button in the tool bar 
- The library name can have a maximum of 8 characters

Library name: maximum 8 characters →



Library Information

Path: C:\laboratoireSAS Browse...

Options:

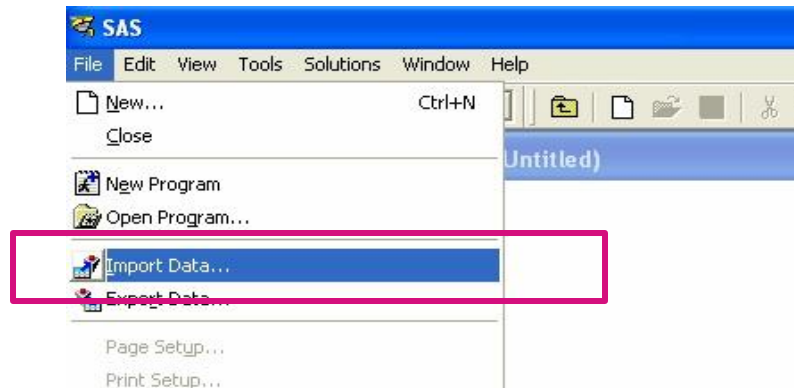
OK Cancel Help

If you check « Enable at startup » the library will be loaded each time you open a new SAS session.

Input the path to the folder where the SAS data files are saved.

# Importing a comma separated file (CSV)

- CSV or XLS data files can be imported using the SAS assistant



# Creating a dataset

- Define a new data file by manually inputting the data lines.

```
data depart;  
  input age age_cat n_visite n_visite_cat sexe;  
cards;  
32 2 10 2 1  
30 2 6 2 1  
48 3 11 3 0  
17 1 21 3 0  
;  
run;
```



VIEWTABLE: Work.Depart						
	age	age_cat	n_visite	n_visite_cat	sexe	
1	32	2	10	2	1	
2	30	2	6	2	1	
3	48	3	11	3	0	
4	17	1	21	3	0	

# Conclusions on data files

- There are essentially 2 ways to define a new dataset in SAS: we can either
  - input the data ourselves
  - Or we can modify an existing dataset

```
data transforme;  
  set depart;  
  region="Montreal";  
  age_2020=age+4;  
run;
```



**Modifying an  
existing dataset**



# DATA and PROC statements

## Part II



HEC MONTRÉAL

# SAS naming conventions, variable types

Table and variable names:

- Maximum length of 32 characters
- Must begin with a letter or an underscore ( \_ )
- All other characters in the name should be letters, numbers or underscores

Types of variables

- **Character chains** (alphanumeric)
- **Numerical**
  - Note that there are various display formatting options for dates, amounts in dollars and percentages.
- **Missing values**
  - For numerical variables, missing values are represented by a period .
  - For alphanumeric variables (characters), missing values are represented by a space between quotations ( " ") or apostrophes ( ' ' )

# Creating datasets: temporary and permanent

DATA step:

- The principal (**data**) statement specifies the name of the dataset/table that will be created as well as the library where the table will be stored.  
→ libref.table\_name
- The **set** instruction makes reference to a table that already exists

Recall the `aapl` dataset imported into the `multi` library during the first exercises.

```
*Fichier de données aapl importé dans la bibliothèque multi;
```

```
data aapl;  
  set multi.aapl;  
run;
```

temporary data file

```
data work.aapl;  
  set multi.aapl;  
run;
```

# Creating datasets: temporary and permanent

DATA step:

- Note: if a SAS dataset/table is to be available to use in future SAS sessions, it is important to store it in a permanent library (e.g., *multi*)

```
data multi.aapl_final;  ← permanent data set
  set aapl;
run;
```

- The contents of the WORK library is deleted at the end of every SAS session.

# Transforming variables

## Operations

- + Addition
- Subtraction
- \* Multiplication
- / Division
- \*\* Exponentiation
- || Concatenation character chains

```
data intro_ex1_mod;  
    set intro_ex1;  
    age_2020=age+4;  
run;
```

If we perform operations with missing records, the transformation yields missing values.

# Transforming variables

## Mathematics functions

- **LOG**(*number*) : natural logarithmic
- **EXP**(*number*) : exponentiate
- **ABS**(*number*) : absolute value
- **SQRT**(*number*) : square root
- **INT**(*number*) : truncate (to whole number)
- **ROUND**(*number*, *rounding digits*) : Rounding (*unit* = 1 /0.1/ 100/...)

```
*Fonctions mathématiques;  
data intro_ex1_mod;  
    set intro_ex1_mod;  
    n_visite_mois=round(n_visite/12,0.1);  
run;
```

# Transforming variables

- **MIN**(*nombre1,nombre2,...*): minimum of a series of variables
- **MAX**(*nombre1,nombre2,...*): maximum of a series of variables
- **SUM**(*nombre1,nombre2,...*): sum of a series of variables
- **MEAN**(*nombre1,nombre2,...*): mean of a series of variables
- **MEDIAN**(*nombre1,nombre2,...*): median of a series of variables
- **NMISS**(*nombre1,nombre2,...*): number of missing values
- **N**(*nombre1,nombre2,...*): number of non-missing values

```
*Fonctions statistiques;  
data intro_ex1_mod;  
    set intro_ex1_mod;  
    n_visite_max=max(n_visite,n_visite_future);  
run;
```

# Logical statements

## The IF...THEN et ELSE IF...THEN statements

- Allow to apply an instruction to a subset of observations
- Useful for recoding a continuous variable into a categorical variable, for example, categorizing age into different age brackets.
- Syntax:

```
IF expression THEN instruction;  
ELSE IF expression THEN instruction;  
:  
:  
ELSE instruction;
```

- **IF** allows to state a logical expression to select a subset of observations
- **THEN** gives the instruction to be applied to the targeted observations.
- **ELSE** indicates that the expression or instruction only applies to the remaining observations that have not previously been selected.



# Logical statements

## The IF...THEN et ELSE IF...THEN statements

```
*Conditions;  
data intro_ex1_mod;  
  set intro_ex1;  
  if age <= 17 then adulte=0;  
  else adulte=1;  
  if (adulte=1 & n_visite_cat = 3) then adulte_fanatigue = 1;  
  else adulte_fanatigue = 0;  
  if (n_visite=1 | n_visite=2) then visiteur_frequent=0;  
  else visiteur_frequent=1;  
run;
```

# Processing variables

**DROP=** variables to delete in the dataset/table

```
*Sélection de variables DROP;  
data intro_ex1_mod2;  
    set intro_ex1_mod;  
    drop age adulte_fanatigue visiteur_frequent;  
run;
```

**KEEP=** variables to keep in the dataset/table

```
*Sélection de variables KEEP;  
data intro_ex1_mod2;  
    set intro_ex1_mod;  
    keep age adulte_fanatigue visiteur_frequent;  
run;
```

# Select all variables within range

- Several variables can be specified within a given instruction. We can use the following syntax to make reference to several variables at a time:
  - `sat1-sat3: variables sat1, sat2, sat3.`
  - `varA -- varB: all variables between varA et varB in the dataset/table.`

```
data Elnino2;  
  set Multi.Elnino;  
  keep obs year--day x1-x5;  
run;
```

=

```
data Elnino2;  
  set Multi.Elnino;  
  keep obs year month day x1 x2 x3 x4 x5;  
run;
```

# Selecting observations

Creating a new dataset/table that contains a subset of observations from some reference dataset.

The IF statement followed by a logical condition allows to:

- select the observations to be kept  
or
- identify the observations to be excluded and subsequently deleted

Keep observations if the indicator `visiteur_frequent` is equal to 1.

```
*Sélection d'observations;  
data intro_ex1_mod3;  
  set intro_ex1_mod;  
  if (visiteur_frequent=1);  
run;
```

Delete observations if the indicator `visiteur_frequent` is different from 1.

```
data intro_ex1_mod3;  
  set intro_ex1_mod;  
  if (visiteur_frequent NE 1) then delete;  
run;
```

# Procedures

- PROC **CONTENTS** : to obtain information of the data file
- PROC **PRINT**: print tables in SAS output
- PROC **FREQ** : frequency tables and cross tabular tables
- PROC **MEANS**: descriptive statistics (e.g., mean, standard deviation)
- PROC **UNIVARIATE**: descriptive statistics and graphs
- PROC **SORT** : sort datasets
  - NODUP and NODUPKEY options: remove duplicates
  - MERGE option: merge databases
- PROC **TRANSPOSE** (change format from wide to long and vice-versa)

# PROC CONTENTS

**Provides information on the dataset/table:**  
number of observations, number of variables,  
list of variables, creation date, etc.

```
proc contents data=intro_ex1_mod;  
run;
```

The CONTENTS Procedure

Data Set Name	WORK.INTRO_EX1_MOD	Observations	100
Member Type	DATA	Variables	12
Engine	V9	Indexes	0
Created	Tue, Aug 23, 2016 10:05:10 o'clock AM	Observation Length	96
Last Modified	Tue, Aug 23, 2016 10:05:10 o'clock AM	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_32		
Encoding	wlatin1 Western (Windows)		

Example  
of SAS  
output



# PROC CONTENTS

## Example of SAS output (continued)...

```
proc contents data=intro_ex1_mod;  
run;
```

Alphabetic List of Variables and Attributes			
#	Variable	Type	Len
10	adulte	Num	8
11	adulte_fanatique	Num	8
1	age	Num	8
7	age_2020	Num	8
2	age_cat	Num	8
3	n_visite	Num	8
4	n_visite_cat	Num	8
5	n_visite_future	Num	8
9	n_visite_max	Num	8
8	n_visite_mois	Num	8
6	sexe	Num	8
12	visiteur_frequent	Num	8

# PROC PRINT

This procedure **print** outputs data in the SAS results window. For example, the following prints the first five observations

```
proc print data=multi.elnino (obs=5);  
run;
```

Obs.	obs	year	month	day	date	latitude	longitude	zon_winds	mer_winds	humidity	air_temp	s_s_temp
1	1	80	3	7	800307	-0.02	-109.46	-6.8	0.7	.	26.14	26.24
2	2	80	3	8	800308	-0.02	-109.46	-4.9	1.1	.	25.66	25.97
3	3	80	3	9	800309	-0.02	-109.46	-4.5	2.2	.	25.69	25.28
4	4	80	3	10	800310	-0.02	-109.46	-3.8	1.9	.	25.57	24.31
5	5	80	3	11	800311	-0.02	-109.46	-4.2	1.5	.	25.30	23.19

As for other procedures, it is possible to hide certain default characteristics of the proc, here the observation id already given in the first column.

```
proc print data=multi.elnino (obs=10) noobs;  
run;
```

Le Système SAS												
obs	year	month	day	date	latitude	longitude	zon_winds	mer_winds	humidity	air_temp	s_s_temp	
1	80	3	7	800307	-0.02	-109.46	-6.8	0.7	.	26.14	26.24	
2	80	3	8	800308	-0.02	-109.46	-4.9	1.1	.	25.66	25.97	
3	80	3	9	800309	-0.02	-109.46	-4.5	2.2	.	25.69	25.28	
4	80	3	10	800310	-0.02	-109.46	-3.8	1.9	.	25.57	24.31	
5	80	3	11	800311	-0.02	-109.46	-4.2	1.5	.	25.30	23.19	
6	80	3	12	800312	-0.02	-109.46	-4.4	0.3	.	24.72	23.64	
7	80	3	13	800313	-0.02	-109.46	-3.2	0.1	.	24.66	24.34	
8	80	3	14	800314	-0.02	-109.46	-3.1	0.6	.	25.17	24.14	
9	80	3	15	800315	-0.02	-109.46	-3.0	1.0	.	25.59	24.24	
10	80	3	16	800316	-0.02	-109.46	-1.2	1.0	.	26.71	25.94	



# PROC FREQ

The FREQ procedure

- Produces frequency tables for numerical and alphanumerical variables.
- Produces one-, two- and even three-dimensional tables.
- Can also calculate different statistics, such as the odds ratio, relative risk, etc.

```
PROC FREQ data=table_SAS <options>;  
    BY variables;  
    TABLES variable(s) / <options>;  
RUN;
```

# PROC FREQ

## One-dimensional table

- The TABLES instructions specifies the variable(s) to be used in the frequency table.

```
proc freq data=intro_ex1;  
    tables n_visite_cat;  
run;
```

The FREQ Procedure

n_visite_cat	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	14	14.00	14	14.00
2	63	63.00	77	77.00
3	23	23.00	100	100.00

# PROC FREQ

## Two-dimensional table

- Produces cross-tabular tables for two variables.
- The two variables are specified in the TABLES instruction. The symbol (\*) indicates that the two variables will be crossed.
- It is possible to create several tables within the same TABLES instruction.

```
❏ proc freq data=intro_ex1;  
    tables age_cat*n_visite_cat;  
run;
```

# PROC FREQ

## SAS output (results):

```
proc freq data=intro_ex1;  
  tables age_cat*n_visite_cat;  
run;
```

Percentages (row) →

Percentages (column) →

Frequency Percent Row Pct Col Pct	Table of age_cat by n_visite_cat				
	age_cat	n_visite_cat			
		1	2	3	Total
1	1	11	6	14	31
		11.00	6.00	14.00	31.00
		35.48	19.35	45.16	
		78.57	9.52	60.87	
2	2	2	32	4	38
		2.00	32.00	4.00	38.00
		5.26	84.21	10.53	
		14.29	50.79	17.39	
3	3	1	25	5	31
		1.00	25.00	5.00	31.00
		3.23	80.65	16.13	
		7.14	39.68	21.74	
Total	Total	14	63	23	100
		14.00	63.00	23.00	100.00

# PROC MEANS

The MEANS procedure

- Provides descriptive statistics for a numerical variable: number of values, sum, mean, standard deviation, minimum, maximum, median, quantiles, etc.

```
PROC MEANS data=table_SAS <statistics> <options>;  
    BY variables;  
    CLASS variable(s) / <options>;  
    VAR variable(s);  
RUN;
```

# PROC MEANS

The VAR instruction specifies the numerical variables to perform the procedure on (if a variable is not specified, then all variables will be used).

The default descriptive statistics are:

- the number of non-missing observations;
- the mean (MEAN);
- the standard deviation (Std Dev);
- the minimum (Minimum)
- the maximum (Maximum).

```
proc means data=intro_ex1_mod;  
  var age;  
run;
```

The MEANS Procedure

Analysis Variable : age				
N	Mean	Std Dev	Minimum	Maximum
100	30.2800000	9.9392905	14.0000000	61.0000000

# PROC MEANS

List of some of the available statistics

Option PROC MEANS	Statistic
N	Number of complete observations
NMISS	Number of missing observations
MEAN	Mean
SUM	Sum
MIN	Minimum
MAX	Maximum
MEDIAN	Median
STD	Standard deviation
VAR	Variance
CLM	95% confidence interval (for the mean)
Q1	1 <sup>st</sup> quartile (25 <sup>th</sup> percentile)
Q3	2 <sup>nd</sup> quartile (75 <sup>th</sup> percentile)
QRANGE	Interquartile range (Q3-Q1)

**Note:** the option `maxdec=value` is useful to control the number of decimal places displayed in the results table, for example:

```
proc means data=multi.elnino maxdec=2;
```

In this example, all of the statistics except N and NMISS will be reported with two decimal places. This option applies to all types of statistics.

```
proc means data=intro_ex1_mod n mean std maxdec=2;  
  var age;  
run;
```

# PROC SORT

The SORT procedure allows to sort data

```
proc sort data=intro_ex1_mod;  
  by age_cat descending n_visite;  
run;
```

Options:

- **BY:** identifies the variable(s) according to which the data will be sorted.
- **DESCENDING:** sort in descending order. When this option is specified **before** a variable, the data will be sorted such that the largest value for this variable is placed first and the smallest value is at the end.

VIEWTABLE: Work.Intro\_ex1\_mod

	age	age_cat	n_visite
24	18	1	3
25	22	1	2
26	18	1	2
27	18	1	2
28	21	1	1
29	24	1	1
30	24	1	1
31	17	1	1
32	30	2	12
33	29	2	11
34	33	2	11
35	32	2	11
36	32	2	10
37	25	2	10
38	34	2	10
39	28	2	10
40	34	2	10
41	26	2	10
42	29	2	10
43	26	2	9
44	34	2	9
45	31	2	9

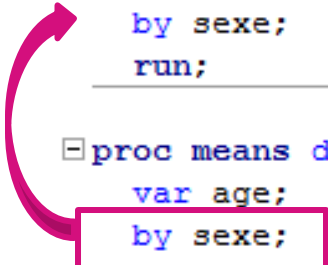


# PROC MEANS

The BY instruction:

- Groups descriptive statistics according to one (or more) categorical variables specified in the BY instruction.
- **Careful:** the table must be sorted **beforehand** according to the variable(s) specified in the BY instruction.

```
proc sort data=intro_ex1_mod;  
  by sexe;  
run;  
  
proc means data=intro_ex1_mod n nmiss mean std median min max;  
  var age;  
  by sexe;  
run;
```



# PROC MEANS

## The MEANS Procedure

sexe=0

Analysis Variable : age						
N	N Miss	Mean	Std Dev	Median	Minimum	Maximum
59	0	29.1864407	10.2460667	28.0000000	14.0000000	56.0000000

sexe=1

Analysis Variable : age						
N	N Miss	Mean	Std Dev	Median	Minimum	Maximum
41	0	31.8536585	9.3796614	31.0000000	18.0000000	61.0000000

**BY:** the results will be shown for each level of the categorical variable specified in the BY command

# MERGE option

To merge datasets in SAS, we use the **MERGE** option

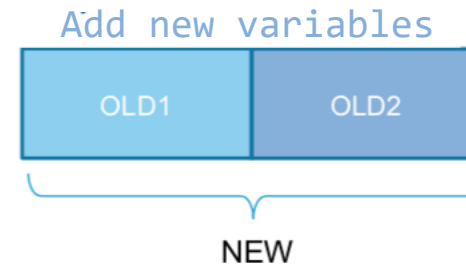
Data must always be sorted according to the same variable before using **MERGE** in a data step

```
data multi.fahrenheit;  
set multi.elnino;  
air_temp_f=(air_temp*(9/5))+32;  
keep obs air_temp_f;  
run;
```

```
proc sort data=multi.elnino;  
by obs;  
run;
```

```
proc sort data=multi.fahrenheit;  
by obs;  
run;
```

```
data elninomerge;  
merge multi.elnino multi.fahrenheit;  
by obs;  
run;
```



# PROC TRANSPOSE

The **TRANSPOSE** procedure creates a restructured data by transposing selected variables, thereby making them into observations (or columns). This is helpful to transform tables from wide to long format, notably for correlated or longitudinal data.

- **by** gives the columns we want to keep "as is".
- **var** indicates the variables to be transposed (stacked)
- Any column not selected is omitted from the copy whose name is specified by **out**
- SAS creates a categorical variable (**name**) to label the transposed variables

```
proc transpose
  data=multi.elnino
  out=elnino_long
  name=zone
  prefix=temp;
by obs--mer_winds;
var air_temp s_s_temp;
run;
```

obs	zone	temp1
1	air_temp	26.14
1	s_s_temp	26.24
2	air_temp	25.66
2	s_s_temp	25.97

# ODS

## Part III



**HEC MONTRÉAL**

# ODS: Export to a document

ODS allows one to organize data export and results output into a PDF, RTF or HTML document.

```
ods rtf;
```

```
proc freq data=intro_ex1;  
  tables age_cat*n_visite_cat;  
run;
```

```
ods rtf close;
```

The SAS System

The FREQ Procedure

Table of age\_cat by n\_visite\_cat

age_cat	n_visite_cat			
	1	2	3	Total
Frequency				
Percent				
Row Pct				
Col Pct				
1	11 11.00 35.48 78.57	6 6.00 19.35 9.52	14 14.00 45.16 60.87	31 31.00
2	2 2.00 5.26 14.29	32 32.00 84.21 50.79	4 4.00 10.53 17.39	38 38.00
3	1 1.00 3.23 7.14	25 25.00 80.65 39.68	5 5.00 16.13 21.74	31 31.00
Total	14 14.00	63 63.00	23 23.00	100 100.00

# ODS: Storing results in a table

- The ODS module also allows to select a portion of SAS output and to store it as a dataset.

A description of each part of the output is given in the journal (Log)

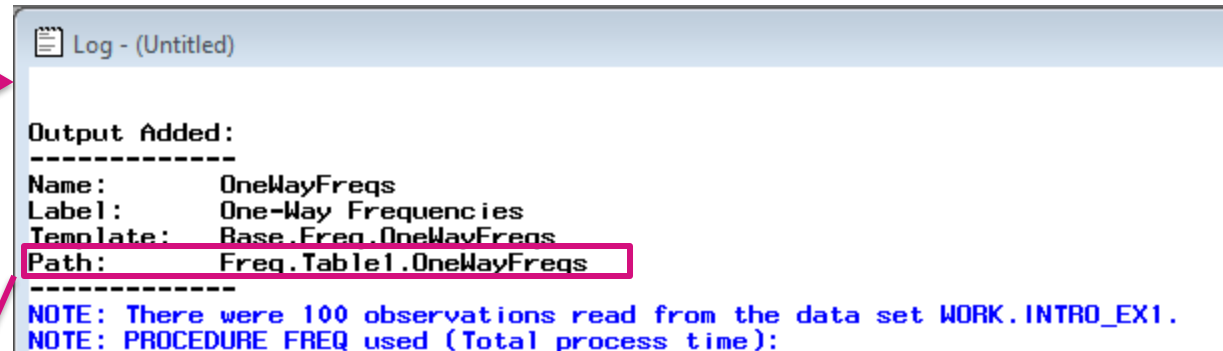
```
ods trace on;
```

---

```
proc freq data=intro_ex1;  
  tables age;  
run;
```

---

```
ods trace off;
```



```
Log - (Untitled)  
  
Output Added:  
-----  
Name:      OneWayFreqs  
Label:     One-Way Frequencies  
Template:   Base.Freq.OneWayFreqs  
Path:      Freq.Table1.OneWayFreqs  
-----  
NOTE: There were 100 observations read from the data set WORK.INTRO_EX1.  
NOTE: PROCEDURE FREQ used (Total process time):
```

```
proc freq data=intro_ex1;  
  tables age;  
  ods output Freq.Table1.OneWayFreqs=dist_age;  
run;
```

# ODS: Storing results in a table

- The results table is called Freq.Table1.OneWayFreqs.
- It will be saved in the table **dist\_age**.

```
proc freq data=intro_ex1;  
  tables age;  
  ods output Freq.Table1.OneWayFreqs=dist_age;  
run;
```

VIEW TABLE: Work.Dist_age (One-Way Frequencies)							
	Table	age	age	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	Table age	14	14	1	1.00	1	1.00
2	Table age	15	15	2	2.00	3	3.00
3	Table age	16	16	2	2.00	5	5.00
4	Table age	17	17	3	3.00	8	8.00
5	Table age	18	18	7	7.00	15	15.00
6	Table age	19	19	2	2.00	17	17.00
7	Table age	20	20	1	1.00	18	18.00
8	Table age	21	21	1	1.00	19	19.00
9	Table age	22	22	7	7.00	26	26.00
10	Table age	23	23	1	1.00	27	27.00
11	Table age	24	24	4	4.00	31	31.00
12	Table age	25	25	6	6.00	37	37.00
13	Table age	26	26	4	4.00	41	41.00
14	Table age	27	27	2	2.00	43	43.00
15	Table age	28	28	4	4.00	47	47.00
16	Table age	29	29	4	4.00	51	51.00
17	Table age	30	30	4	4.00	55	55.00



# Graphs

## Part IV



HEC MONTRÉAL

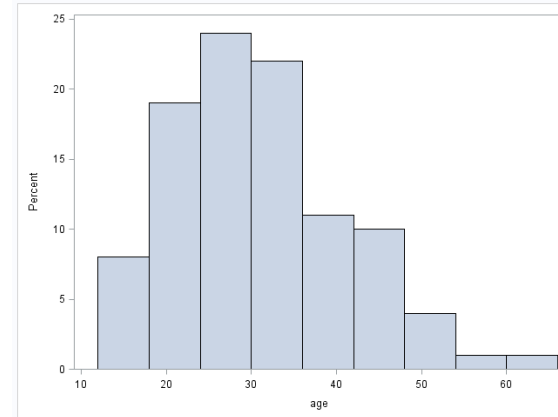
# SGPLOT procedure

## PROC SGPLOT

- Graphs that display the distribution of a variable: histogram, etc.
- Graphs that display relations between two variables: scatterplots, boxplots, etc.
- Possible to superimpose graphs
- Many options available to control the appearance of the graph and add various elements, such as a legend.

```
proc sgplot data=intro_ex1;  
  histogram age;  
run;
```

Specifies the type of graph



# SGPLOT procedure

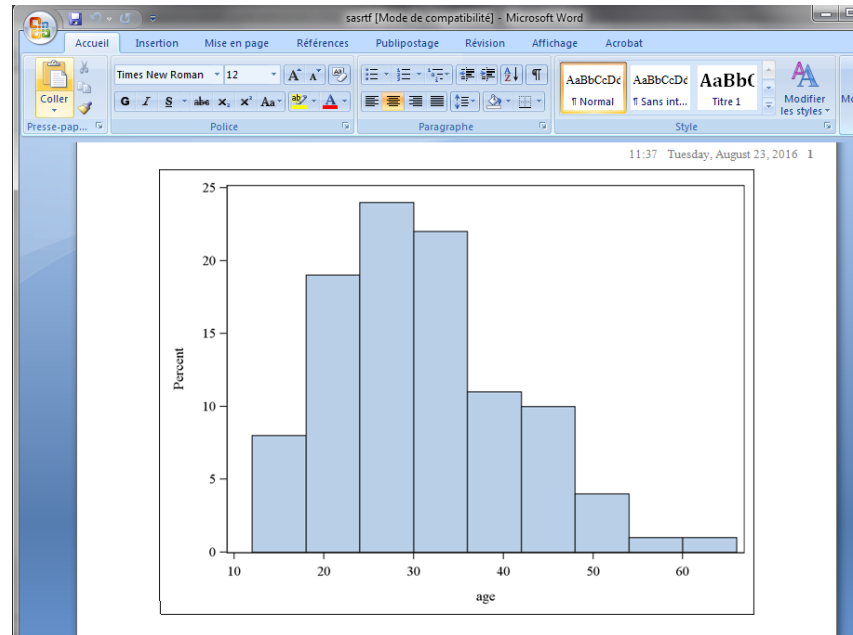
## ODS and graphs

- Graphs can be automatically output in a PDF or Word document using ODS
- This graph will have a better appearance.

```
ods graphics on;  
ods rtf;
```

```
proc sgplot data=intro_ex1;  
  histogram age;  
run;
```

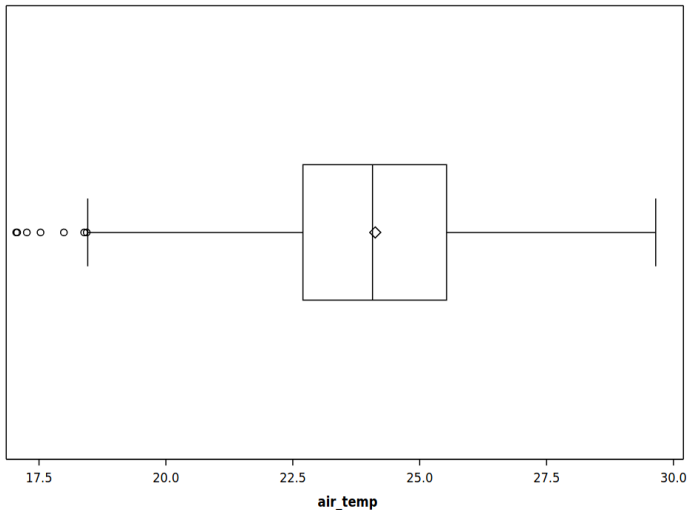
```
ods rtf close;  
ods graphics off;
```



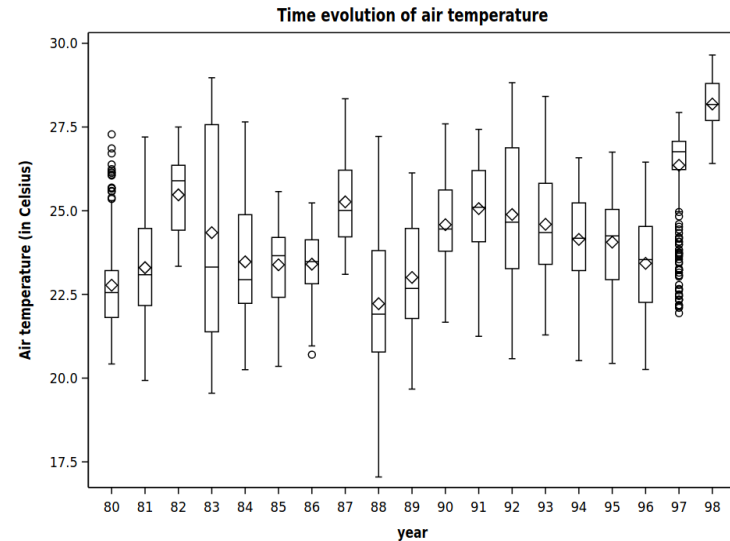
# PROC SGPLOT: Box-and-whiskers plots

The **SGPLOT** procedure can be used to create horizontal (**hbox**) or vertical (**vbox**) boxplots, but also a boxplot per category.

```
proc sgplot data=multi.elnino;  
hbox air_temp;  
run;
```



```
proc sgplot data=multi.elnino;  
vbox air_temp / category = year;  
title "Time evolution of air temperature";  
yaxis label = "Air temperature (in Celsius)";  
run;
```



# Consulting the SAS help

## Part V



HEC MONTRÉAL

# Helpful references

- Ron Cody. *Learning SAS by Example : A Programmer's Guide*. Sas Institute Inc.
- Lora D. Delwiche, Susan J. Slaughter. 1998. *The Little SAS Book*, Fourth Edition. Sas Institute Inc.
- SAS Help and Documentation: <https://support.sas.com/en/support-home.html>
- SAS online tech support: <https://support.sas.com/en/technical-support.html>
- SAS website: <https://www.sas.com>
- Website with interesting graphics examples :  
<http://support.sas.com/sassamples/graphgallery/index.html>