

A Brief Overview of EpiData Software

Visit:  at: <http://www.epidata.dk>

To obtain free:  EpiData Entry

 EpiData Analysis

Note: This overview was created using EpiData Entry Version 3.1 (Build 270108) and EpiData Analysis Version V2.2.1. Beware that using an older version may require some changes in commands, while using a more recent version (if available) may add functionality and fixes known bugs.

Before you start:

- Create a C:\EPIDATA_COURSE folder
- Install EpiData Entry
- Install EpiData Analysis

EpiData Entry

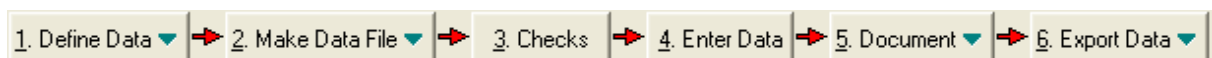
EpiData Entry is used for entering and validating data in a simple and highly efficient format, not available in any proprietary software. It is intuitively appealing and does not have a steep learning curve. Data can be used for analysis in EpiData Analysis or exported for analysis in any proprietary statistical analysis package.

This brief introduction to EpiData Entry introduces the basic principles for creating the necessary data entry files, entering some sample data, double-entering the data, and validating them.

The data entry form: creating a QES file

In the data entry form, we define the structure of the data file, the variables (or fields) and their definition.

We proceed along the process bar:



starting with:



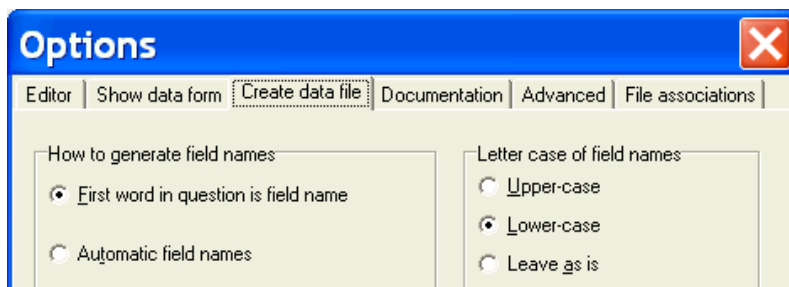
to create a New .QES (questionnaire) file.

Write a title like you would in any text editor (which this is):
A Questionnaire introducing EpiData Entry

There will be five fields (variables) for this exercise:

Field name	Field label	Field type	Field length
Id	Identification code	Text	3
Name	Name of person	Text	15
Dob	Date of birth	Date	10
Age	Age in years	Numeric	3
Sex	Sex of person	Numeric	1

While EpiData Entry is not case-sensitive (except of course for values in text fields), it is good practice to use lower case names for field names as some statistical packages in which data may later be analyzed are case-sensitive (see “File “Options” and tab “Create data file”):



There are two pairs of information sets for a field:

Mandatory	Optional metadata
Field name	Field label
Field definition	Value label

Field Name and **Field Definition** are mandatory: the Field Name defines the name of the variable, the Field Definition defines the characteristics of the field (type of field and field length) to fit the field values that are expected.

Field label and Value Label are so-called meta-data. Like in many other software, there are restrictions in EpiData on the Field name: a field name must be a single word of a length not exceeding 10 characters. Therefore, the field label (metadata) is very useful to provide an a bit more comprehensive description of the field name. For the **Field Definition** preference is given to numeric coding that makes both data entry and data analysis much more efficient. However, if numeric coding is used, it is virtually impossible to enter data error-free or to make sense of the **Field Values**. Therefore, values should always be accompanied by **Value Labels**.

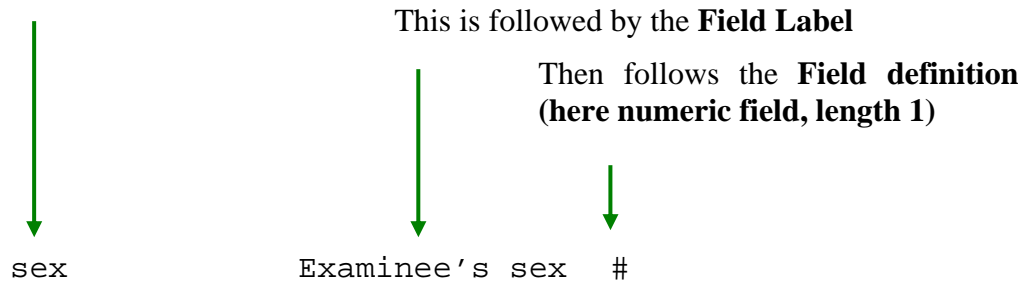
Practically then, there are three components that are written into a data entry form:

Field name

Field label

Field definition

The first word is the **Field Name**



Type the first field name, followed by a space and tell EpiData Entry that it is an upper-case text field of field length 3, then use a hard carriage return and type the second field name, followed by its field label as follows:

```
id Unique identification code <AAA>  
name Person's name
```

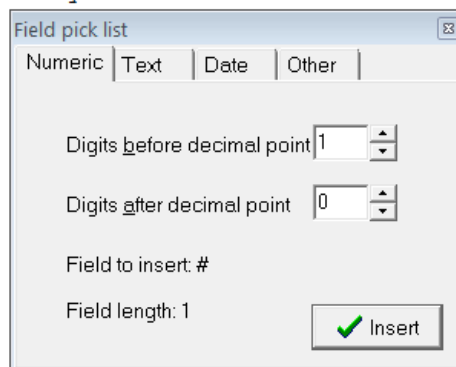
You could do the same for the field NAME as with ID and repeat manually <AAAAAAAAAAAAAAAA>. This is tedious and error-prone (counting the As). Easier is to use the short-cut:

CTRL+Q

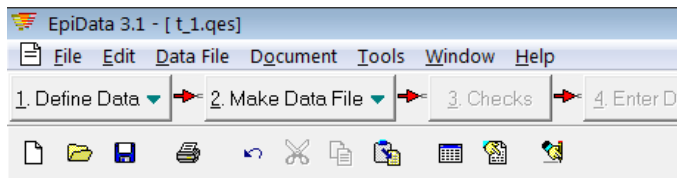
To get:

A questionnaire introducing EpiData Entry

```
id Unique identification code <AAA>  
name Person's name
```



You do not need to exit this screen, just add with its help all the fields until the questionnaire is completed:



```
A questionnaire introducing EpiData Entry

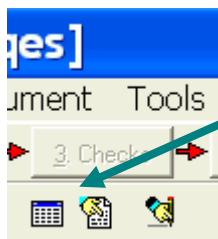
id Unique identification code <AAA>
name Person's name <A >
dob Date of birth <dd/mm/yyyy>
age Person's age in years ##.# on 11 February 2009
sex Person's sex #|
```

and then save the Questionnaire file as:

t_1.qes

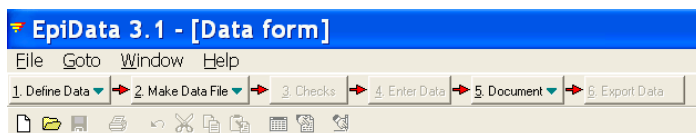
to the C:\EPIDATA_COURSE folder (all work in this project is done in this folder).

We will thus change the **t_1.qes** file. Aligning it and previewing it with the third icon from the right on the lower panel:



Previewing EpiDa

we may get (visualize the layout with **CTRL+T**):



```
A Questionnaire introducing EpiData Entry

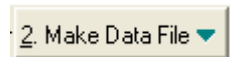
id Identification code 
name Name of person 
dob Date of birth  day-mo-yr format
age Age in years  on 5 February 2006
sex Sex of person 
```

And frequently save (**CTRL+S**).

The data file: creating the REC file

The QES file defined the data structure. This information must be integrated into the actual data file.

Proceed with making a *.REC (data) file:



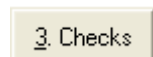
EpiData Entry suggests the same name of the file, but with a different extension, and that is how it has to be, thus create:

t_1.rec

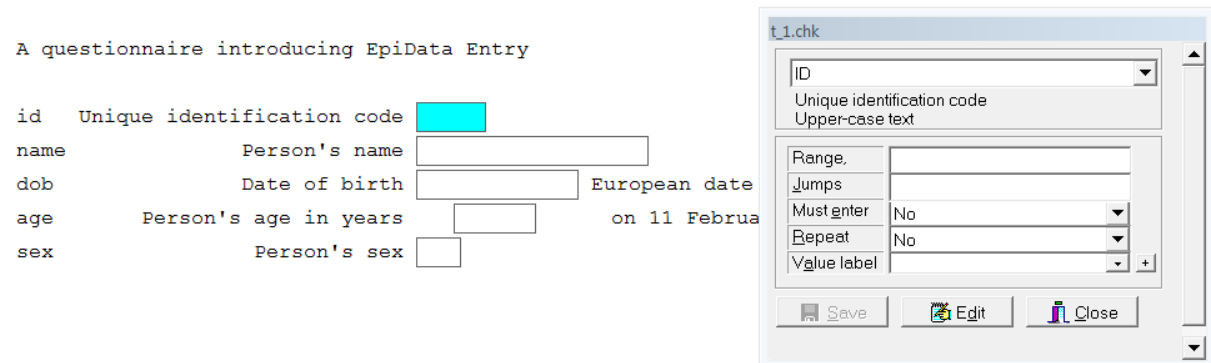
You have created a new and empty data file based on the data structure defined in the questionnaire file. All information from the QES file has now become part of the REC file.

Control data entry: creating the CHK file

While the REC file has everything that defines the data structure, it does not have metadata on numeric values that will be entered nor are there any restrictions on what can be entered into a given field as long as the value is compatible with the field type and field length. This is where the CHK file comes in. A CHK file is always paired with the REC file and ensures data entry control. While you could already enter data now, it is thus best to create the CHK file before you enter any data:

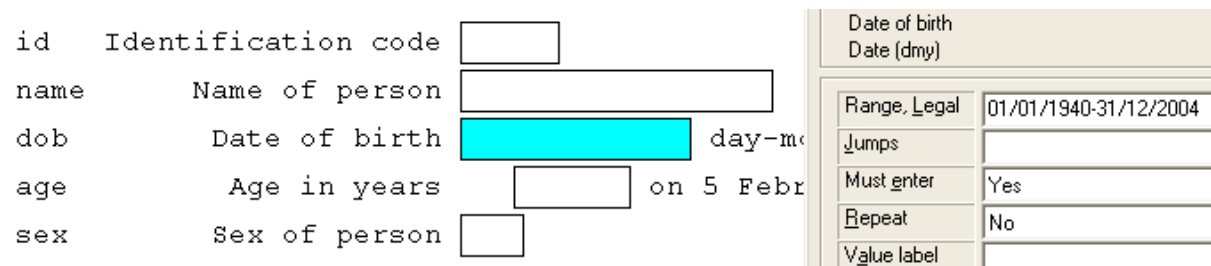


This opens the data entry form and a box to make checks:



Make ID, NAME, DOB, SEX “Must enter” fields. While the field AGE will be calculated by the computer, it should not be a “Must enter” field. Edit it later to “NOENTER”.

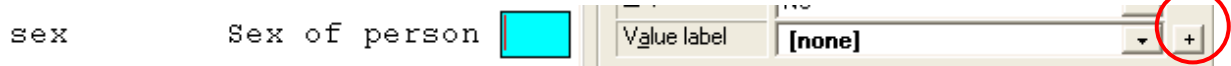
For the field DOB (as an example) you can enter a legal range: it forces the data entry person to use only dates within this range:



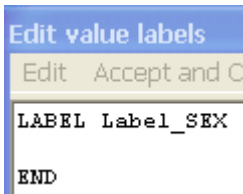
For the field SEX you could enter as Range, Legal:

1, 2, 9

Alternatively (never do both) – and much preferable – you can use a Value label:



by clicking the “+” to the very right and the Check file opens already prepared to proceed:



Just complete it as follows:

```
LABEL label_sex
  1 "Female sex"
  2 "Male sex"
  9 "No sex recorded"
END
```

On the left you enter the legal Field Values, followed by a space and the Field Value Label in quotation marks (quotation marks are not required for single-word Field Value Labels).

Preference is given to numeric coding, using Labelblocks. Use legal values only, if Labelblocks are not possible. This makes data entry efficient.

Accept and close and save the Check file.

You have now three files:

t_1.qes
t_1.rec
t_1.chk

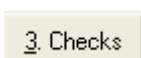
but it is not yet quite finished: nothing has yet been done about the calculation of AGE. In order to do that in the Check file, you have *two options*:

- 1) You use the same approach as above, accessing one field at a time and edit it.
- 2) You open the Check file and make the necessary changes there manually.

Option 1. Make changes using editing Checks from the Process bar

This is the preferred method we will be using here. You can make here all necessary commands that refer to the field in which the focus is. Some do not refer to a particular field, but to the whole record or the entire file. If this type of commands is needed, there is no other way than to use option 2.

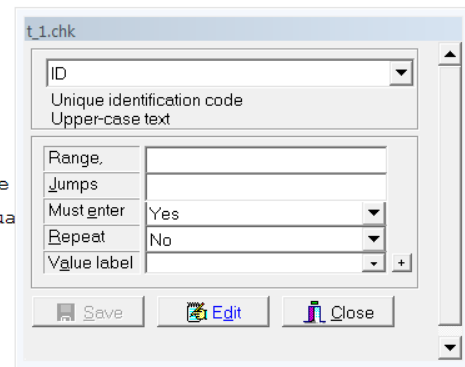
Here we make changes referring to the field on which the focus is, thus choose again:



to get:

A questionnaire introducing EpiData Entry

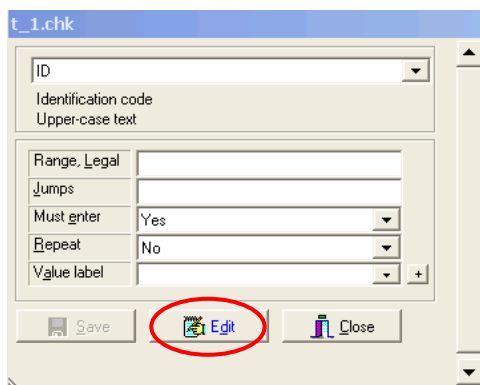
id Unique identification code
name Person's name
dob Date of birth European date
age Person's age in years on 11 Februa
sex Person's sex



what you had just saved before.

The most important variable in any dataset is a unique identifier. You must always be in a position to find a record. Not having a unique identifier is the same as if a person would have no name.

First, we want to ensure that the identifier ID is truly unique for every record. As it is impossible in a large dataset to remember which identifiers had been used before, EpiData Entry should check after entry of an ID whether it had ever been used before and prevent its use if the latter is the case. To this end, ensure that you are in the field ID and click the box Edit (short-cut **ALT+D**):



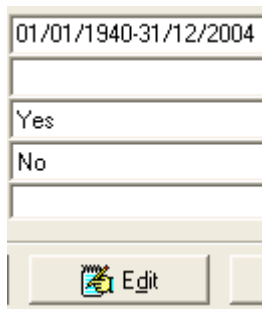
and edit the field as follows:

```
id  
  KEY UNIQUE 1  
  MUSTENTER  
END
```

This will create an index file **t_1.eix** in addition to the three files you already have.

Further down in sequence of the fields, we must the command that will calculate the AGE of the person. This can only be done after the information on DOB has been entered, but must obviously be done before the field AGE can be completed.

If you move the cursor to the field DOB and click the box Edit (short-cut **ALT+D**):



You get the Check file information for just that field:

```
dob
  RANGE 01/01/1940 31/12/2004
  MUSTENTER
END
```

The command that something has to be done after the information has been entered is:

```
AFTER ENTRY
```

and it must be closed with

```
END
```

Leaving an empty line between the command and ending it, we type and get thus:

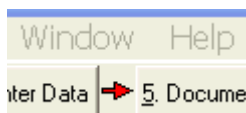
```
dob
  RANGE 01/01/1940 31/12/2004
  MUSTENTER
  AFTER ENTRY
  . . .
  END
END
```

Into the empty line, the calculation for AGE in years on 9 February 2009 is written as follows:

```
age=( "09/02/2009"-DOB)/365.25
```

Why is this so?

Dates in computer software are stored as the interval between the date in question and an internal “anchor date”. Subtracting one date from another will thus also give the number of days between the two dates. To obtain AGE in years, the result of the calculation must accordingly be divided by the average number of days there are in one year. An AFTER ENTRY command must be followed by an END command. To learn more about Check file commands go to:



Access Help with **Alt+H** or mouse

where you'll find everything you need and much more.

If you understand why, then it is clear why the completed block must be as follows:

```

dob
  RANGE 01/01/1940 31/12/2004
  MUSTENTER
  AFTER ENTRY
  age=( "05/02/2006"-DOB)/365.25
  END
END

```

After Accept and Close, the next field is AGE to Edit. It is a NOENTER field:

```

age
  NOENTER
END

```

“MUSTENTER” is not what is needed. To the contrary, it is preferred that a data entry person does not even get into this field as it is calculated and should not be changed inadvertently. Thus, the existing block is changed to get:

```

age
  NOENTER
END

```

Finally, the field SEX:

```

sex
  COMMENT LEGAL USE label_sex
  MUSTENTER
END

```

specifies which LABEL is to be used, but without additional commands, this label is not displayed. In addition, it is also nice to have the value label written to the right of the field during data entry to verify that the value that was just entered really corresponds to the value label that was intended. The block is thus appended to obtain:

```

sex
  COMMENT LEGAL USE label_sex SHOW
  TYPE COMMENT
  MUSTENTER
END

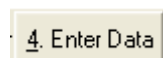
```

Information: The Check file has only information on fields to which a command applies.

Now you are all set to go, save and close the file.

Entering data

Proceed now to:



The following are the data on the three individuals that you should enter into the

t_1.rec

file:

ID	NAME	DOB	SEX
AA1	Hans	20/09/1950	Male
AB2	Hoa	03/01/1980	Female
AC3	Ahmed	12/10/1973	Male

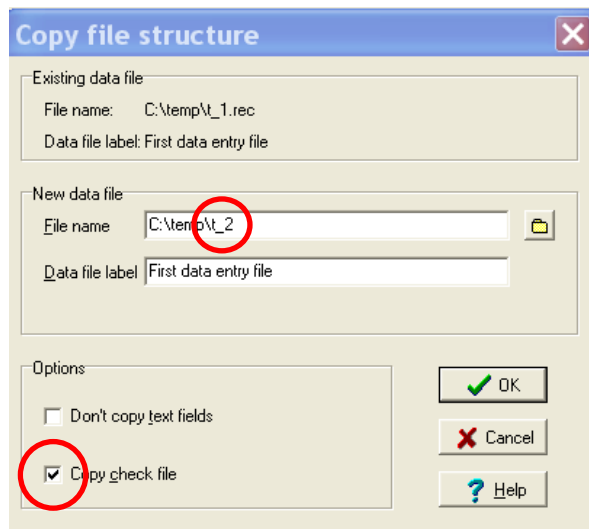
Note that you are prompted right after the last entry in each record to save to disk. Therefore you never lose any information with EpiData Entry should you get a software freeze or another problem forcing you to terminate the program, except the very one record on which you had been working.

Data validation

How can you know that the data in your computer are an accurate electronic copy of the above paper copy? One likes to assume that no data entry errors were made, but that is not how humans work. There is no point in proceeding to a more or less sophisticated analysis with such data else you may end up with a “garbage in – garbage out” situation getting you into the awkward situation to defend data quality rather than the interpretation of the findings.

The only way to ensure that the electronic data file corresponds precisely to the paper copy is to enter the data a second time, compare the two files and then check the discordances against the original paper copy and correct the errors.

In the “Tools” menu you find “Copy structure”, choose it and use the **t_1.rec** file to get:



where you have to add **t_2** to name the second file. Ensure that the “Copy check file” is ticked (default setting) as the second data entry should use exactly the same Check file.

Enter the data a second time into the:

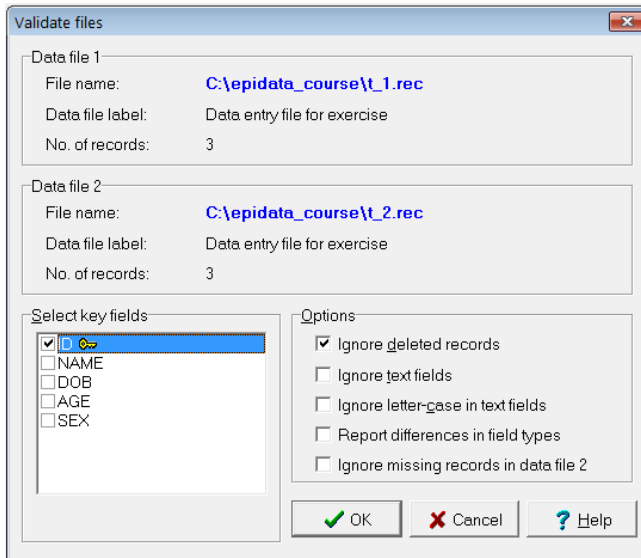
t_2.rec

file.

To make a comparison of the two files use:

5. Document ▼

and select “Validate duplicate files” and edit the opening window appropriately, using the field ID as the key field on which to compare:



and you will get a validation report with a lot of information of which only the part with the list of errors is shown here:

```
-----  
DATA FILE 1 | DATA FILE 2  
-----  
Record key field(s): (Rec. # 1) | Record # 1  
id = AA1 |  
|  
dob = 20/09/1950 | dob = 20/08/1950  
age = 58.4 | age = 58.5  
-----  
Record key field(s): (Rec. # 3) | Record # 3  
id = AC3 |  
|  
name = AHMED | name = AMEHD  
-----
```

Save this file as:

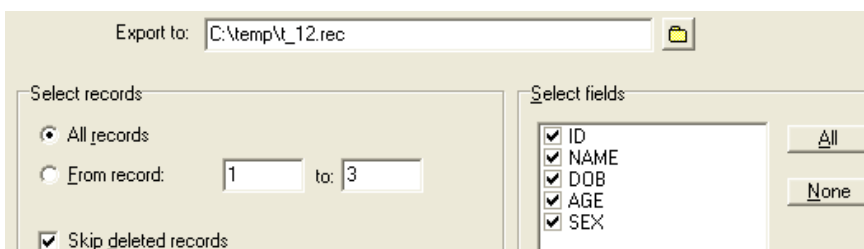
t_12.not

Then use:

6. Export Data ▼

for one of the original files (e.g. **t_1.rec**) to EpiData, giving the data file the new name

t_12.rec

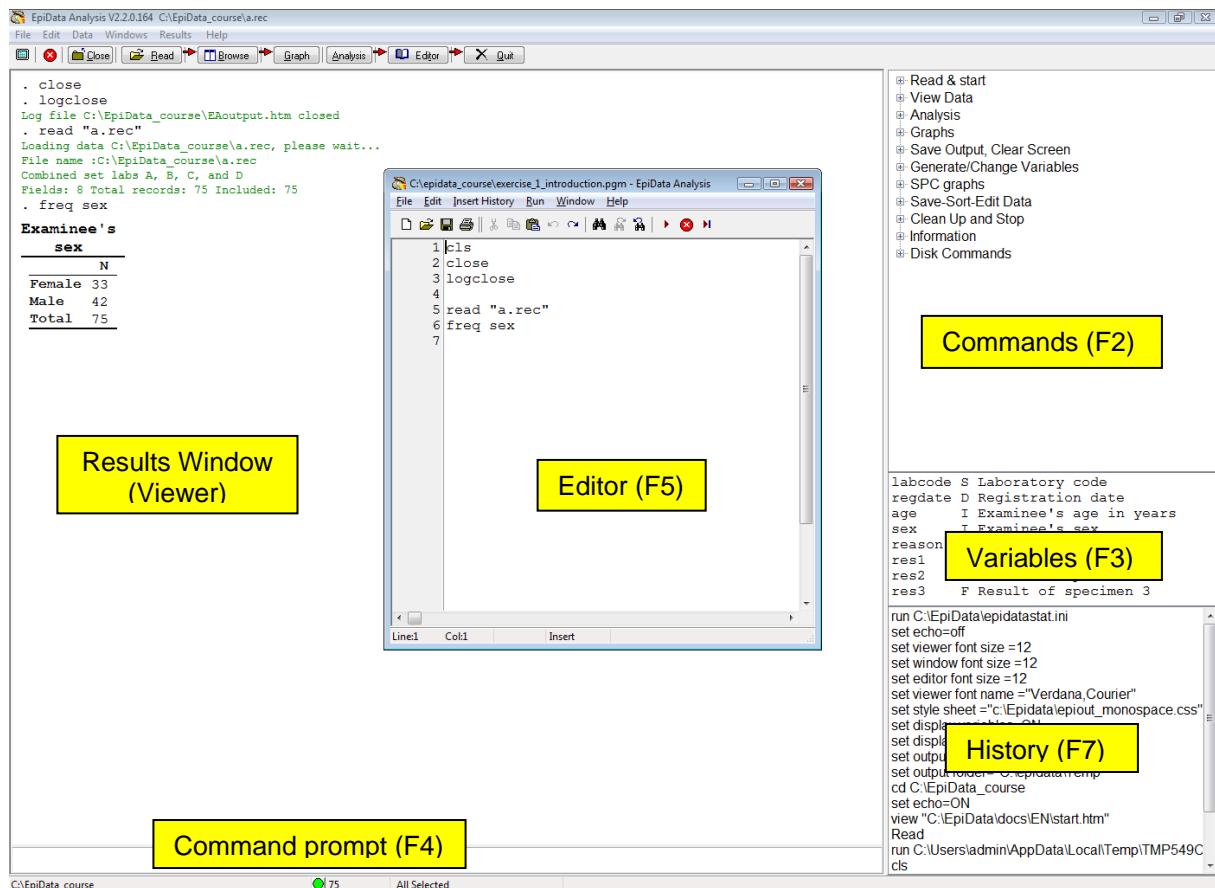


You did get a Check file **t_12.chk** when exporting, but a small bug changed KEY UNIQUE to KEY 1 and the NOENTER may have become a MUSTENTER by error. Although not strictly necessary anymore for this file, it might be preferable to correct it:

```
id
  KEY UNIQUE 1
  NOENTER
END
```

This brief introduction cannot show the full power of using Check commands. You may wish to look up in the Help file the many possibilities you have. As you are learning progressively, you will be able to use more and more of these commands that will make your data entry forms more and more efficient.

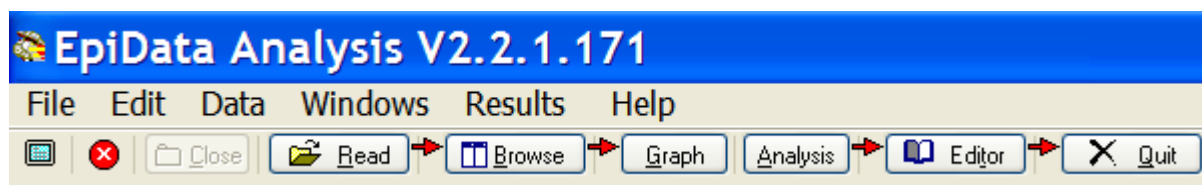
EpiData Analysis



EpiData Analysis is used to analyze EpiData *.rec files, dBase *.dbf files, and text *.csv files. You find the following four files included:

- a.rec**
- b.rec**
- c.rec**
- d.rec**

When you open EpiData Analysis, you see a familiar menu and a process bar:



For beginners, it is best to use initially the process bar.

There is no data file active, it is opened by using:



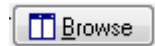
Choose the file:

a.rec

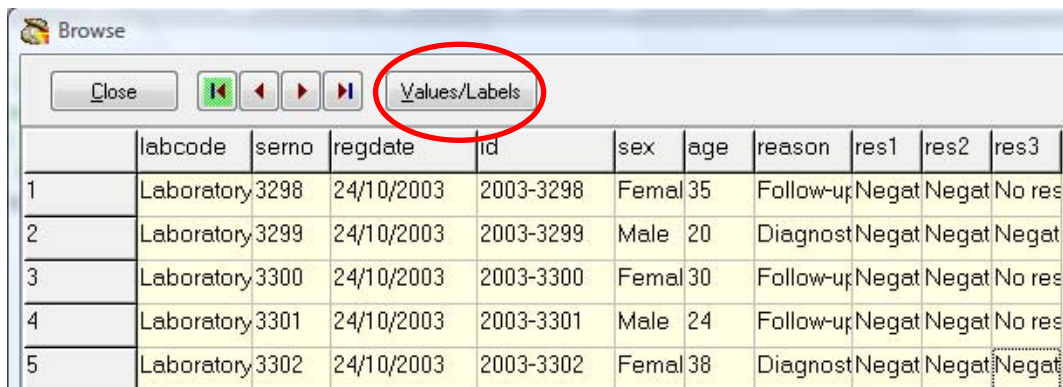
from the C:\EPIDATA_COURSE folder. In the Viewer window you should then get:

```
. read
Loading data C:\epidata_course\a.rec, please wait...
File name :C:\epidata_course\a.rec
Combined set labs A, B, C, and D
Fields: 10 Total records: 75 Included: 75
```

Use:

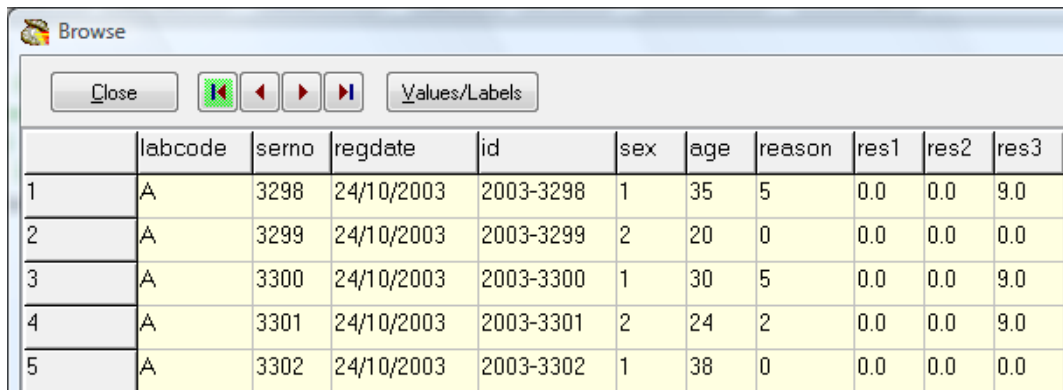


to get a line listing:

A screenshot of the "Browse" window. The "Values/Labels" button is circled in red. The table below shows the field labels for the data.

	labcode	serno	regdate	id	sex	age	reason	res1	res2	res3
1	Laboratory	3298	24/10/2003	2003-3298	Femal	35	Follow-up	Negat	Negat	No res
2	Laboratory	3299	24/10/2003	2003-3299	Male	20	Diagnost	Negat	Negat	Negat
3	Laboratory	3300	24/10/2003	2003-3300	Femal	30	Follow-up	Negat	Negat	No res
4	Laboratory	3301	24/10/2003	2003-3301	Male	24	Follow-up	Negat	Negat	No res
5	Laboratory	3302	24/10/2003	2003-3302	Femal	38	Diagnost	Negat	Negat	Negat

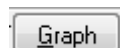
Note the “switch” Values/Labels. If you click on it you get:

A screenshot of the "Browse" window. The "Values/Labels" button is now disabled. The table below shows the field values for the data.

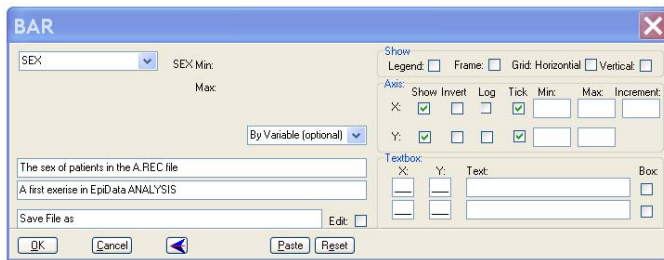
	labcode	serno	regdate	id	sex	age	reason	res1	res2	res3
1	A	3298	24/10/2003	2003-3298	1	35	5	0.0	0.0	9.0
2	A	3299	24/10/2003	2003-3299	2	20	0	0.0	0.0	0.0
3	A	3300	24/10/2003	2003-3300	1	30	5	0.0	0.0	9.0
4	A	3301	24/10/2003	2003-3301	2	24	2	0.0	0.0	9.0
5	A	3302	24/10/2003	2003-3302	1	38	0	0.0	0.0	0.0

You can thus BROWSE either the Field Values or the Field Labels (it requires of course that the CHK file associated with the REC file is in the same folder).

Use:

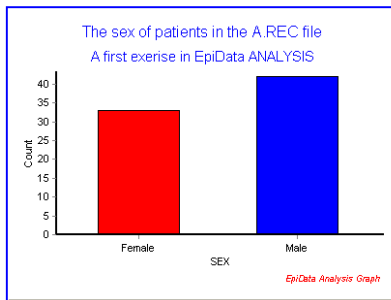


to make for instance a bar graph (or any other):

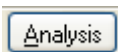


Choose the X Variable (SEX in this example) and modify the lines for Title and Sub-Title if you wish, then click OK to get:

```
. BAR SEX /TI="The sex of patients in the A.REC file" /SUB="A first exercise in EpiData ANALYSIS"
```



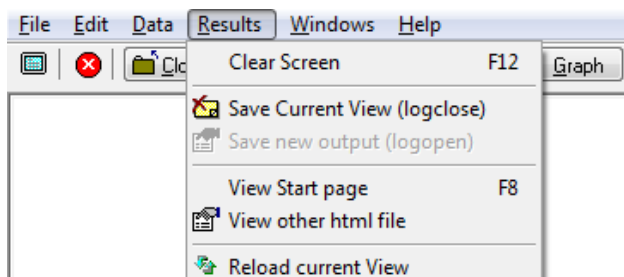
Use:



To, e.g. DESCRIBE the content of a numeric field such as AGE and get:

Variable	N=75	Sum	Mean	(95% cfi)	Min	p5	p10	p25	Median	p75	p90	p95	Max
AGE	75	2997.00	39.96	35.35 44.57	14.00	16.60	19.00	24.00	35.00	50.00	72.20	81.00	99.00

In the Results menu, you can clear the screen (and memory):



to clear the Viewer output with “Clear screen” (Note that function key **F12** accomplishes the same thing).

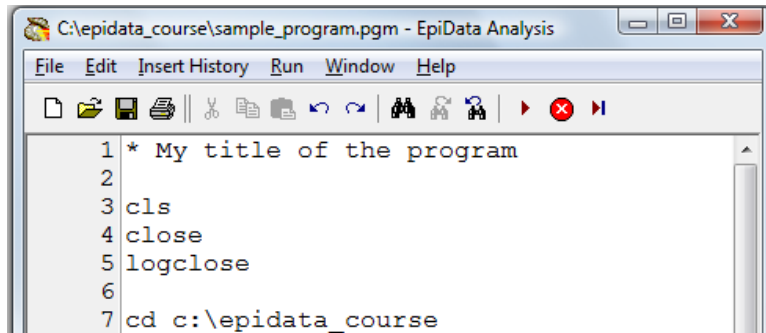
This is all good and fine what might be called “interactive analysis”. The advantage is that you can very quickly “fish” in your dataset and familiarize yourself with it. The disadvantage of interactive analysis is that everything is lost once you leave EpiData Analysis. It defies thus the underwritten purpose that everything you do in research must be thoroughly documented.

The solution of EpiData Analysis to documentation is to use:



(Function key **F5** also opens the Editor)

It opens a new window which is simply a text editor for EpiData Analysis programs that take the extension *.PGM. There are many different ways to do it, but things become more quickly automated if done always in the same way. The proposal is that the first few command lines have the following content:



```
C:\epidata_course\sample_program.pgm - EpiData Analysis
File Edit Insert History Run Window Help
[Icons]
1 * My title of the program
2
3 cls
4 close
5 logclose
6
7 cd c:\epidata_course
```

If the first character on a line is an asterisk *, then EpiData Analysis will not execute the line. You thus use

* to be followed by a comment to remind yourself

cls clears the screen and removes information from the memory and makes a program run faster.

close closes any open data file which is necessary as you should and cannot have two data files open at the same time.

logclose closes any open log file: a log file records what you are currently doing. As you start from scratch at the beginning of a program, no log file should be open.

cd c:\epidata_course tells EpiData in what directory it should work to get files and to place any output.

Save the program as e.g.:

t_1.pgm

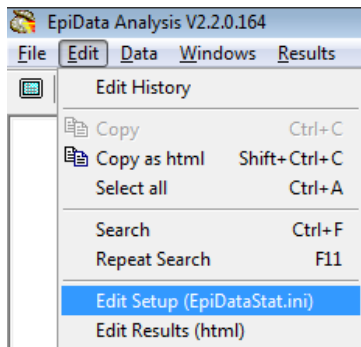
Using relative paths

Above we gave the example of an *absolute path*:

```
cd c:\epidata_course
```

If we give this program to another persons, that person will only be able to run the program without error message if that person has the same folder (c:\epidata_course) on his or her computer. Much more convenient would be a way that that the program would not contain such a command and could thus be run in any folder. In other words, the user would define where the data are rather than the program dictating where they have to be.

In the menu you see “Edit” with a sub-menu:



The line now marked in blue points to the EpiDataStat.ini file. Whenever EpiData Analysis is accessed, it has the built-in instruction to look for precisely this file and anything written in that file is executed first without the user even noticing it. This file is entirely editable. When you click on it, it will open in the editor and be somewhat different from what we show below, because the file shown here has already been edited:

```

1 * EpiData Analysis default settings file
2 * Edit the next lines to change size or font
3
4 set echo=off
5 * Viewer font and size:
6 set viewer font size =12
7 set window font size =12
8 set editor font size =12
9 set viewer font name =\"Verdana,Courier\"
10 set style sheet =\"c:\Epidata\epiout_monospace.css\"
11
12 set display variables=ON
13 set display databrowser=OFF
14 set output open=ON
15
16 set output folder=\"C:\epidata\Temp\"
17 cd C:\EpiData_course
18 * cd c:\temp
19 set echo=ON
20
21

```

In lines 6 to 8 you see that the font size of various windows has been defined.

In line 9 you see that the font type has been defined

In line 17 you see that the course folder has been defined.

In line 18 you see that another folder has been defined but the asterisk in front of it changes it to a comment and it is thus not executed. This is very useful and strongly recommended that for any project you carry out, you define the folder in which the data records are and in which the work will be done is defined here. This allows then that you do no more need to define this every time in your programs and that your programs can be used by anyone in whatever folder that person's work is done.

In the material you have the four files:

a.rec

b.rec

c.rec

d.rec

They have all exactly the same structure. If you like to APPEND the four files into one single file, extend your program to look as follows:

```
1 * Append REC files
2
3 cls
4 close
5 logclose
6
7 cd c:\epidata_course
8
9 read "a.rec"
10 append /file="b.rec"
11 append /file="c.rec"
12 append /file="d.rec"
13 savedata "abcd.rec" /replace
14 close
15
16 read "abcd.rec"
```

Note the following here:

- File names must always be in quotation marks
- A newly created data file is saved with the command `SAVEDATA` followed by the name of the new data file. Unless you tell EpiData Analysis explicitly that overwriting is permitted, you will get an error message. In case the file exists, the option to allow overwriting is `/REPLACE` following the `SAVEDATA` command. This will simultaneously replace both the data (REC) and the Check file that are created.
- After this command, the **a.rec** file is the open data file: it must be closed before the new file **abcd.rec** can be opened.

You can **Run** the program from the menu or simply with **F9** and the last lines in the Viewer window are:

```
Loading data C:\epidata_course\abcd.rec, please wait...
File name :C:\epidata_course\abcd.rec
Combined set labs A, B, C, and D
Fields: 10 Total records: 300 Included: 300
```

F3

is a toggle key which displays (or turns off displaying) the fields in this dataset:

```
.....
labcode S Laboratory code
.....
serno    I Laboratory serial number
regdate  D Registration date
id       S Unique identifier
sex      I Examinee's sex
age      I Years of age
reason   I Examination reason
res1     F Result of first examination
res2     F Result of second examination
res3     F Result of third examination
```

F5

opens again the Editor window and

F4

gets you to the command line where you could type:

```
freq res3
```

to get a frequency of the values for the field RES3 which encodes the results of the third sputum smear examination:

Result of third examination	
	N
Negative	107
1+ positive	13
2+ positive	7
3+ positive	3
No result recorded	170
Total	300

Note here that the Value labels are displayed here, not the values. When you created the **abcd.rec** file, a Check file **abcd.chk** was created from the Check files of the source data files.

To display both values and value labels use the option:

```
/vl
```

("1" as in "lambda"), thus:

```
freq res3 /vl
```

to get:

Result of third examination	
	N
0.0 Negative	107
1.0 1+ positive	13
2.0 2+ positive	7
3.0 3+ positive	3
9.0 No result recorded	170
Total	300

or

```
tables sex reason
```

to get:

Examinee's sex			
Examination reason	Female	Male	Total
Diagnostic examination	52	82	134
Follow-up at 2 months	18	22	40
Follow-up at 3 months	4	8	12
Follow-up at 4 months	0	3	3
Follow-up at 5 months	13	22	35
Follow-up at 6 months	11	20	31
Follow-up at 7 months or later	3	13	16
Follow-up examination - no time	8	19	27
No reason recorded	0	2	2
Total	109	191	300

Within the same program, you can create a new variable defining a CASE. If you define a CASE as an examinee with at least one positive result among the three and then make a table by sex, you would append your program as follows:

```

15 close
16 read "abcd.rec"
17
18 define case #
19             let case=0
20 if res1>0 and res1<9 then case=1
21 if res2>0 and res2<9 then case=1
22 if res3>0 and res3<9 then case=1
23 label case "Definition of case"
24 labelvalue case /0="Non-case" /1="Case"
25
26 tables case sex /r
27

```

As the **abcd.rec** file has already been created, it is not necessary to run again the whole program with **F9**, it suffices to mark the program section (as above) to start with a clean slate (close) and then run that partial with **F8**. This comes in handy when one builds up a complex program and verifies its functionality progressively.

Note the commands:

```

label case "Definition of case"
labelvalue case /0="Non-case" /1="Case"

```

These create value labels for the new variable CASE and a field label for it.

You get:

Definition of case				
Examinee's sex	Non-case	% Case	% Total	%
Female	94 (86.2)	15 (13.8)	109 (100.0)	
Male	168 (88.0)	23 (12.0)	191 (100.0)	
Total	262 (87.3)	38 (12.7)	300	

Percents: (Row)

Note the option for the `tables` command that was used:

```
/r
```

EpiData Analysis uses an approach similar to Stata in that you get only what you ask for. Without the option `/r`:

```
tables case sex
```

you get:

Definition of case			
Examinee's sex	Non-case	Case	Total
Female	94	15	109
Male	168	23	191
Total	262	38	300

thus no row percentages. For more options, such as column percentages and total percentages you would just add the options you wish:

```
tables case sex /r /c /tp
```

to get:

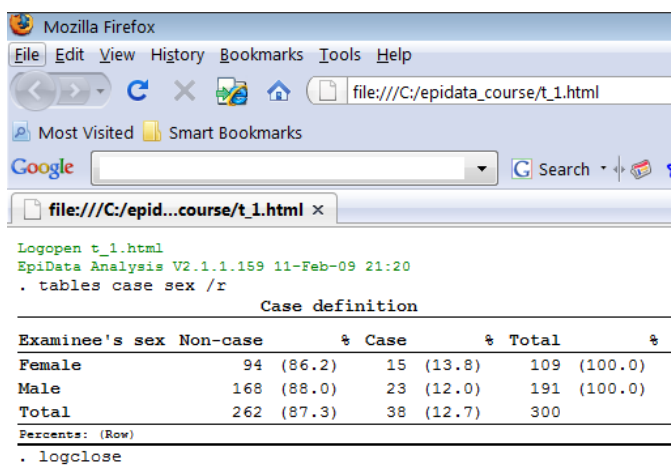
Definition of case									
Examinee's sex	Non-case	% Case				% Total			
Female	94 (86.2) {35.9} [31.3]	15 (13.8) {39.5} [5.0]	109 (100.0) {36.3} [36.3]						
Male	168 (88.0) {64.1} [56.0]	23 (12.0) {60.5} [7.7]	191 (100.0) {63.7} [63.7]						
Total	262 (87.3) {100.0} [87.3]	38 (12.7) {100.0} [12.7]	300						

Percents: (Row) {Col} [Total]

To write the table into a file, change the program as follows:

```
logopen "t_1.html" /replace
tables case sex /r
logclose
```

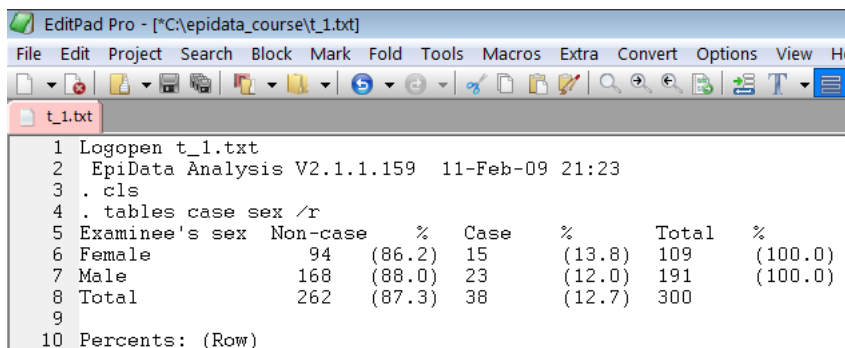
HTML is the Web's language and if you double-click the `t_1.html` file it will open in your default browser:



You may also save the table as a text file:

```
28 logopen "t_1.txt" /replace
29 tables case sex /r
30 logclose
```

that opens in your default text editor:



Any of these output formats can be pasted into your spreadsheet application if needed.

Using EpiData Analysis graphics capabilities

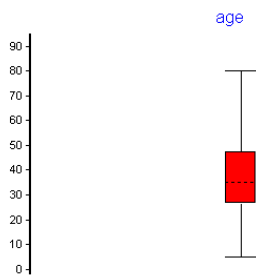
EpiData Analysis offers various graphics capabilities. We introduced how you can use graphs simply with the menus. Often, one desires to use several of the available options though. It is thus easiest to put the commands into the program and then add options as you go along testing the effect of each.

A simple sample graph without options for the field AGE reads as follows:

```
select age <>99
```

```
boxplot age
```

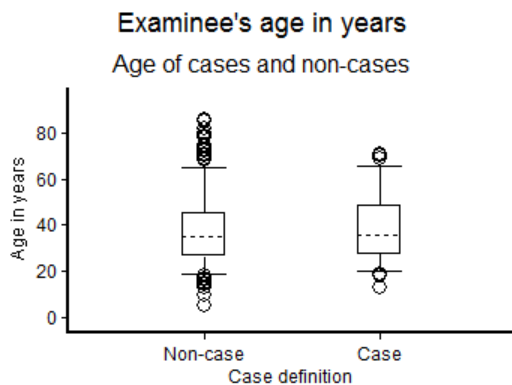
and you get



Perhaps you wish to introduce a stratification by CASE, label the axes and get a legend, control the text color, show outliers, define the length of the whiskers, and so on. These and many more options are available (see Help file). Your expanded program reads then for instance:

```
cls
select age<>99
set option graph /sizex=400 /sizey=600
boxplot age /by=case /out /bw \
  /ymin=0 /ymax=93 \
  /ytext="Age in years" \
  /sub="Age of cases and non-cases" \
  /p1090 \
  /fn=" "
```

and you get:



For a confidence interval graph CIPILOT without options to obtain the proportion of cases among by WHO standard age groups, you would define a new variable creating age groups from AGE and type:

```
define agegrp #
if age>=00 and age<15 then agegrp=1
if age>=15 and age<25 then agegrp=2
if age>=25 and age<35 then agegrp=3
if age>=35 and age<45 then agegrp=4
if age>=45 and age<55 then agegrp=5
if age>=55 and age<65 then agegrp=6
if age>=65 and age<99 then agegrp=7
if age =99 then agegrp=9
label agegrp "WHO standard age groups"
labelvalue agegrp /1="0- to 14-yr-old"
labelvalue agegrp /2="15- to 24-yr-old"
labelvalue agegrp /3="25- to 34-yr-old"
labelvalue agegrp /4="35- to 44-yr-old"
labelvalue agegrp /5="45- to 54-yr-old"
labelvalue agegrp /6="55- to 64-yr-old"
labelvalue agegrp /7="65-yr-old and older"
labelvalue agegrp /9="Unknown age"

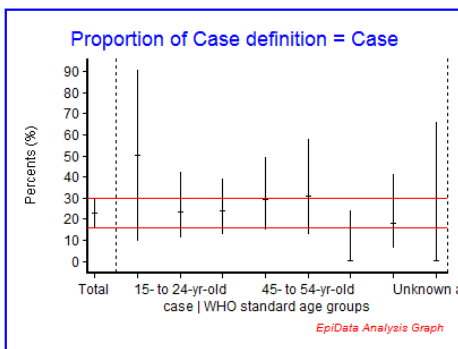
select
select reason=0
ciplot case agegrp
```

and get:

Crude: Proportion of Case definition = Case among all.

variable	stratum	Total N	n _{Case definition=Case}	%	(95% CI)
Case definition	Total	134	30	22.4	(16.2-30.2)
WHO standard age groups	0- to 14-yr-old	2	1	50.0	(9.5-90.5)
	15- to 24-yr-old	26	6	23.1	(11.0-42.1)
	25- to 34-yr-old	38	9	23.7	(13.0-39.2)
	35- to 44-yr-old	24	7	29.2	(14.9-49.2)
	45- to 54-yr-old	13	4	30.8	(12.7-57.6)
	55- to 64-yr-old	12	0	0.0	(0.0-24.3)
	65-yr-old and older	17	3	17.6	(6.2-41.0)
	Unknown age	2	0	0.0	(0.0-65.8)

Crude: Proportion of Case definition = Case among all.



which could of course be customized with a multitude of options.