SAS® Summer Institute 2009
Part I
Accessing Data with SAS

University of Iowa SAS User Group
Tuesday, July 7, 2009

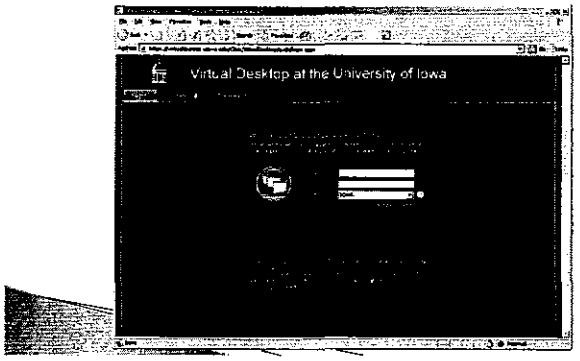
Bill McKinney
Center for Research in the Implementation of Innovative Strategies in Practice
Department of Veterans Affairs Medical Center, Iowa City, IA

Let's Start SAS 9.2

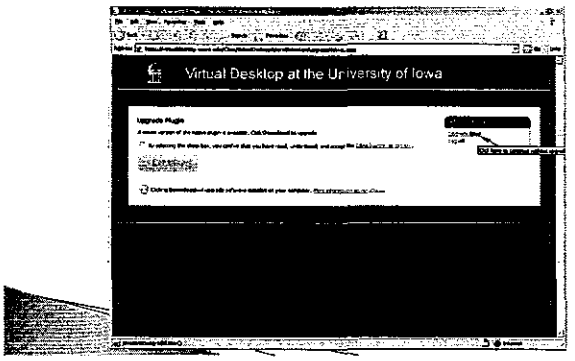
- Follow-along in the next slides as we:
  1. Locate UI Virtual Desktop
     http://virtualdesktop.uiowa.edu
  2. Login with our HawkIDs
  3. Start SAS 9.2
- SAS mentors are available to assist

Virtual Desktop Login Screen

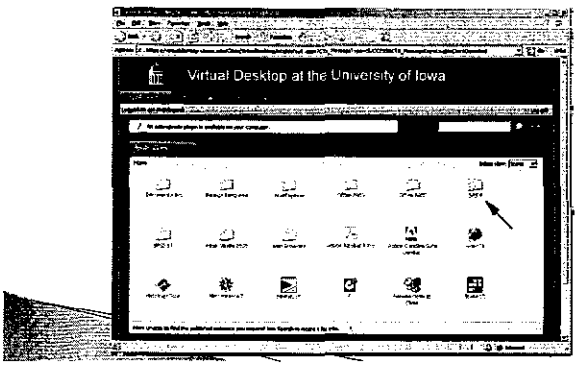
http://virtualdesktop.uiowa.edu



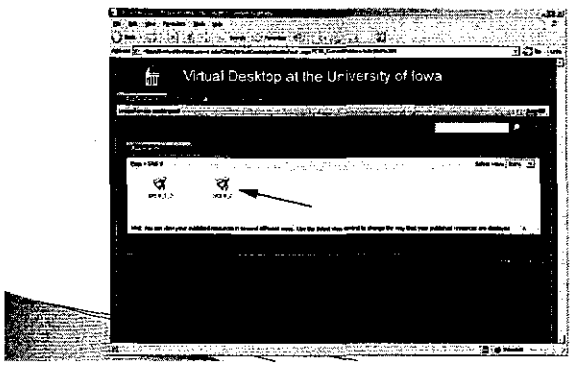
Click 'Upgrade Later'...



Select the 'SAS 9' folder



Select SAS 9_2



Overview: Accessing Data

Two main parts

I. SAS interface & Data Access:

INTERMISSION...

II. Import/Export:

- At any point along the way we can play around with the code, and try other ideas

Overview: Accessing Data

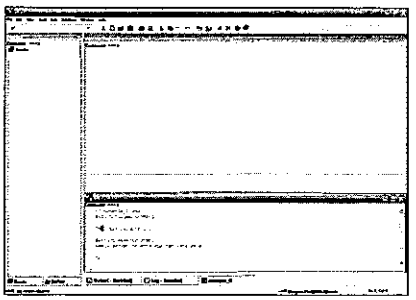
Details...

SAS Interface and Data Access:
- Quick review of the SAS interface & customization tips
- Example 1: SAS libraries and basic dataset concepts
- Example 2: Reading / Writing text (ASCII) data

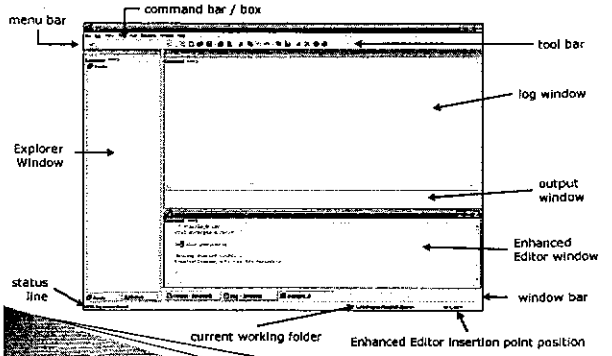
INTERMISSION...

Import/Export:
- Example 3: Import / Export (MS-Excel)
- Example 4: Import / Export (MS-Access)
- Other formats if time permits

The SAS interface



The SAS interface



The SAS interface

- SAS opens up with a default user profile
  - That's what you see right now
- The 'profile' can be customized
  - Window size and placement
  - Fonts and type size
  - Custom toolbar icons
  - Other stuff...

Customizing your SAS session

- We can change many things with the default profile and save those settings
- First, let's use a command box to clear the log:
  - Locate the command box and type
    CLEAR LOG;
    and press the 'Enter' key

Customizing your SAS session

- A similar command will clear the OUTPUT window.
  - Let's make some output...
  - in the program editor type:

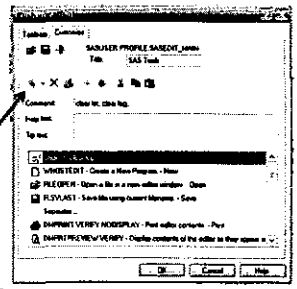
```
proc contents data=sashelp.class;
run;
```

...select and 'run' this program

  - Run the program by pressing the RUNNING FIGURE icon at the top of your toolbar
  - CONTENTS will provide information about the dataset we specify (LIBRARY=sashelp, DATASET=class.)

Add a tool...

- Let's add a tool to help us clear the OUTPUT and LOG windows
- Right click your toolbar
- Select CUSTOMIZE
- Click the icon with the light blue dot



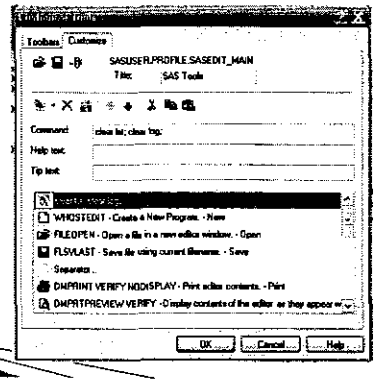
Add a tool...

- Add the commands:
  - CLEAR LST; CLEAR LOG;
- Pick an icon (seven clicks right...look for the pencil erasing paper)
- Save when prompted
  - This changes your **PROFILE.sas7bcat** file
    - The profile is typically found in:

C:\Documents and Settings\[USER]\My Documents\My SAS Files\9.2

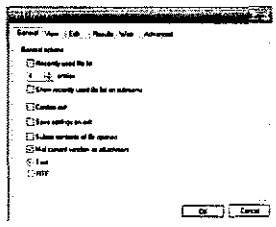
Add a tool...

You can add
Help text
Tip text and
Use Arrows to
Select location of
Tools in toolbar



Let's change the windows...

- There's plenty we can change.
  - To save window positions go to:
    - Tools / options / preferences
    - And check
      - SAVE SETTINGS ON EXIT
  - Let's explore other settings before moving on to our first program lessons...



Example 1 - overview

- Open the Example 1 program from your SAS_pgm folder: sas_pgms\example_1.sas
  - Lessons:
    1. sashelp library
    2. the 'work' library (temporary)
    3. 'hard-coding' data
    4. macro tip (reduce repetition)
    5. 'put' stuff in log, then file
    6. read text file and print report
    7. save dataset as permanent file

SAS data libraries (LIBNAME)

- What's a SAS Library?
  - A shortcut: to a directory 'path'
  - A filter: in the SAS explorer window, only SAS objects are displayed
    - (i.e.: datasets, catalogs, other SAS stuff)
  - We can define them with the mouse, with SAS code, or use existing libraries
- Existing Libraries
  - SAShelp is a good place to start
    - Provides a variety of sample datasets and catalogs
    - Is typically included with every SAS installation
  - Let's BROWSE SAShelp.CLASS

Let's browse some data...

```
/* LESSON 1 */
/* let's browse some data:
   SASHELP.CLASS

Look in the 'explorer' window
   left-most window
   lower tab: Explorer

Click the icon named 'SASHELP' and look for 'Class'.
Double-click 'Class'
Looks like Excel, but not as friendly
Close the window
Right-click the 'Class' icon and select VIEW COLUMNS
This provides a quick view of the dataset characteristics
   (CONTENTS)
*/
```

Let's make a copy of the data

```
/* LESSON 2 */
/* now let's make a WORKing copy of CLASS */
data class; SET sashelp.class;
run;
```

Remember when we ran CONTENTS on this dataset?

Let's 'hard-code' the data...

```
/* LESSON 3 */
/* let's pretend we are HARD-CODING these
   data
in the SAS program window using a
   CARDS/DATALINES statement
NOTE: you can substitute CARDS, below with
DATALINES, CARDS4 or DATALINES4
*/
```

Let's do less typing...

- The SAS macro facility is generally not for beginners, except for this little trick...

```
/* LESSON 4 ... touching on advanced stuff to save
   typing */

/* Let's make a SHORTCUT to our directory path
   with a MACRO VARIBLE using %LET... */
%let whatpath=H:\uisug\sas_pgms;

/* Let's test the result */
%put &whatpath.;
```

Another use for the LOG window

```
/* LESSON 5 */
/* Let's write data directly to the SAS LOG
   and then put it in a text file */
data class2; set class;
*file "&whatpath.\example_1_class_ages2.txt";
put name @25 age;
run;
```

Let's print a list

```
/* LESSON 6 */
/* Let's read the data we just wrote using the
   same input statment
   from LESSON 3, above
   and then PRINT it.
*/
```

Let's save a permanent dataset

- LESSON 7
  - Let's save our dataset to a permanent SAS dataset file
  - First we define the library name (LIBNAME)
  - Then we use the libname in our dataset statement
  - We will create an 'Age_String' variable to make age categories
  - We will finish by checking the frequencies of age and age_string for class data, and use a FORMAT as another way of categorizing data.

INTERMISSION...

- Let's take a break
  - get a soda
  - chat
  - take care of business
- Come-on back in 15 minutes for...
  - Importing / Exporting
    - MS-Excel
    - MS-Access
    - Other formats if there's time...

The Import/Export Business

- Reading and Writing
  - MS- Access and -Excel
    - PROC IMPORT / EXPORT
    - LIBNAME
  - Open program: examples_3_and_4.sas in H:\uisug\sas_import_export\sas_pgms

Example 3: Proc Import/Export

FIRST...More shortcuts...
Simple macro variables to save typing:

```
/* Define macro variables to store directory path and file name info */
%let whatpath=H:\uisug\sas_import_export\examples\access_data;
%let exclpath=H:\uisug\sas_import_export\examples\excel_data;
%let whatfile=access_data.mdb;

/* Define a macro variable to hold the value of TODAY's date as YYYYMMDD */
%let datestamp=%sysfunc(today(),yymmddN8.);
%put &datestamp;
```

Example 3: Proc Import/Export

Simple macro variables to save typing:

```
%let whatpath=H:\uisug\sas_import_export\examples\access_data;
```

As we saw earlier, the macro variable 'whatpath' holds the characters that follow the equal sign and end at the semi-colon. Anytime the string is needed it can be called with &whatpath.

```
%put &whatpath.;
```

```
H:\uisug\sas_import_export\examples\access_data
```

Example 3: Proc Import/Export

- export / import Access tables
  - write sashelp.class data to Access table
    - we will add date and time variables to illustrate how SAS® dates must be formatted prior to export
      - SAS® dates begin at 1-1-1960 but Microsoft dates begin at 1-1-1900.

Example 3: Proc Import/Export

Using the SASHELP.CLASS data as an example of how to export something to MS-Access

```
/* the REPLACE option must be used with care */
PROC EXPORT DATA= WORK.CLASS
        OUTTABLE= "class_list"
        DBMS=ACCESS REPLACE;
    DATABASE="&whatpath.\&whatfile.";
RUN;
```

Example 3: Proc Import/Export

- export /import Excel speadsheet
  - exporting our WORK.CLASS_LIST example
  - import
    - simple and easy excel structure with variable names in first row
    - more complex excel file
      - » non-SAS® column names
      - » mixed data
      - » mixed date data

Example 3: Proc Import/Export

Using the SASHELP.CLASS data as an example of how to export something to MS-Excel

```
/* time in excel will need to be formatted by point & click
   it will arrive looking something like this...
     1/0/1900  1:47:19 PM                    */
proc export data=WORK.CLASS
   outfile="&excipath.\class_list.xls";
   sheet='Class_list';
run;
```

Example 3: Proc Import/Export

Using the exported 'class_list.xls' data as an example of how to IMPORT something easy from MS-Excel into SAS

```
PROC IMPORT OUT= WORK.test_easy_excel
        DATAFILE= "&excipath.\class_list.xls"
        DBMS=EXCEL REPLACE;
   SHEET="Class_list";
   GETNAMES=YES;
   MIXED=NO;
   SCANTEXT=YES;
   USEDATE=YES;
   SCANTIME=YES;
   RUN;
```

GETNAMES tells SAS® to read variable names from first row

MIXED can convert numeric data types to mixed if there are character data that need to be preserved

SCANTEXT and **SCANTIME** tells SAS® to use the longest WIDTH in a column to set variable width

USEDATE tells SAS® to format date as such instead of DATETIME

Example 3: Proc Import/Export

Using the exported 'class_list.xls' data as an example of how to IMPORT something TOUGH from MS-Excel into SAS

```
PROC IMPORT OUT= WORK.test_tough_excel
        DATAFILE= "&excipath.\tough_excel_class_list.xls"
        DBMS=EXCEL REPLACE;
   SHEET="Class_list";
   GETNAMES=YES;
   MIXED=YES;
   SCANTEXT=YES;
   USEDATE=YES;
   SCANTIME=YES;
   RUN;
```

Browse the imported data to see how the MIXED data are treated with MIXED=YES vs

MIXED=NO

There are many other options to help ease the pain of importing data that are not so easy to import with the standard import wizard settings

Example 4: Read/Write with LIBNAMEs

- LIBNAME assignments are another way to use SAS® to read and write data in different formats like MS-ACCESS

```
libname morpt access "&whatpath.\mo_rpt_de_identified.mdb";
run;
```

Here we create 'morpt' library with the 'access' engine.

Example 4: Read/Write with LIBNAMEs

- Once the libname is established the tables and queries in the Access database file will be available for SAS to read.
- SAS cannot write over the tables using the libname engine
  - this could cause trouble if more than one user is working with the data and you want to overwrite it.

Example 4: Read/Write with LIBNAMEs

- It's easy to read the tables, but if one has spaces in the table name then you must code it like this...

```
/* here the table name has spaces, but SAS can ignore that */
data report; set morpt.'Monthly Report Data'n;
date_seen = datepart(date_seen); format date_seen mmddyy10.;
run;
```

NOTE: Dataset name in the set statement is surrounded by single quotes, and the ending quote is immediately followed by the letter 'n'.

Example 4: Read/Write with LIBNAMEs

- It's easy to write NEW tables using DATA and SET statements.

```
data morpt.report_&datestamp.; set report;
keep auto_id date_seen;
run;
```

NOTE: There is no REPLACE option for the Access Libname engine; therefore, you cannot overwrite, delete or modify existing access tables with this method. Probably a good thing.

Example 4: Read/Write with LIBNAMEs

- It's easy to read MS-EXCEL data with a libname.

```
/* define the file name */
%let book=class_list.xls;
libname excel "&exclpath.\&book."   ver=2002;

data check_class_list; set excel.class_list;
format asof_date mmddyy10. asof_when timeampm.;
run;
```

Example 4: Read/Write with LIBNAMEs

NOTE: When you view the library in your SAS display manager windows you will notice spread sheets appear to be doubled with the second in the pair ending in a dollar sign:

class_list

class_list$

Either version seems to work for reading.

Example 4: Read/Write with LIBNAMEs

It's easy to write new excel spreadsheets within a workbook using the excel libname engine: Here we just added a variable that should appear in the new exported worksheet.

```
data excel.new_class_list; set
check_class_list;
newvar='checked the file';
run;
```

Example 4: Read/Write with LIBNAMEs

Remember to CLEAR your excel libnames; otherwise, the file cannot be browsed because SAS has a lock on it. Once the libname is cleared the file can be opened again and used by others.

```
/* cannot browse excel file until libname is cleared */
libname excel clear;
```

That's about it...

- Should we try other ideas?
- Import/Export other formats
- Take a peek at the import/export 'wizard'

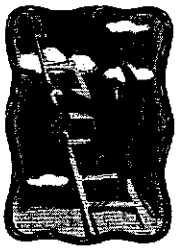
- Do you have any questions?
- Do you have any comments?

Acknowledgements

SUGI-27 Paper: Accessing MICROSOFT EXCEL and MICROSOFT ACCESS Through the Use of a Simple Libname Statement. By Kee Lee, Purdue University, West Lafayette, Indiana.

SUGI-31 Paper: De-Mystifying the SAS® LIBNAME Engine in Microsoft Excel: A Practical Guide

Special Thanks to our SAS Mentors



Thank You

Email Questions / Comments to:
bill-mckirgan@uiowa.edu
Or
lowell.mckirgan@va.gov