Hockin J, Lauritsen J
EpiData Association
Denmark

CMD – How to write user commands in EpiData Analysis

The CMD command allows users to execute a series of commands in a .PGM file. This can be very useful when creating reports or when you need to perform the same operations on a series of variables. CMD allows users to create more complex programs, using some of the concepts of traditional computer programming. Although there are few program controls commands (e.g. typical command sequences such as DO … WHILE … END are not available), CMD is quite flexible.

Syntax

```
CMD doxyz
  Command
    …
  Command
END
```

The CMD block (between the CMD command and the following END) may appear anywhere in a .PGM file. Any EpiData command may be used inside the CMD block, though not all will make sense. All variables, including GLOBAL and RESULT variables may be used and you may use @var to insert the current value of var into the command.

The user command will be executed when the name of the CMD appears in the .PGM file. To run the user command once, just put it alone on a line:

```
doxyz
```

To run the use command on each record, you must use the following:

```
if TRUE then doxyz
```

This works because evaluation of the logic IF command occurs for each record.

Examples

1. Rather than use the regular Analysis output from the DESCRIBE command, you wish to print out two lines that include the variable name, and only the mean, the range and coefficient of variation. Putting the calculations and TYPE commands into a CMD block allows you to have only one set of calculations, which is a way to reduce errors and allows you to change your mind about what you want to print out without having to repeat the statements required.

```
* cvcalc command
define zfield __________ global
define zcv ######.##### global
cmd cvcalc
  describe @zfield /q
  zcv = $mean*100.0/$sd
  type "@zfield mean = @$mean Coefficient of variation =
@zcv"
  type "range = (@$min - @$max) "
end
```

* display all statistics

```
define bmi ####.##
bmi = weight*weight/height
describe height weight bmi
* short program to display mean, cv and range for several
variables
zfield = "weight"
cvcalc
zfield = "height"
cvcalc
zfield = "bmi"
cvcalc
```

2. You wish to print several lines per record with only certain fields, for all records entered since a certain date. This could be used to set up labels.

```
* dolabel command
cmd dolabel
  type "@idnum - @name" /h1
  type "@phonenum"
  type "@sex @dob"
  type "Weight=@weight, Height=@height, BMI=@bmi"
end
* print labels for recent records
if datentry>"31/12/2006" then dolabel
```

Advanced programming

A user command may invoke other user commands, including itself (this is called ‘recursion’). This allows some interesting possibilities. Familiarity with simple computer programming of macros or in any programming language will be an asset. Caution: if you create a recursive user command, you must have an IF … THEN … ENDIF block to decide when to stop! The following user command is simply a check that recursion works.

```
* test recursive cmd *
set echo=off
define s # global
s=1
test

cmd test
imif s<1000 then
  s = s + 1
  test
  endif
end

type "All done when s=@s"
set echo=on
```

Limitations

You cannot include the named command on a line with multiple commands:

```
doxyz ; doxyz
```

currently will not work.

Example of file manipulation for all files in subfolders.

Assume you wish to add a variable indicating whether time at visit has a value < 5 (visits before 3rd project year) to all files in a given project. In the project there are 15 participating centres who must supply six data forms, but have several combinations of up to six of additional files. Therefore it is not an easy task to add the variable manually.

Techniques used here are:

- Read the filenames into an array by use of the “dir” command.
- Execute a user defined command for each individual file found in the array
- In the user specified command pass by a given file indicated by name
- Save the file with new structure in a folder for use in the next phase of the project

```
*pgm to add a variable to several working folders:
Define workdir _____________ global
Define fn _____________ global
* Set echo = off         // when it is working we can set
echo to off
Workdir =”Center 1”
Nextone
Workdir =”Center 2”
Nextone
* Quit        // if there are several centres we might want
to quit afterwards
* ..................................................
CMD nextone
   cd @workdir
   mkdir jan2008
   dir "*.rec"
    * repeat command for all files found:
       Loop filename=$FN* do selectrec
END

*.................................................
CMD selectRec
 fn = upper(@filename)
 type “working on : @workdir <br> Current file @fn: ” /h3
 cd @workdir
read "@fn"  /close
 IMIF  ("@fn" <> "ID.REC") tHEN
       var drop x
       gen i x = 0
      if tid > 5 then x = 1
          select x = 1
          sort idtid
      ENDIF
type "writing @fn"
 savedata "jan2008\@fn"   /replace
 close
END
```