Graduate School Class Reminders

- Maintain six feet of distancing
- Please sit in the same chair each class time
- Observe entry/exit doors as marked
- Use hand sanitizer when you enter/exit the classroom
- Use a disinfectant wipe/spray to wipe down your learning space before and after class
- Media Services: 414 955-4357 option 2

Documentation on the web

- CRAN: `http://cran.r-project.org`
- R manuals: `https://cran.r-project.org/manuals.html`
- SAS: `http://support.sas.com/documentation`
- SAS 9.3: `https://support.sas.com/en/documentation/documentation-for-SAS-93-and-earlier.html`
- Step-by-Step Programming with Base SAS 9.4 (SbS): `https://documentation.sas.com/api/docsets/basess/9.4/content/basess.pdf`
- SAS 9.4 Programmer s Guide: Essentials (PGE): `https://documentation.sas.com/api/docsets/lepg/9.4/content/lepg.pdf`
- Wiki: `https://wiki.biostat.mcw.edu` (MCW/VPN)

HW: stratified random sampling and the NTDB

- Write a SAS DATASTEP program to perform stratified random sampling: see the details in lecture 4, slide 7
- Hints: use the `rand("unif")` function and the `ordinal` function to create permutations
- A variable list can be used in many functions with the of clause like `of VAR1-VARn`
  for example, `ordinal(m, of VAR1-VARn)` instead of
  `ordinal(m, VAR1, ..., VARn)`

Sorting data sets with `proc sort`

- A big part of learning the *SAS way* of doing things is working with sorted data sets
- You can sort a data set in ascending order
  `proc sort data=OLD out=NEW; by VAR1 ... VARn:`
  `run;`
  Or `proc sort data=OLD; by VAR1 ... VARn: run;`
  if you have write access to `OLD`
- Or descending order: each corresponding `VAR` needs the `descending` modifier since ascending is the default
  `proc sort data=OLD out=NEW; by descending VAR;`
  `run;`

What is a unique key?

- What is a unique key?
- Each NTDB patient is anonymized by the identifier `inc_key`: is `inc_key` a unique key, i.e., ONE record for each distinct value?
- If so, then the `/UNIQUE` option will succeed
  if NOT, it will generate an error
- Each hospital is represented by the anonymized identifier `traumactr`: is `traumactr` a unique key?

```
proc sort data=ntdb.elder
    out=traumactr(index=(inc_key/UNIQUE));
    by traumactr inc_key;
run;
```

Creating a unique key with PROC SORT

- There are several ways to create a unique key when one doesn't exist
- For example, there is the PROC SORT option `NODUPKEY`

```
proc sort NODUPKEY data=ntdb.elder out=nodupe;
    by inc_key;
run;
```

DATASTEP automatic variables and automatic macro variables

- *Automatic* variables are temporary and not stored in the `NEW` data set: typically, they start and end with an underscore
- `_N_` is the number of the current observation
  `_N_=1` for the first, etc.
- `_ERROR_` is 1 if an error has occurred in the current observation and 0 otherwise
- `set OLD end=LAST` produces the variable `LAST` which is 1 for the last observation and 0 otherwise
- `data NEW; set OLD end=LAST; if LAST; run;`
- Also, there are *PDV* lists: `_ALL_`, `_NUMERIC_` and `_CHARACTER_`
- Like an automatic variable, the keyword `_last_` is the last data set actually created
  `data NEW2; set _last_; run;`
- The same as the `syslast` automatic macro variable
  `data NEW2; set &syslast; run;`
- *Automatic* macro variables start with `sys`

By-group processing

- In the DATASTEP, the `by` statement is very useful when the data set is sorted by one or more variables
  `data NEW; set OLD; by VAR1 ... VARn;`
- There are two *automatic* variables for each `VAR`
- `FIRST.VAR` is 1 at the first of observation of a *by-group* and 0 for all others
- `LAST.VAR` is 1 at the last of observation of a *by-group* and 0 for all others
- If there is only one record in the *by-group*, then `FIRST.VAR=LAST.VAR=1`
- If there is more than one record per *by-group*, you can create a unique key, if needed, with a subsetting IF: `if FIRST.VAR=1;`
- Or alternatively: `if LAST.VAR=1;`

Summaries of multiple observations

- The `retain` statement creates a variable that RETAINs its value across DATASTEP observations
  unlike variables in a data set which acquire a new value from each observation due to automatic looping
- `retain VAR1 VALUE1 ... VARn VALUEn;`
  the starting values are `VAR1=VALUE1; ...; VARn=VALUEn;`
- for no starting value, place the variables at the end
  `retain X1 VALUE1 ... Xn VALUEn Y1 ... Ym;`
  `Y1 ... Ym` are missing
- For example, suppose that you want a total of the variable `Z`
  `data NEW; set OLD end=LAST; retain TOT 0;`
  `keep TOT; TOT+Z; if LAST; run;`
- NTDB: let's calculate average annual case volume for each trauma center
- see `NTDB/sas/volume.sas`