Reading From Alternate Sources: What To Do When The Input Is Not a Flat File

presented by Michael Davis
on June 13, 2007 at
PhilaSUG meeting held at
Omnicare Clinical Research,
King of Prussia, PA

Presentation Overview

- program input is not always in flat files and SAS data sets
- often data resides on the web, non-SAS servers, and DBMSs (Oracle, DB2)
- purpose is to survey SAS tools that can be used to access data in other formats
- try this at home!

FTP Access Method

- File Transfer Protocol
- problem: where to park that file?
- SAS can define FTP source as a fileref
- FTP access available wherever TCP used
- syntax:

  FILENAME *fileref* **FTP**
  ‘external file’ <ftp options>

You Can Try It Out

- sample comma-separated value file is posted on the Bassett Consulting web site
- only need access to SAS Release 6.12 or later and any form of Internet connection

```
filename myfile ftp 'baseball.csv'
user= 'ftpdownload'
pass= 'test123'
host= 'bassettconsulting.com' recfm=v debug ;
```

- Import Wizard used to generate code to read file

FTP Log Messages

```
NOTE: 220 ProFTPD 1.3.0a Server (www.bassettconsulting.com)
   [209.35.121.141]
NOTE: <<< 220 ProFTPD 1.3.0a Server (www.bassettconsulting.com)
   [209.35.121.141]
NOTE: >>> USER ftpdownload
NOTE: <<< 331 Password required for ftpdownload.
NOTE: >>> PASS XXXXXXX
NOTE: <<< 230 User ftpdownload logged in.
NOTE: >>> PORT 192,168,1,100,8,151
NOTE: <<< 200 PORT command successful
NOTE: >>> TYPE A
NOTE: <<< 200 Type set to A
NOTE: >>> PWD
NOTE: <<< 257 "/" is current directory.
NOTE: >>> RETR baseball.csv
NOTE: <<< 150 Opening ASCII mode data connection for baseball.csv (32303
   bytes)
```

FTP Log Messages (con’t)

```
NOTE: User ftpdownload has connected to FTP server  on Host
   bassettconsulting.com .
NOTE: The infile MYFILE is:
      Filename=baseball.csv,
      Pathname= "/" is current directory,
      Local Host Name=wonderdog,
      Local Host IP addr=192.168.1.100,
      Service Hostname Name=bassettconsulting.com,
      Service IP addr=209.35.121.141,
      Service Name=FTP,Service Portno=21,Lrecl=32767,
      Recfm=Variable

NOTE: <<< 226 Transfer complete.
NOTE: >>> QUIT
NOTE: 322 records were read from the infile MYFILE.
      The minimum record length was 86.
      The maximum record length was 116.
NOTE: The data set WORK.BASEBALL has 322 observations and 22 variables.
NOTE: DATA statement used (Total process time):
      real time           1.67 seconds
      cpu time            0.40 seconds
```

MVS FTP Example

```
/* set the FTP fileref */
filename rsrawdat ftp
"'<data set name>'"
user='<user account>'
host='<IP address>'
prompt rcmd='site rdw' debug ;
```

- why use ***prompt*** option ?

EBCDIC, COB2SAS

- mainframes use EBCDIC characters
- PCs, UNIX use ASCII
- use $ebcdic and S370fpd informats
- use COB2SAS to calculate offsets
- can only use FTP with disk files
- watch out for archiving (HRECALL)

Reading Data From MVS FTP

```
data testraw ;
infile rsrawdat lrecl=1080
missover ;
input
@1 var1 $ebcdic2.
@3 var2 s370fpd5.0
@8 var3 s370fpd4.0
@12 var4 $ebcdic5.
@17 var5 $ebcdic21.
<…more variables read…>
;
run ;
```

URL Access Method

- yes, SAS can read HTML from a URL
- CGI delivers tables on demand
- syntax:

  FILENAME *fileref* **URL**

  '*external-file*' <*url-options*>;
- fileref consists of:

  http://*hostname<:port>/filename*

Reading a URL is the Trick

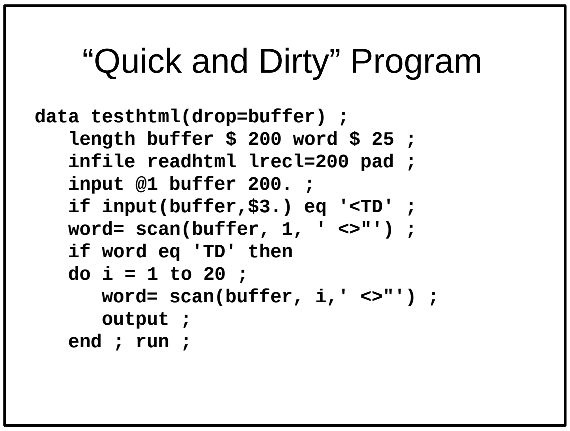
- HTML contains opening, closing tags
- data tables often between <table> tags
- rows are between <tr> tags
- cells (columns) are between <td> tags
- for practice, see sample page at:

bassettconsulting.com/baseball.htm

Reading a URL Assumptions

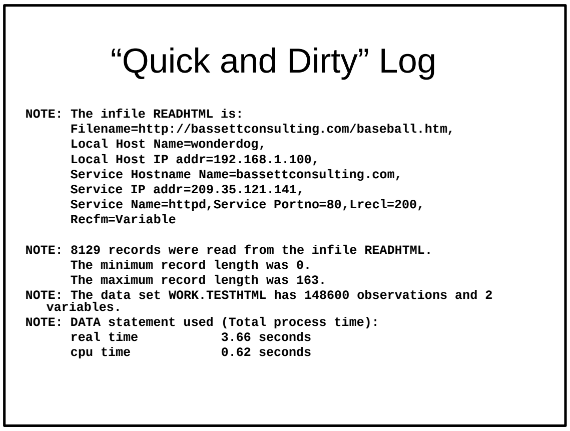
- baseball example created by ODS
- <TD>, <TR> tags at the start of line
- all data cells preceded by font tag

  Color=#000000
- <TD> for character cells contains “ALIGN=LEFT” and “ALIGN=RIGHT” for numeric cells



“Quick and Dirty” Program

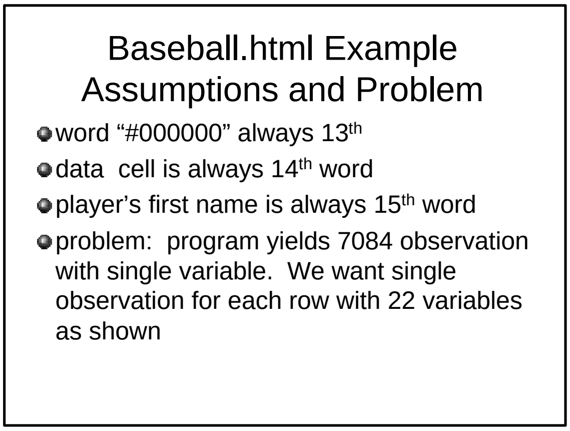
```
data testhtml(drop=buffer) ;
   length buffer $ 200 word $ 25 ;
   infile readhtml lrecl=200 pad ;
   input @1 buffer 200. ;
   if input(buffer,$3.) eq '<TD' ;
   word= scan(buffer, 1, ' <>"') ;
   if word eq 'TD' then
   do i = 1 to 20 ;
      word= scan(buffer, i,' <>"') ;
      output ;
   end ; run ;
```



“Quick and Dirty” Log

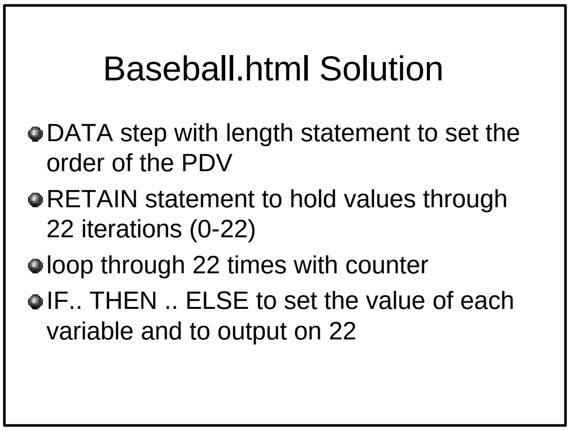
```
NOTE: The infile READHTML is:
      Filename=http://bassettconsulting.com/baseball.htm,
      Local Host Name=wonderdog,
      Local Host IP addr=192.168.1.100,
      Service Hostname Name=bassettconsulting.com,
      Service IP addr=209.35.121.141,
      Service Name=httpd,Service Portno=80,Lrecl=200,
      Recfm=Variable

NOTE: 8129 records were read from the infile READHTML.
      The minimum record length was 0.
      The maximum record length was 163.
NOTE: The data set WORK.TESTHTML has 148600 observations and 2
   variables.
NOTE: DATA statement used (Total process time):
      real time           3.66 seconds
      cpu time            0.62 seconds
```



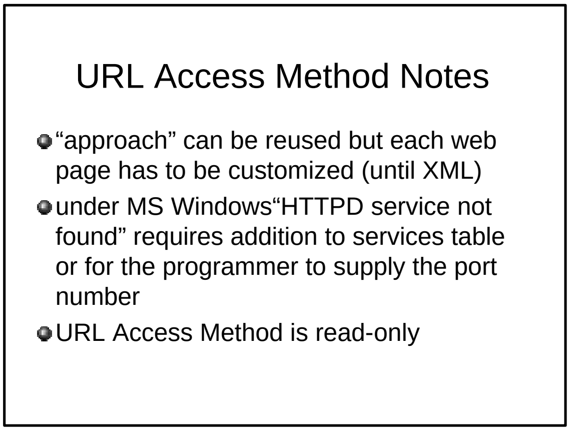
Baseball.html Example Assumptions and Problem

- word “#000000” always 13th
- data cell is always 14th word
- player’s first name is always 15th word
- problem: program yields 7084 observation with single variable. We want single observation for each row with 22 variables as shown



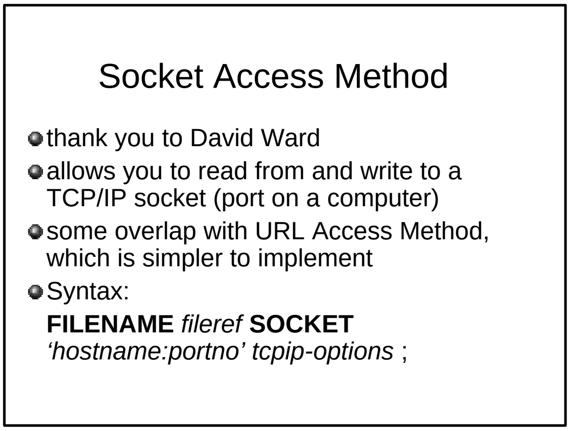
Baseball.html Solution

- DATA step with length statement to set the order of the PDV
- RETAIN statement to hold values through 22 iterations (0-22)
- loop through 22 times with counter
- IF.. THEN .. ELSE to set the value of each variable and to output on 22



URL Access Method Notes

- “approach” can be reused but each web page has to be customized (until XML)
- under MS Windows“HTTPD service not found” requires addition to services table or for the programmer to supply the port number
- URL Access Method is read-only



Socket Access Method

- thank you to David Ward
- allows you to read from and write to a TCP/IP socket (port on a computer)
- some overlap with URL Access Method, which is simpler to implement
- Syntax:

  FILENAME *fileref* **SOCKET** *‘hostname:portno’ tcpip-options* ;

You Can Try This Out, Too! *[filename only]*

```
filename web socket ':80' server
 termstr=CRLF ;
```

Log from Socket Access

```
NOTE: TCP/IP XX Access Method Listen portno is 80.
NOTE: The infile WEB is:
      Local Host Name=wonderdog,
      Local Host IP addr=192.168.1.100,
      Listen Portno=80,Client Hostname,
      Client IP addr,Lrecl=256,Recfm=Variable
```

Socket Access Method Notes

- personal web servers may also use Port 80
- either change the port or temporarily stop the service to run example
- firewalls and other security measures may block socket access
- see David Ward’s paper referenced in the bibliography

CATALOG Access Method

- catalogs are special types of SAS files used to store different types of information in partitions called catalog entries
- this method allows the reading of text information in log, output, and source entries
- syntax:

FILENAME *fileref* **CATALOG** '*catalog*' *<catalog-options>*;

Catalog Four-Part Names

- format: *library.catalog.entry.type*
- some specify just the last two or three parts of a catalog but the author recommends specifying the full four-part names

One More To Try At Home

```
filename dummycat catalog
'work.mycat.dummydat.source' ;
data _null_ ;
   file dummycat ;
   put 'here is some sample data' ;
run ;
data stuff ;
   length buffer $ 20 ;
   infile dummycat ;
   input @1 buffer $20. ;
run ;
proc print; run ;
```

Why Write to Catalogs ?

- some SAS features rely on catalog entries for storage (e.g., SAS/Warehouse Administrator)
- platform independence (slashes)
- scratch entries to WORK get cleaned up automatically
- sometimes there is little choice

Reading Data - Named Pipes

- not just for UNIX !
- can use with Windows
- allows bi-directional exchange of data
- syntax:

FILENAME *fileref* **NAMEPIPE** '*pipe-specification*‘ <*named-pipe-options*>;

Named Pipe Example

- transmitting computer creates pipe named “women”

filename women **namepip**e ‘\\.\pipe\women’ server retry=30;

- receiving computer creates fileref “in”

filename in **namepipe**
‘\\.\pipe\women’ client retry=30;

What Is That Dot ?

- denotes a pipe on a single computer
- use IP address or network name on the receiving computer to create a pipe between computers

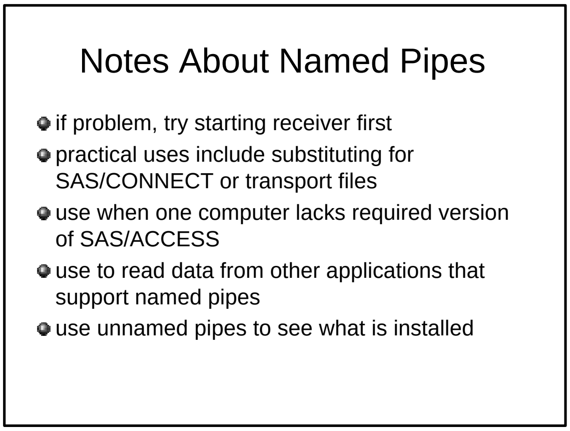
Transmitting Computer Log

```
NOTE: The file WOMEN is:
      Named Pipe Access Device,
      PROCESS=\\.\pipe\women,RECFM=V,LRECL=256

NOTE: 3 records were written to the file WOMEN.
      The minimum record length was 7.
      The maximum record length was 10.
NOTE: The data set WORK.CLASS has 5 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           5.08 seconds
      cpu time            0.04 seconds
```

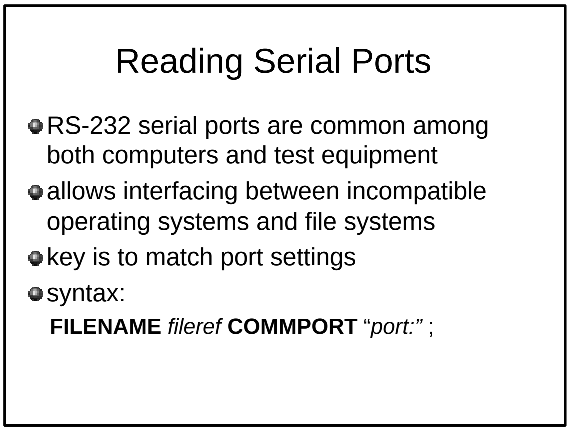
Receiving Computer Log

```
NOTE: 3 records were read from the infile IN.
      The minimum record length was 7.
      The maximum record length was 10.
NOTE: The data set WORK.FEMALE has 3 observations and 2
   variables.
NOTE: DATA statement used (Total process time):
      real time           10.81 seconds
      cpu time            0.12 seconds
NOTE: There were 3 observations read from the data set
   WORK.FEMALE.
NOTE: PROCEDURE PRINT used (Total process time):
      real time           0.56 seconds
      cpu time            0.15 seconds
```



Notes About Named Pipes

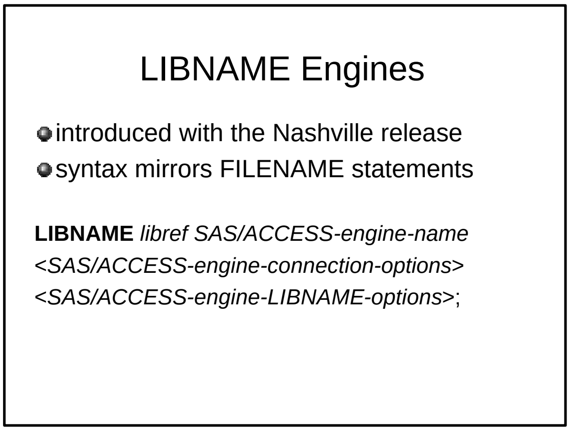
- if problem, try starting receiver first
- practical uses include substituting for SAS/CONNECT or transport files
- use when one computer lacks required version of SAS/ACCESS
- use to read data from other applications that support named pipes
- use unnamed pipes to see what is installed



Reading Serial Ports

- RS-232 serial ports are common among both computers and test equipment
- allows interfacing between incompatible operating systems and file systems
- key is to match port settings
- syntax:

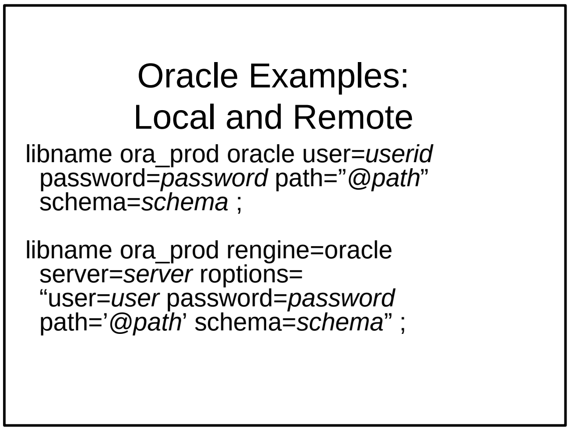
FILENAME *fileref* **COMMPORT** "*port:"* ;



LIBNAME Engines

- introduced with the Nashville release
- syntax mirrors FILENAME statements

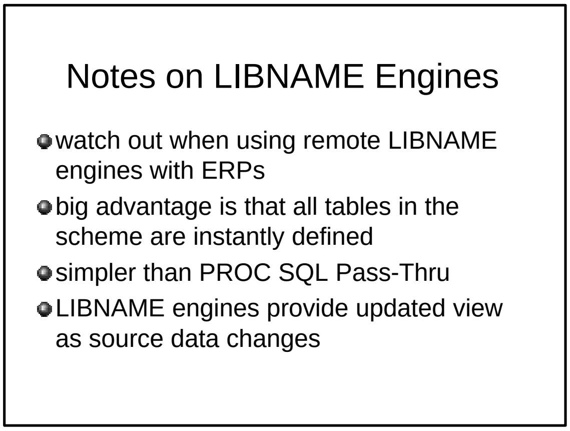
LIBNAME *libref SAS/ACCESS-engine-name*
<*SAS/ACCESS-engine-connection-options*>
<*SAS/ACCESS-engine-LIBNAME-options*>;



Oracle Examples: Local and Remote

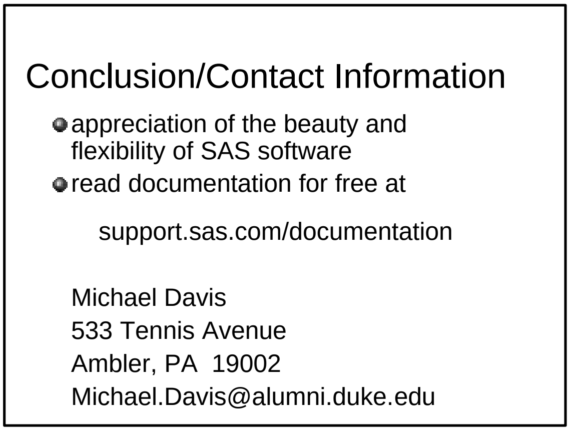
libname ora_prod oracle user=*userid*
password=*password* path="*@path*"
schema=*schema* ;

libname ora_prod rengine=oracle
server=*server* roptions=
"user=*user* password=*password*
path='*@path*' schema=*schema*" ;



Notes on LIBNAME Engines

- watch out when using remote LIBNAME engines with ERPs
- big advantage is that all tables in the scheme are instantly defined
- simpler than PROC SQL Pass-Thru
- LIBNAME engines provide updated view as source data changes



Conclusion/Contact Information

- appreciation of the beauty and flexibility of SAS software
- read documentation for free at

support.sas.com/documentation

Michael Davis
533 Tennis Avenue
Ambler, PA 19002
Michael.Davis@alumni.duke.edu