

Part D. More on EpiData software

Part D: More on EpiData software

Exercise 1: Relational database and aggregating vs from “Long-to-wide”

Exercise 2: A statistical process control chart

Exercise 3: A simplified survival analysis

Exercise 4: Creating a menu for standard reports

Introductory note

Part D will address operationally relevant concepts in data collection and data analysis:

- How do we deal with a situation, where each individual has a varying number of observations?
- How do we determine statistically relevant deviations from a proportion over an observation period when the denominator varies with each time subset over the observation period?
- What is survival analysis and how to deal with it in EpiData Analysis?
- How to make a menu-driven, HTML-based EpiData Analysis interface to run standard reports?

Exercise 1: A relational database and “Aggregating” vs from “long-to-wide”

At the end of this exercise you should be able to:

- a. Create a relational database for a varying number of observations
- b. Merge a child file to the parent file
- c. Recognizing when to use “Aggregate” and when to transpose data
- d. Transpose multiple observations (columns) into a single record (row)

Not all laboratories keep their registers as The Union and WHO recommend for the Tuberculosis Laboratory Register, where 1 line corresponds to 1 examinee rather than to 1 examination. In fact, many laboratories enter the results for each examination on one line. If such an approach is chosen, we may find a register as follows:

Patient	Date of exam	Sex	Marital status	Blood sugar	Sputum	Result
A	24-Mar-2007	Male	Married	6.3	Muroid	1+
B	24-Mar-2007	Male	Divorced	4.9	Muco-purulent	Neg
C	24-Mar-2007	Female	Single	5.2	Purulent	Neg
D	24-Mar-2007	Female	Widowed	7.3	Blood-tinged	2 per 100
A	25-Mar-2007	Male		7.3	Salivary	Neg
D	25-Mar-2007	Female		7.4	Muroid	2+
A	26-Mar-2007			7.2	Purulent	1+
C	26-Mar-2007	Female		4.8	Muco-purulent	Neg
E	27-Mar-2007	Male	Married	8.2		1+
F	27-Mar-2007	Female	Annulled	7.4	Purulent	Neg
G	27-Mar-2007	Male	Cohabiting	6.9	Salivary	Neg
G	28-Mar-2007	Male		7.2	Muroid	2+
E	28-Mar-2007	Male		7.9	Purulent	2+
F	31-Mar-2007	Female		7.2	Muco-purulent	3+
H	31-Mar-2007		Married	6.6	Muroid	Neg
I	31-Mar-2007	Male	Separated	8.3	Salivary	Neg
H	1-Apr-2007	Female		6.9	Muco-purulent	1+
F	1-Apr-2007	Female	Engaged	7.7	Purulent	2+
I	1-Apr-2007	Male	Single	8.0	Muroid	8 per 100
G	1-Apr-2007	Male		7.6	Muco-purulent	1+
K	2-Apr-2007	Female	Married	4.5	Purulent	Neg
I	2-Apr-2007	Male		8.2	Muco-purulent	
H	2-Apr-2007	Female		6.6	Muroid	1+
I	3-Apr-2007	Male		8.1	Muroid	1+

This type of a register requires a different approach to both data entry and data analysis than we used before. Two important things need to be considered:

- 1) The same patient may appear again and again on sequential dates

2) Not every patient has the same number of visits

Some patient characteristics do not change over time such as, in this example, the identity of the patient, sex, and marital status (well, perhaps not during these short intervals). Others do change, such as the date of examination, blood sugar, the aspect of the sputum and the sputum smear examination result.

To capture such information in a single data entry form as was done in Part A would be very inconvenient: 1) one would have to anticipate the maximum of allowable visits, and 2) if one patient has a single visit, one would still have to complete all fields with the codes for missing values up to the maximum allotted if we insist that all fields must be MUSTENTER fields.

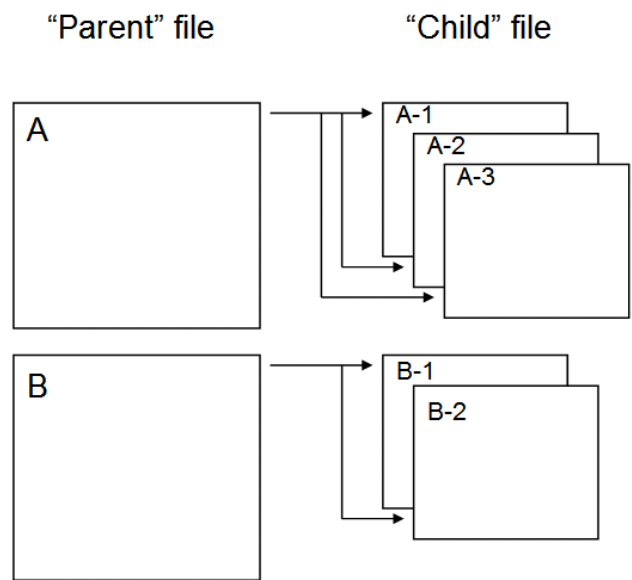
Rule: *If an individual has a single observation for each variable or a fixed number of observations for each variable, then a single EpiData Entry triplet is the best solution. If an individual has a variable number of observations for each variable, the choice is a relational data base.*

Building a relational database requires determining which information is static for an individual and which changes over repeated observations.

EpiData Entry

Introduction

What is the structure of such a relational database? The figure below outlines a relational database with two levels.



At the Level 1 (the "Parent" file), we may have patients, denoted here with A and B. At Level 2 (the "Child" file), denoted here as A-1, A-2, and A-3, may reflect different visits to a clinic where each time the blood sugar, blood pressure, and weight are measured. Each

individual may have at least one, but up to an unlimited number of such visits, and the number of visits varies for each individual.

Note: Because of the simplicity, it is always preferable to have a single data entry form.

However, if the data available concern an individual with fixed information (e.g. age, sex, etc) on one hand, and variable information (e.g. serial examinations) and there are fixed sets of serial examinations (e.g. repeated measures of blood sugar and blood pressure) and each examinee has a varying number of examinations, then it may become more efficient to use a relational database.

In the following, we will call the records at level 1 (the parent file) records on **Examinees** (a person) and the records at level 2 records on **Examinations**.

For building the relational database, we want to collect different information on each level and link the two levels, so that at some point during entering information for an examinee a trigger will open the file to enter records on examinations for the specific examinee. The information for one or more records about examinations is entered until one wishes to return back to the field following the trigger at the upper level.

Important components for the structuring of the relation between the two EpiData Entry triplets are:

- The two files have a common identifier field. In the Check file this field must be KEY UNIQUE at Level 1 (Examinees are unique) and KEY at Level 2 for examinations (where there might be multiple records for the same examinee). This identifier might be called IDPERSON.
- The Examination file must have its own unique identifier, which might be called IDEXAM.
- At an appropriate point in the Check file at Level 1 (Examinee) a RELATE command must be entered, making the relation to Level 2 (examinations).
- After returning from Level 2 back up to Level 1, one must arrive “somewhere” and it will be the next field after the field from which the diversion occurred. Thus, the field with the RELATE command at Level 1 should not be the last field of the Level 1 (examinee) record.

Level 1: Check file of the examinee :	Level 2: Check file of the examination :
1_examinee.rec	2_examination.rec
idperson MUSTENTER KEY UNIQUE 1 END	
var01 RELATE idperson 2_examination.rec END	Idperson NOENTER KEY 1 END
	Idexam MUSTENTER

	KEY UNIQUE
	END
varlast	

If there is a single record to be entered in the child record, then the RELATE command in the parent record must indicate so with 1 after the name of the child file:

```
RELATE idperson 2_examination.rec 1
```

If there is frequently only non-variable information on the examinee, but no variable examination information, one can introduce a command so that there is not automatically a deviation to the file on examinations:

```
var01
  MUSTENTER
  AFTER ENTRY
  HELP "Is there information on examinations?\n y: yes\n n: no" KEYS="YN"
  TYPE=CONFIRMATION
  IF RESULTLETTER="Y" THEN
    RELATE idparent 2_examination.rec
  ENDIF
  END
END

Varlast
  MUSTENTER
  END
```

Note the use of KEYS and the command RESULTLETTER (see more about it in the Help file searching for RESULTLETTER).

To return from the lower to the upper level, use the key F10 (there are other possibilities) and best write this into the data entry form.

As always, start with a *data documentation sheet* for the fields that will be used for the above dataset before you start working in EpiData Entry. You must define which fields belong to the “parent” file (defined as the file that has one-time information for the individual) and which fields will belong to the “child” file (defined as the related file containing changing information for the individual).

The working example

You will create two final EpiData Entry triplets with the following names:

```
d_ex01_patient.qes
d_ex01_patient.rec
d_ex01_patient.chk

d_ex01_examination.qes
d_ex01_examination.rec
d_ex01_examination.chk
```

Note the following:

- 1) The identifier that relates the two files must be KEY UNIQUE in the parent file and KEY in the child file.

- 2) The RELATE command should follow after entering the value for the second to last field in the parent file.

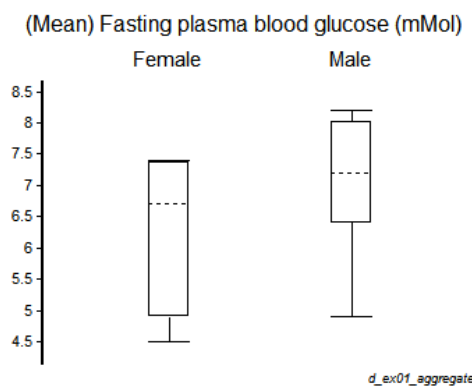
EpiData Entry will automatically ask if you enter the same identifier for a second time (KEY UNIQUE) whether you want to see the record where this identifier was used before. Upon approval, you will get to it and you can then add the information for the additional examination starting from that record in the parent file.

You will note during data entry that this requires a lot of concentration and errors are easily made. It will thus be critical to enter the dataset twice to allow subsequent validation. As you will have to validate each of the two related files independently, you must also have a unique identifier not only in the parent but also in the child file to allow validation on that key.

EpiData Analysis

The final result of the analysis will be to obtain the following EpiData Analysis output:

Part 1 with AGGREGATE



Part 2 with transposing values

Overall microscopy result			
of 4 serial smears	Negative	Positive	Total
N—	2	0	2
NN—	1	0	1
NN9P	0	1	1
NP—	0	1	1
NPP-	0	3	3
PNP-	0	1	1
PP—	0	1	1
Total	3	7	10

Patient's sex				
Incremental yield of first 3 smears	Female	% Male	% Total	%
NNP	0 (0.0)	1 (25.0)	1 (14.3)	
NPx	3 (100.0)	1 (25.0)	4 (57.1)	
Px	0 (0.0)	2 (50.0)	2 (28.6)	
Total	3 (100.0)	4 (100.0)	7	

Percents: (Col)

It doesn't really look like much. Nevertheless, quite a few steps are needed to get from the source files D_EX01_PATIENT.REC and D_EX01_EXAMINATION.REC to this point. We will elaborate on some considerations you have to make and offer hints for the sequential components in EpiData Analysis.

Merging the files

In EpiData Entry you made a relation through a unique identifier from the parent to the child file. In EpiData Analysis you must now merge these to files to get the following dataset (first 13 records only shown):

	patid	examid	dateexam	bs	sputum	micro	sex	marital	MergeVar	exam
1	A	A-3-24	24/03/2007	6.3	1	1.0	2	5	3	1
2	A	A-3-25	25/03/2007	7.3	5	0.0	2	5	3	2
3	A	A-3-26	26/03/2007	7.2	2	1.0	2	5	3	3
4	B	B-3-24	24/03/2007	4.9	3	0.0	2	3	3	1
5	C	C-3-24	24/03/2007	5.2	2	0.0	1	7	3	1
6	C	C-3-26	26/03/2007	4.8	3	0.0	1	7	3	2
7	D	D-3-24	24/03/2007	7.3	4	0.2	1	8	3	1
8	D	D-3-25	25/03/2007	7.4	1	2.0	1	8	3	2
9	E	E-3-27	27/03/2007	8.2	9	1.0	2	5	3	1
10	E	E-3-28	28/03/2007	7.9	2	2.0	2	5	3	2
11	F	F-3-27	27/03/2007	7.4	2	0.0	1	4	3	1
12	F	F-3-31	31/03/2007	7.2	3	3.0	1	4	3	2
13	F	F-4-1	01/04/2007	7.7	2	2.0	1	4	3	3

where each record is an observation from the child file, to which the information from the parent file is repetitively added for each observation for the same individual. In other words, you start from the other way around than in EpiData Entry, starting in EpiData Analysis with the child file and using the parent as a lookup table. The grammar of the commands to accomplish this is:

```
read "childfile.rec"
merge parentidentifier /file="parentfile.rec" /table
```

It is important that following merging, the data are sorted first by the individual identifier, and then within that identifier by the date of the visit.

Note: We commonly suggested to code unknown dates as "01/01/1800". With sorting, the unknown date will thus come first (sorting is ascending by default) and this might not be desirable. In this dataset we don't have unknown dates. But only because the small dataset allows us to see that, you would have to anticipate that possibility with a larger dataset and thus first make a new date variable where the unknowns take a value for a date in the future, so they appear with sorting at the end of the information on an individual.

Following the sorting, it might be advantageous to number the visits in order to be able to make a frequency on them to see what the maximum number of visits is (here we can see that it is a maximum of 4 visits, but in a large database, it would be more difficult).

To create a new variable EXAM and set its default initially to 1 (each record takes first the value 1), we have so far always used the following grammar:

```
define exam #
exam=1
```

EpiData Analysis offers a more elegant one-line alternative approach which accomplishes exactly the same thing:

```
gen i exam=1
```

The command GEN replaces DEFINE and "i" stands for an integer field.

Note: the command GEN will produce integer of length of 9. If you need a shorter and fixed field length integer, you must utilize DEFINE.

There are other similar commands:

```
gen f doorheight=1.85
gen d birthdate=date("31/12/1899")
gen s firstname="john"
```

for date fields (d) float (real number) fields (f), and string (text) fields (s). Look it up in the help file (type `gen+F1` in the command line).

Now that you have a new variable EXAM, how can you tell EpiData that it should look at the person (identified by an ID) and number each visit, starting with 1 until the next individual comes, when it must start again with 1. The command is:

```
if id=id[_n-1] then exam=exam[_n-1]+1
```

This looks admittedly complex. Let's thus take it apart. `[_n]` identifies the current record and accordingly `[_n-1]` the immediately preceding record. Let's say, EpiData Analysis has proceeded to record 547 and looks at the ID of this record. It looks whether record 546 had the same ID as record number 547: `if id=id[_n-1]`. If that is the case, then it should take the EXAM number of record 546 and add 1 to it for record 547: `exam=exam[_n-1]+1`. If it is not the case, then the default stays (which we defined as 1) and it moves on to the next record.

As a result, you should get:

	patid	dateexam	exam
1	A	24/03/2007	1
2	A	25/03/2007	2
3	A	26/03/2007	3
4	B	24/03/2007	1
5	C	24/03/2007	1
6	C	26/03/2007	2
7	D	24/03/2007	1
8	D	25/03/2007	2
9	E	27/03/2007	1
10	E	28/03/2007	2

the first variable column showing the identifier, the second the date of the examination, and the third the number of the examination for that individual.

Aggregating data

If we look at the sex (given the field name SEX), date of examination (given the field name DATEEXAM), and blood sugar (given the field name BS):

	patid	sex	dateexam	bs
1	A	2	24/03/2007	6.3
2	A	2	25/03/2007	7.3
3	A	2	26/03/2007	7.2
4	B	2	24/03/2007	4.9
5	C	1	24/03/2007	5.2
6	C	1	26/03/2007	4.8

SEX remains obviously the same (and is thus from the parent file), and is a categorical variable unique to the examined person, while BS varies by examination date and is a continuous variable. If we want to examine blood sugar by sex, there is no need to transpose the blood sugar values from the vertical to the horizontal as EpiData Analysis has inbuilt a tool to aggregate the data. For the individual and its sex we would write:

```
aggregate patid sex
```

and get (for all ten individuals):

	patid	sex
1	A	2
2	B	2
3	C	1
4	D	1
5	E	2
6	F	1
7	G	2
8	H	1
9	I	2
10	K	1

We can expand this command using an option to calculate the MEAN of blood sugar for each individual:

```
aggregate patid sex /mean="bs"
```

and get:

	patid	sex	N	Nbs	MEAbs
1	A	2	3	3	6.9
2	B	2	1	1	4.9
3	C	1	2	2	5.0
4	D	1	2	2	7.4
5	E	2	2	2	8.0
6	F	1	3	3	7.4
7	G	2	3	3	7.2
8	H	1	3	3	6.7
9	I	2	4	4	8.1
10	K	1	1	1	4.5

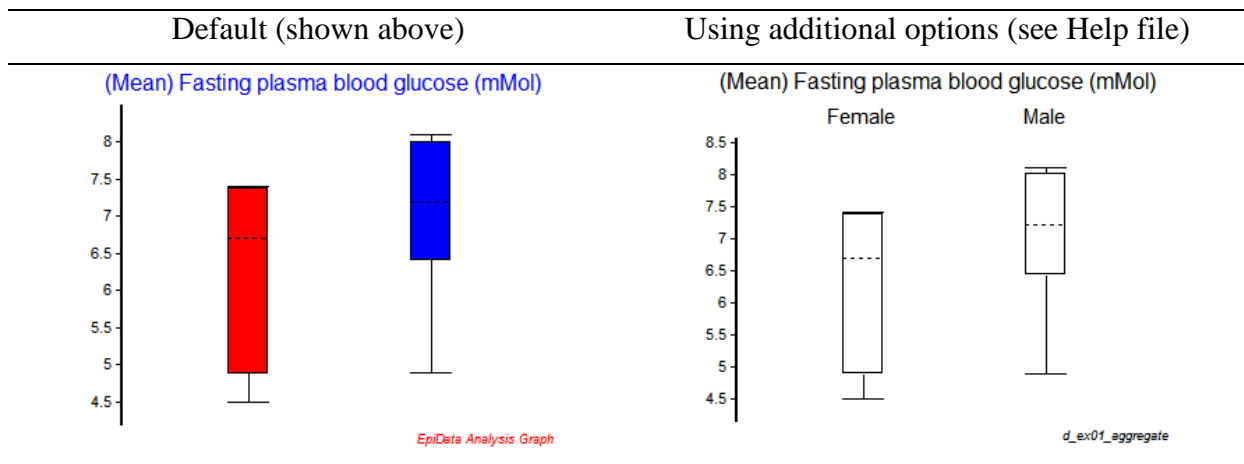
where MEAbs is the calculated mean value of blood sugar for an individual from all the individual's measurements. To save the aggregate data in a file, we add another option:

```
aggregate patid sex /mean="bs" /save="d_ex01_aggregate.rec" /replace
```

Having accomplished this, it is now straight forward to write:

```
cls
close
read "d_ex01_aggregate.rec"
boxplot meabs /by=sex
```

and get:



From long-to-wide

For showing means, there is thus no need to transpose data from the vertical to the horizontal. But it is different for the macroscopic aspect of sputum the examination results. We do not want (nor would we get a sensible result) aggregate sputum smear results. We wish to know each result from each individual.

The first thing before starting copying results from the vertical to the horizontal, we need to know the maximum number of examinations an individual had in the data set. We can get this with a frequency on the EXAM:

```
freq exam
```

Number of examination	
	N
1	10
2	8
3	5
4	1
Total	24

This shows that the maximum number of examinations an individual had in this dataset was 4. We need thus to prepare 4 new variables for each field that are in the sequence of the examination in the vertical but should also become part of each record.

Let's say we have a variable VAR1 with different values for each visit:

ID	VISIT	VAR1
B1	1	1
B1	2	3
B1	3	2
B1	4	2
C1	1	3
D1	1	2

What we need is:

ID	VISIT	VAR1	VAR11	VAR12	VAR13	VAR14
B1	1	1				
B1	2	3				
B1	3	6				
B1	4	2				
C1	1	3				
D1	1	2				

First, we make these 4 variables and give them all the default value of -1 (the minus one is a good way to see missing data and helps later in the selection):

```
gen i var11=-1
gen i var12=-1
gen i var13=-1
gen i var14=-1
```

After these four command lines, our above dataset becomes:

ID	VISIT	VAR1	VAR11	VAR12	VAR13	VAR14
B1	1	1	-1	-1	-1	-1
B1	2	3	-1	-1	-1	-1
B1	3	6	-1	-1	-1	-1
B1	4	2	-1	-1	-1	-1
C1	1	3	-1	-1	-1	-1
D1	1	2	-1	-1	-1	-1

and we are ready to copy the values from the vertical to the horizontal by respecting that the value of VAR1 from VISIT 1 goes to VAR11, the value from VISIT 2 to VAR12, etc.

For VISIT 1, the value for VAR11 is equal to the value of VAR1 and we make it thus the default:

```
VAR11=VAR1
```

Then we use the same approach as above to identify the record:

```
if (id[_n])=(id[_n+1]) then var12=var1[_n+1]
```

This means that if the current record [_n] has the same ID as the next record [_n+1], then VAR12 in the current record should take the value of VAR1 from the next record [_n+1].

Now we do this for all possible 4 records (the maximum of VISITS):

```
if (id[_n])=(id[_n+2]) then var13=var1[_n+2]
if (id[_n])=(id[_n+3]) then var14=var1[_n+3]
```

All the lines to be written for this original field VAR1 are thus:

```
gen i var11=-1
gen i var12=-1
```

```

gen i var13=-1
gen i var14=-1
VAR11=VAR1
if (id[_n])=(id[_n+1]) then var12=var1[_n+1]
if (id[_n])=(id[_n+2]) then var13=var1[_n+2]
if (id[_n])=(id[_n+3]) then var14=var1[_n+3]

```

and we get (assuming that the last patient D1 had only 1 VISIT):

ID	VISIT	VAR1	VAR11	VAR12	VAR13	VAR14
B1	1	1	1	3	6	2
B1	2	3	1	3	6	-1
B1	3	6	1	3	-1	-1
B1	4	2	1	-1	-1	-1
C1	1	3	3	-1	-1	-1
D1	1	2	2	-1	-1	-1

You may note that only for VISIT 1 for patient B1 with four visits all new 4 variables have all respective 4 values from the 4 VISITs.

Now we can safely get rid of the records of VISITs 2, 3, and 4 and we end up just with individuals who have all information from each VISIT:

```
select visit=1
```

and we get:

ID	VISIT	VAR1	VAR11	VAR12	VAR13	VAR14
B1	1	1	1	3	6	2
C1	1	3	3	-1	-1	-1
D1	1	2	2	-1	-1	-1

The conversion from “Long-to-wide” is thus successfully completed, well, for this variable. Of course, before you make this selection, you have to repeat the same approach for each field, so that in the end you get something like:

	patid	sex	marital	dateexam1	dateexam2	dateexam3	dateexam4	sputum1	sputum2	sputum3	sputum4	res1	res2	res3	res4
1	A	2	5	24/03/2007	25/03/2007	26/03/2007	01/01/1900	1	5	2	-1	10	0	10	-1
2	B	2	3	24/03/2007	01/01/1900	01/01/1900	01/01/1900	3	-1	-1	-1	0	-1	-1	-1
3	C	1	7	24/03/2007	26/03/2007	01/01/1900	01/01/1900	2	3	-1	-1	0	0	-1	-1
4	D	1	8	24/03/2007	25/03/2007	01/01/1900	01/01/1900	4	1	-1	-1	0	20	-1	-1
5	E	2	5	27/03/2007	28/03/2007	01/01/1900	01/01/1900	9	2	-1	-1	10	20	-1	-1
6	F	1	4	27/03/2007	31/03/2007	01/04/2007	01/01/1900	2	3	2	-1	0	30	20	-1
7	G	2	2	27/03/2007	28/03/2007	01/04/2007	01/01/1900	5	1	3	-1	0	20	10	-1
8	H	1	5	31/03/2007	01/04/2007	02/04/2007	01/01/1900	1	3	1	-1	0	10	10	-1
9	I	2	7	31/03/2007	01/04/2007	02/04/2007	03/04/2007	5	1	3	1	0	0	90	10
10	K	1	5	02/04/2007	01/01/1900	01/01/1900	01/01/1900	2	-1	-1	-1	0	-1	-1	-1

You have now ten patients and you are at the point where you can continue to work in the same way as you used to work before. While it is much more complex to get to here from 1 line per examination than from 1 line per examinee, it is also obvious that in the end this is much more informative:

For each examination we have the date and for each specimen we have the quality of the sputum. In the Union / WHO approach you have only one date (the date of collection of the first specimen) for a series of three, and the quality of sputum in the Tuberculosis Laboratory is not that informative as it is very possible that every day the quality of the specimen is different, but there is no space allocated to write 3 different ones.

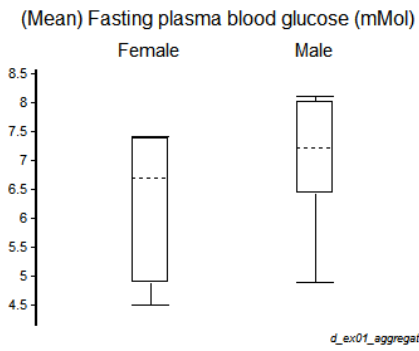
Tasks:

- *Prepare a data documentation sheet*
- *Prepare two EpiData Entry triplets to create a relational database*
- *Enter the data from the following sample data set:*

Patient	Date of exam	Sex	Marital status	Blood sugar	Sputum	Result
A	24-Mar-2007	Male	Married	6.3	Mucoid	1+
B	24-Mar-2007	Male	Divorced	4.9	Muco-purulent	Neg
C	24-Mar-2007	Female	Single	5.2	Purulent	Neg
D	24-Mar-2007	Female	Widowed	7.3	Blood-tinged	2 per 100
A	25-Mar-2007	Male		7.3	Salivary	Neg
D	25-Mar-2007	Female		7.4	Mucoid	2+
A	26-Mar-2007			7.2	Purulent	1+
C	26-Mar-2007	Female		4.8	Muco-purulent	Neg
E	27-Mar-2007	Male	Married	8.2		1+
F	27-Mar-2007	Female	Annulled	7.4	Purulent	Neg
G	27-Mar-2007	Male	Cohabiting	6.9	Salivary	Neg
G	28-Mar-2007	Male		7.2	Mucoid	2+
E	28-Mar-2007	Male		7.9	Purulent	2+
F	31-Mar-2007	Female		7.2	Muco-purulent	3+
H	31-Mar-2007		Married	6.6	Mucoid	Neg
I	31-Mar-2007	Male	Separated	8.3	Salivary	Neg
H	1-Apr-2007	Female		6.9	Muco-purulent	1+
F	1-Apr-2007	Female	Engaged	7.7	Purulent	2+
I	1-Apr-2007	Male	Single	8.0	Mucoid	8 per 100
G	1-Apr-2007	Male		7.6	Muco-purulent	1+
K	2-Apr-2007	Female	Married	4.5	Purulent	Neg
I	2-Apr-2007	Male		8.2	Muco-purulent	
H	2-Apr-2007	Female		6.6	Mucoid	1+
I	3-Apr-2007	Male		8.1	Mucoid	1+

- *Write a program D_EX01.PGM that merges the two files, then prepare sets for the aggregated data and for the “long-to-wide” transformation to produce the following output (obtaining the same numbers is relevant) respectively (see next page):*

From aggregating the data:



From transformation “long-to-wide”:

Overall microscopy result			
of 4 serial smears	Negative	Positive	Total
N—	2	0	2
NN—	1	0	1
NN9P	0	1	1
NP—	0	1	1
NPP—	0	3	3
PNP—	0	1	1
PP—	0	1	1
Total	3	7	10

Patient's sex				
Incremental yield of first 3 smears	Female	% Male	% Total	%
NNP	0 (0.0)	1 (25.0)	1 (14.3)	
NPx	3 (100.0)	1 (25.0)	4 (57.1)	
Px	0 (0.0)	2 (50.0)	2 (28.6)	
Total	3 (100.0)	4 (100.0)	7	

Percent: (Col)

Solution to Exercise 1: A relational database and “Aggregating” vs from “long-to-wide”

Key points:

- A relational database is the solution to a varying number of observations per individual
- The child file is merged with the parent file to give a dataset of all observations
- To obtain means for an individual from continuous variables, aggregating the data is the strategy of choice
- To reduce the dataset to individuals with information on each examination, one must copy the information from observations in the vertical to newly created fields in the first observation of each individual before selecting that record from the individual (“long-to-wide”)

Task

- *Prepare a data documentation sheet*

The documentation sheet is shown on the next page.

Task

- *Prepare two EpiData Entry triplets to create a relational database*

The data entry forms may be made as follows:

D_EX01_PATIENT.QES

Data entry form: Patient information

patid	Unique patient identifier	<input type="text"/>
sex	Patient's sex	<input type="text"/>
marital	Patient's marital status	<input type="text"/>

D_EX01_EXAMINATION.QES

Data entry form: Examination information

patid	Unique patient identifier	<input type="text"/>
examid	Unique examination identifier	<input type="text"/>
dateexam	Date of examination	<input type="text"/> Enter 01/01/1800 if missing
bs	Fasting plasma blood glucose (mMol)	<input type="text"/> Enter 99.9 if missing
sputum	Macroscopic sputum aspect	<input type="text"/>
micro	Microscopy result	<input type="text"/>

The data documentation sheet:

Data documentation sheet

	Field name	Field label	Field type	Field length	Field values	Value label	Field comment
Patient file	patid	Unique patient identifier	U	1	A,....,Z		Any given unique ID
	sex	Patient's sex	I	1		1 Female 2 Male 3 Unknown	
	marital	Patient's marital status	I	1		1 Annulled 2 Cohabiting 3 Divorced 4 Engaged 5 Married 6 Separated 7 Single 8 Widowed 9 Unknown	

NOTE: If SEX is given during an earlier examination and left empty in a subsequent examination, keep information from earlier
 If SEX is different in different examinations, record as UNKNOWN
 If MARITAL is given during an earlier examination and left empty in a subsequent examination, keep information from earlier
 If MARITAL is different in different examinations, update to most recent information

Examination file	examid	Unique examination identifier	S	12	A-2007-01-31,...		Automatically calculated
	datexam	Date of examination	dd/mm/yyyy	10	01/01/2007,....,31/12/2007 01/01/1800		Legal visit date recordings Enter if visit date is missing
	bs	Fasting plasma blood glucose (mMol)	F	4	2.5,....,19.9 99.9		Legal valid value Enter if blood sugar is missing

sputum	Macroscopic sputum aspect	I	1	1 Mucoid 2 Purulent 3 Muco-purulent 4 Blood-tinged 5 Salivary 6 Other 9 Unknown
result	Examination result	F	3	0 Negative 1 "1+ positive" 2 "2+ positive" 3 "3+ positive" 9 "No result recorded" 4 "Positive, not quantified" 5 "Scanty, not quantified" 0.1 "Scanty, 1 AFB per 100 fields" 0.2 "Scanty, 2 AFB per 100 fields" 0.3 "Scanty, 3 AFB per 100 fields" 0.4 "Scanty, 4 AFB per 100 fields" 0.5 "Scanty, 5 AFB per 100 fields" 0.6 "Scanty, 6 AFB per 100 fields" 0.7 "Scanty, 7 AFB per 100 fields" 0.8 "Scanty, 8 AFB per 100 fields" 0.9 "Scanty, 9 AFB per 100 fields"

The D_EX01_PATIENT.CHK file:

```
LABELBLOCK
  LABEL label_sex
    1 Female
    2 Male
    9 Unknown
  END
  LABEL label_marital
    1 Annulled
    2 Cohabiting
    3 Divorced
    4 Engaged
    5 Married
    6 Separated
    7 Single
    8 Widowed
    9 Unknown
  END
END

patid
  KEY UNIQUE 1
  MUSTENTER
END

sex
  COMMENT LEGAL USE label_sex SHOW
  MUSTENTER
  TYPE COMMENT
  AFTER ENTRY
  RELATE patid d_ex01_examination.rec
  END
END

marital
  COMMENT LEGAL USE label_marital SHOW
  MUSTENTER
  TYPE COMMENT
END
```

Note:

The line:

```
RELATE patid d_ex01_examination.rec
```

is for the final file. It must be adapted accordingly for the two duplicate files which you make first for validation!

The D_EX01_EXAMINATION.CHK file

```
LABELBLOCK
  LABEL label_sputum
    1 Mucoid
    2 Purulent
    3 Muco-purulent
    4 Blood-tinged
    5 Salivary
```

```

        6 Other
        9 Unknown
    END
    LABEL label_result
        0.0 Negative
        1.0 "1+ positive"
        2.0 "2+ positive"
        3.0 "3+ positive"
        9.0 "No result recorded"
        4.0 "Positive, not quantified"
        5.0 "Scanty, not quantified"
        0.1 "Scanty, 1 AFB per 100 fields"
        0.2 "Scanty, 2 AFB per 100 fields"
        0.3 "Scanty, 3 AFB per 100 fields"
        0.4 "Scanty, 4 AFB per 100 fields"
        0.5 "Scanty, 5 AFB per 100 fields"
        0.6 "Scanty, 6 AFB per 100 fields"
        0.7 "Scanty, 7 AFB per 100 fields"
        0.8 "Scanty, 8 AFB per 100 fields"
        0.9 "Scanty, 9 AFB per 100 fields"
    END
END

patid
    KEY 1
    NOENTER
END

examid
    KEY 2
    NOENTER
END

dateexam
    RANGE 01/01/2007 31/12/2007
    LEGAL
        01/01/1800
    END
    MUSTENTER
    AFTER ENTRY
        examid=patid+"-"+month(dateexam)+"-"+day(dateexam)
    END
END

bs
    MUSTENTER
END

sputum
    COMMENT LEGAL USE label_sputum SHOW
    MUSTENTER
    TYPE COMMENT
END

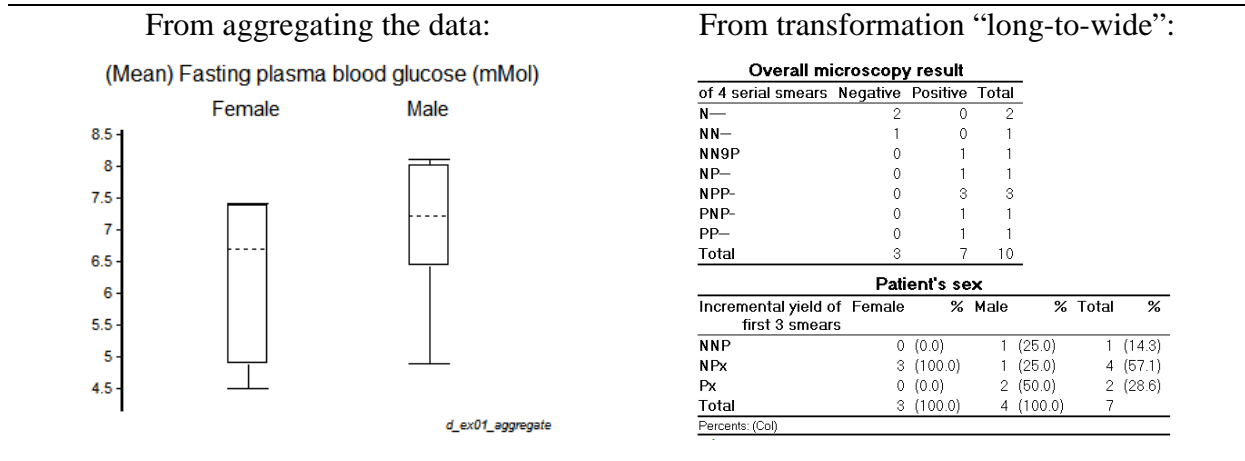
micro
    COMMENT LEGAL USE label_result SHOW
    MUSTENTER
    TYPE COMMENT
END

```

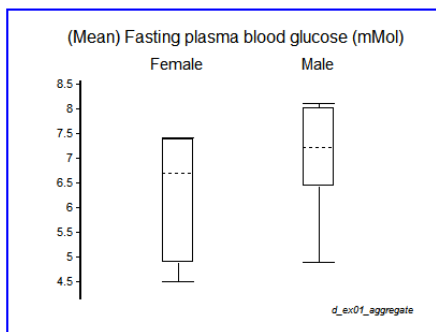
This file is the same for all three files.

Tasks:

- Write a program *D_EX01.PGM* that merges the two files, then prepare sets for the aggregated data and for the “long-to-wide” transformation to produce the following output (obtaining the same numbers is relevant) respectively:



To get:



"Table 1. Pattern of serial smear results"

definition by microscopy			
of 4 serial smears	Negative	Positive	Total
N—	2	0	2
NN—	1	0	1
NN9P	0	1	1
NP—	0	1	1
NPP—	0	3	3
PNP—	0	1	1
PP—	0	1	1
Total	3	7	10

"Table 2. Incremental yield among positive results"

Patient's sex				
Incremental yield of first 3 smears	Female	% Male	% Total	%
NNP	0 (0.0)	1 (25.0)	1 (14.3)	
NPx	3 (100.0)	1 (25.0)	4 (57.1)	
Px	0 (0.0)	2 (50.0)	2 (28.6)	
Total	3 (100.0)	4 (100.0)	7	

Percents: (Col)

The D_EX01.PGM reads:

```
* Exercise D_EX01
* Merging files and aggregating files
* Copying and transposing data from "Long-to-wide"

cls
close
logclose

read "d_ex01_examination.rec"
merge patid /file="d_ex01_patient.rec" /table

sort patid dateexam
gen i exam=1
if patid=patid[_n-1] then exam=exam[_n-1]+1
label exam "Number of examination"

savedata "templ.rec" /replace

cls
close
read "templ.rec"

*****
* Create an aggregate data set
* to determine means

aggregate patid sex /mean="bs" /save="d_ex01_aggregate.rec" /replace

cls
close
read "d_ex01_aggregate.rec"

* set display databrowser=on
* browse
* tables sex meabs
* boxplot meabs /by=sex
*****
* Transpose / copy "long-to-wide"

cls
close
read "templ.rec"

* freq exam

cls
gen d dateexam1=date("31/12/1899")
gen d dateexam2=date("31/12/1899")
gen d dateexam3=date("31/12/1899")
gen d dateexam4=date("31/12/1899")
                                dateexam1=dateexam
if (patid[_n])=(patid[_n+1]) then dateexam2=dateexam[_n+1]
if (patid[_n])=(patid[_n+2]) then dateexam3=dateexam[_n+2]
if (patid[_n])=(patid[_n+3]) then dateexam4=dateexam[_n+3]

cls
gen f micro1=-1
gen f micro2=-1
gen f micro3=-1
gen f micro4=-1
                                micro1=micro
if (patid[_n])=(patid[_n+1]) then micro2=micro[_n+1]
```

```

if (patid[_n])=(patid[_n+2]) then micro3=micro[_n+2]
if (patid[_n])=(patid[_n+3]) then micro4=micro[_n+3]

cls
gen i sputum1=-1
gen i sputum2=-1
gen i sputum3=-1
gen i sputum4=-1

                                sputum1=sputum
if (patid[_n])=(patid[_n+1]) then sputum2=sputum[_n+1]
if (patid[_n])=(patid[_n+2]) then sputum3=sputum[_n+2]
if (patid[_n])=(patid[_n+3]) then sputum4=sputum[_n+3]

select exam=1
savedata "temp2.rec" /replace

cls
close
read "temp2.rec"

define res1 ##
res1=integer(micro1)*10
label res1 "Microscopy result of 1st examination"

cls
define res2 ##
res2=integer(micro2)*10
label res2 "Microscopy result of 2nd examination"

cls
define res3 ##
res3=integer(micro3)*10
label res3 "Microscopy result of 3rd examination"

cls
define res4 ##
res4=integer(micro4)*10
label res4 "Microscopy result of 4th examination"

labelvalue res1-res4 / 0="Negative"
labelvalue res1-res4 /10="1+ positive"
labelvalue res1-res4 /10="1+ positive"
labelvalue res1-res4 /20="2+ positive"
labelvalue res1-res4 /30="3+ positive"
labelvalue res1-res4 /90="No res recorded"
labelvalue res1-res4 /40="Positive, not quantified"
labelvalue res1-res4 /50="Scanty, not quantified"
labelvalue res1-res4 / 1="Scanty, 1 AFB per 100 fields"
labelvalue res1-res4 / 2="Scanty, 2 AFB per 100 fields"
labelvalue res1-res4 / 3="Scanty, 3 AFB per 100 fields"
labelvalue res1-res4 / 4="Scanty, 4 AFB per 100 fields"
labelvalue res1-res4 / 5="Scanty, 5 AFB per 100 fields"
labelvalue res1-res4 / 6="Scanty, 6 AFB per 100 fields"
labelvalue res1-res4 / 7="Scanty, 7 AFB per 100 fields"
labelvalue res1-res4 / 8="Scanty, 8 AFB per 100 fields"
labelvalue res1-res4 / 9="Scanty, 9 AFB per 100 fields"

cls
label sputum1 "Macroscopic sputum of 1st examination"
label sputum2 "Macroscopic sputum of 2nd examination"
label sputum3 "Macroscopic sputum of 3rd examination"
label sputum4 "Macroscopic sputum of 4th examination"
labelvalue sputum1-sputum4 /1="Mucoid"
labelvalue sputum1-sputum4 /2="Purulent"
labelvalue sputum1-sputum4 /3="Muco-purulent"
labelvalue sputum1-sputum4 /4="Blood-tinged"

```

```

labelvalue sputum1-sputum4 /5="Salivary"
labelvalue sputum1-sputum4 /6="Other"
labelvalue sputum1-sputum4 /9="Unknown"

cls
define res1c _
                res1c="-"
if res1= 0      then res1c="N"
if res1> 0 and res1<90 then res1c="P"
if res1=90     then res1c="9"

cls
define res2c _
                res2c="-"
if res2= 0      then res2c="N"
if res2> 0 and res2<90 then res2c="P"
if res2=90     then res2c="9"

cls
define res3c _
                res3c="-"
if res3= 0      then res3c="N"
if res3> 0 and res3<90 then res3c="P"
if res3=90     then res3c="9"

cls
define res4c _
                res4c="-"
if res4= 0      then res4c="N"
if res4> 0 and res4<90 then res4c="P"
if res4=90     then res4c="9"

define pattern ____
pattern=res1c+res2c+res3c+res4c
label pattern "Pattern of 4 serial smears"

* freq pattern

cls
define case #
                case=0
if substr(pattern,1,1)="P" then case=1
if substr(pattern,2,1)="P" then case=1
if substr(pattern,3,1)="P" then case=1
if substr(pattern,4,1)="P" then case=1

label case "Case definition by microscopy"
labelvalue case /0="Negative"
labelvalue case /1="Positive"

define yield ____
if substr(pattern,1,3)="NNN" then yield="NNN"
if substr(pattern,1,3)="NN9" then yield="NN9"
if substr(pattern,1,3)="N99" then yield="N99"
if substr(pattern,1,3)="NN-" then yield="NN9"
if substr(pattern,1,3)="N--" then yield="N99"
if substr(pattern,1,1)="P" then yield="Px"
if substr(pattern,1,2)="NP" then yield="NPx"
if substr(pattern,1,3)="NNP" then yield="NNP"
if substr(pattern,1,4)="NN9P" then yield="NNP"
label yield "Incremental yield of first 3 smears"

keep patid sex marital \
    dateexam1 dateexam2 dateexam3 dateexam4 \
    sputum1 sputum2 sputum3 sputum4 \
    res1 res2 res3 res4 \

```

```

        case pattern yield
savedata "d_ex01_examinee.rec" /replace

*****
* Produce tables on smear pattern and incremental yield

cls
close
read "d_ex01_aggregate.rec"

set option graph /sizex=400
set graph footnote="d_ex01_aggregate"

set echo=off
boxplot meabs /by=sex /bw /sub="    Female                Male"

close
read "d_ex01_examinee.rec"

title "Table 1.  Pattern of serial smear results"
tables case pattern
select case=1
title "Table 2.  Incremental yield among positive results"
tables sex yield /c /PCT
set echo=on

*****
define yesno # global
set echo=off
yesno=?Delete graphs: 1=yes 0=no?
imif yesno=1 then
    erasepng /all /noconfirm
    select
    cls
    type "All graphs erased" /h2
else
    select
endif
set echo=on

*****
* Clean up
set echo=off
yesno=?Delete temporary files: 1=yes 0=no?
imif yesno=1 then
    erase "templ.rec"
    erase "templ.chk"
    erase "temp2.rec"
    erase "temp2.chk"
    select
    cls
    type "All temporary files erased" /h2
    type "File D_EX01_EXAMINEE.REC remains open" /h2
else
    select
endif
set echo=on

```

Exercise 2: A statistical process control chart

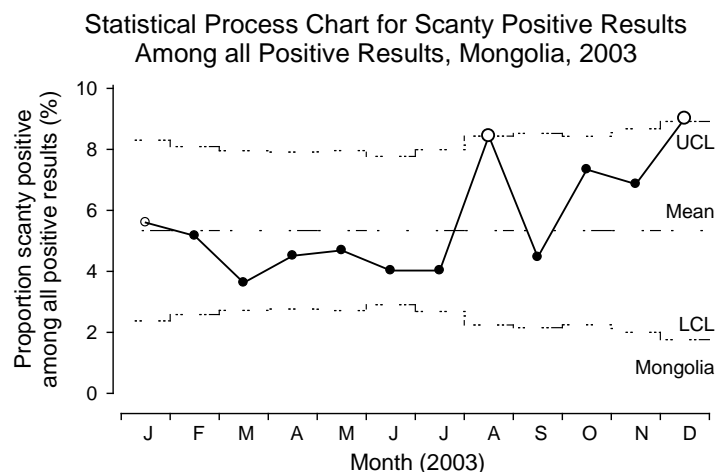
At the end of this exercise you should be able to:

- Aggregate data into the format required for a binomial outcome
- Create a statistical process control chart for a proportion

EpiData Analysis offers a variety of statistical process control (SPC) graphs. In this exercise we will deal with the determination of a proportion that changes over a period of time, and to what extent this variation deviates significantly from the expectation.

Let's assume that you have 1,200 observations during one year in a laboratory. Among these, 10 per cent (120) have positive result. We could determine the standard deviation and a confidence interval around the positive result or go one step further and take the average we expect for one month (12 of 100) with some measurement of uncertainty around this monthly estimate and then chart the actually observed monthly proportion. This would be a correct procedure if the expected monthly denominator is exactly one twelfth of the annual observation. However, this is rarely the case if ever and more likely is the scenario that the denominator varies in each month.

An SPC graph takes these fluctuations in the denominator into account and calculates the uncertainty as a function of the denominator in the element that is of interest (in this example the month). While there are some discussions on what is best to use, it has become customary to use 3 standard deviations as the upper and lower, so-called control limits. In the chart below, the proportion of scanty positive sputum smear microscopy results among all positive results is shown for Mongolia from the large laboratory register study over a one-year period:



Because the denominator differs in every month, the upper and lower control limits also differ from month to month.

We will be looking at examinations (not at examinees) and determine five different proportions (see below).

The dataset for the exercise

The dataset to be used in this exercise is the cleaned dataset of the four-country laboratory study which was provided in the solution to Part C, Exercise 1, dataset C_EX01.REC which is included as a “supplementary required file” with the current exercise with the name MMUZ.REC. MMUZ.REC is slightly different from C_EX01.REC in that the coding for the registration date has been corrected in a few records with an apparent error and the field REGYEAR has been removed. Make sure that you have both the REC and the CHK file in the folder in which you carry out the analysis.

To simplify the task (see specifics at the end), you will limit the analysis to the laboratories in Uganda and to tuberculosis suspects presenting for a diagnostic examination.

The time periods

The Uganda dataset contains information on three years and the unit of measurement of time will be the month. Because each month will thus appear three times (in each year), a new variable must be created that gives a sequential number for each month over the three-year period.

The outcome

You will be determining five different proportions.. A “low-positive” result is defined here as a result that is either scanty positive or 1+ positive. The outcome is the monthly proportion of some kind of positive results among either all smears or among positive smears.

There are thus five different proportions we might be interested in: 1) positive smears of any grade among all smears, 2) scanty positive smear among all smears, 3) low-positive smears among all smears, 4) scanty positive smears among all positive smears, and 5) low-positive smears among all positive smears.

Aggregating the data in the correct format

What is thus needed are counts of examinations with the characteristic (a low-positive result) among all examinations (any examination with at least 1 AFB). You know from tabulations, that tables aggregate individual observations. It is possible to write the same aggregate information that is shown in a table into an EpiData REC file. As an example, you have seven participants in a course from two countries, two of whom are female, and five are male, the only information collected being on COUNTRY and SEX (dataset d_ex02_example1.rec). Browsing the data file you get:

	sex	country
1	Male	Uganda
2	Male	Uganda
3	Male	Uganda
4	Female	Tanzania
5	Male	Uganda
6	Female	Uganda
7	Male	Tanzania

If you make a table:

tables sex country

you get:

Participant's sex			
Participant's country	Female	Male	Total
Tanzania	1	1	2
Uganda	1	4	5
Total	2	5	7

If you replace the command TABLES with AGGREGATE:

```
aggregate sex country
```

you get:

sex	country	N
Female	Tanzania	1
Female	Uganda	1
Male	Tanzania	1
Male	Uganda	4

You can save this in an EpiData REC file with the following options:

```
aggregate sex country /save="temp01.rec" /replace /close  
close  
read "temp01.rec"
```

and then BROWSE to see:

	sex	country	N
1	Female	Tanzania	1
2	Female	Uganda	1
3	Male	Tanzania	1
4	Male	Uganda	4

You now have a REC file with data aggregated for SEX and COUNTRY.

Let's assume now that we have one more field, BANANA, which says how many bananas each individual has in the lunch bag (dataset d_ex02_example2.rec) and browse this dataset, we have:

	sex	country	banana
1	Male	Uganda	2
2	Male	Uganda	3
3	Male	Uganda	1
4	Female	Tanzania	4
5	Male	Uganda	2
6	Female	Uganda	6
7	Male	Tanzania	3

If we ask to aggregate by SEX and COUNTRY, and making a sum of the bananas held by each aggregated group, we would write:

```
aggregate sex country /sum=banana /save="temp01.rec" /replace /close  
close
```

```
read "temp01.rec"
```

and see when browsing:

	sex	country	N	Nbanana	SUMbanana
1	Female	Tanzania	1	1	4
2	Female	Uganda	1	1	6
3	Male	Tanzania	1	1	3
4	Male	Uganda	4	4	8

We noted by browsing the individual dataset that there were 4 Ugandan males with 2, 3, 1, and 2 bananas respectively. With the `/SUM=banana` command option, we sum the bananas for these four Ugandan males and get therefore the 8 bananas in an EpiData Analysis created field `SUMbanana`, which is the total, while `Nbanana` is the count of people with the characteristic (e.g., 4 male Ugandans).

If we modify our questionnaire and ask each of 40 individuals on any of 5 days whether they have a banana with them or not (a binomial outcome which we may code as 0 for not, and 1 for having a banana) and have thus only two variables (in addition to the individual identifier), and where the day sequence of entering the data does not matter, then we would see in browsing the data (only the first 20 individuals, dataset `d_ex02_example3.rec`):

Showing value labels

	dd	ban
1	1	Has no banana
2	1	Has a banana
3	1	Has no banana
4	2	Has a banana
5	3	Has a banana
6	3	Has a banana
7	4	Has a banana
8	4	Has a banana
9	4	Has a banana
10	4	Has no banana
11	5	Has a banana
12	5	Has no banana
13	2	Has a banana
14	3	Has no banana
15	4	Has a banana
16	5	Has a banana
17	1	Has a banana
18	3	Has no banana
19	1	Has a banana
20	3	Has a banana

Showing values

	dd	ban
1	1	0
2	1	1
3	1	0
4	2	1
5	3	1
6	3	1
7	4	1
8	4	1
9	4	1
10	4	0
11	5	1
12	5	0
13	2	1
14	3	0
15	4	1
16	5	1
17	1	1
18	3	0
19	1	1
20	3	1

where `dd` is the field name for the day and `ban` the field name for possession of a banana.

A `tables` command:

```
tables ban dd
```

gives us the number of individuals who have and who do not have a banana on any of these five days:

Having a banana			
Day of observation	Has no banana	Has a banana	Total
1	2	6	8
2	2	3	5
3	6	5	11
4	3	6	9
5	4	3	7
Total	17	23	40

Aggregating the data by day and asking a SUM for the number of “yes” responses to having a banana:

```
aggregate dd /sum=ban /close
```

gives:

dd	N	Nban	SUMban
1	8	8	6
2	5	5	3
3	11	11	5
4	9	9	6
5	7	7	3

Here, `dd` is the day, `N` and `Nban` are identical and are the denominator (the number of individuals asked on that day) and `SUMban` is the count of those who have a banana on that day.

The proportions with a banana on a given day is thus $SUMban/Nban$ (or $SUMban/N$).

Making a statistical process control (SPC) chart for proportions over time

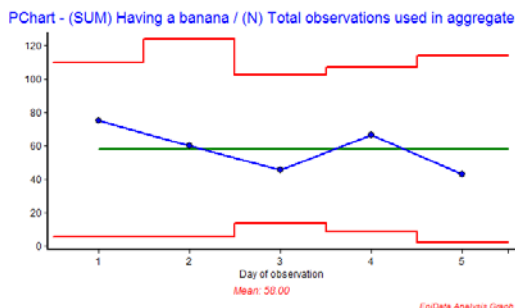
The above aggregate file is all we need to get an SPC chart, the general command for which is:

```
pchart count total [time]
```

or specifically here:

```
aggregate dd /sum=ban /close
pchart sumban N dd
```

and the graphic output is:



The mean proportion here (the green line in the middle) says that 58% of all 40 individuals had a banana, but the upper and lower control limits (the two red lines) differ for each day because the standard deviation varies with the denominator on each day.

Proposed procedure in writing the program

It is proposed to split up the program into five distinct sequential procedures:

- 1) Make the basic dataset
- 2) Count all smears, all positive, all low-positive smears, and all scanty positive smears
- 3) Aggregate the months and sum up the smears in question
- 4) Make the SPC charts

1) Make the basic dataset

In the basic dataset we need to create the years and the months as separate variables and make the appropriate selections:

Month and year of recording: the registers were collected reporting laboratory results during one year up to three years between January 1999 and December 2003. When making a cross-tabulation of country versus registration year (create a field for registration year from REGDATE), we see that:

Year of registration	Country				Total
	Moldova	Mongolia	Uganda	Zimbabwe	
1999	0	0	17300	0	17300
2000	0	0	18662	0	18662
2001	0	0	18088	1213	19301
2002	0	149	0	29307	29456
2003	17725	22406	0	3958	44089
Total	17725	22555	54050	34478	128808

It is thus best to sequentially number all the 60 months from beginning to the end, even if at the end we will require only the three years covered by Uganda.

Select for diagnostic examinations, country, and range of months.

Save the dataset with a new name.

2) Count all smears, all positive, all low-positive smears, and all scanty positive smears

Create four new variables that count for each record respectively all smears, all positive smears, all low-positive smears and all scanty positive smears

3) Aggregate the months and sum up the smears in question and save them to four different files

The element to be aggregated is the month and for each month one must have the relevant smears (all, all positive, all scanty positive, all low-positive). The aggregated data are saved into four different files

4) Merge the four files

The four aggregated files are merged (with the month used as the unique identifier).

5) Make the SPC charts

The appropriate chart type for this binomial outcome is a PChart which has the format:

pchart numerator denominator time

Up to four have to be produced to get the four proportions of interest on the time axis.

Task

- ***Produce five PCharts to display the proportion of 1) positive smears among all smears, 2) low-positive smears among all smears, 3) scanty positive smears among all smears, 4) low-positive smears among all positive smears, and 5) scanty positive smears among all positive smears.***

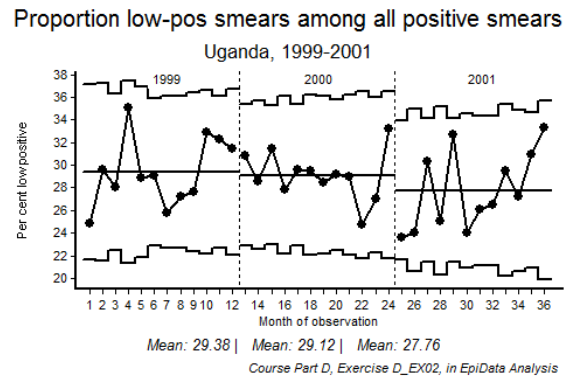
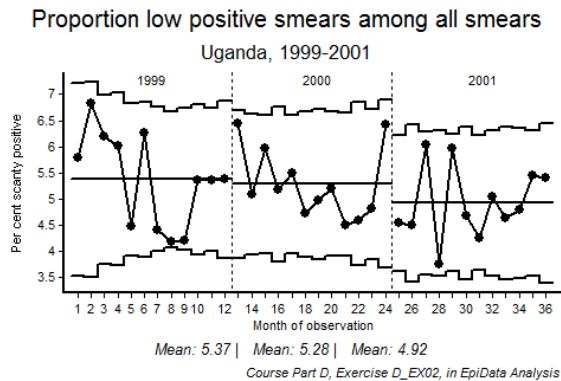
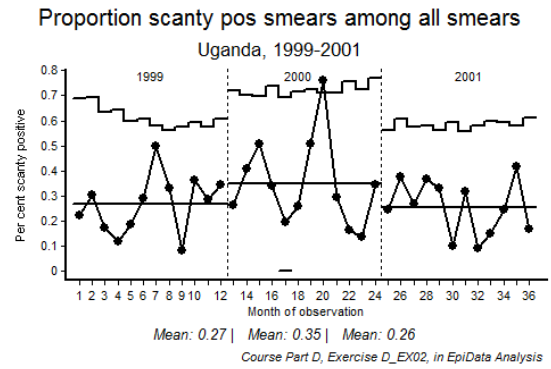
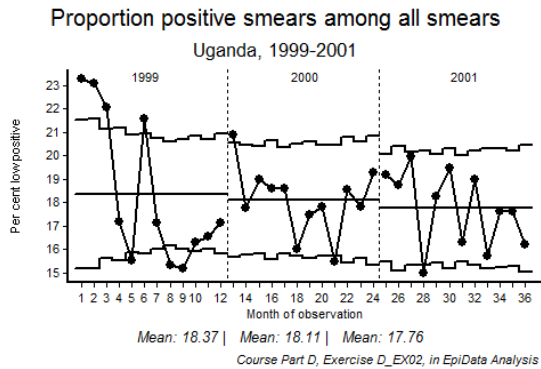
Solution to Exercise 2: A statistical process control chart

- Key points:**
- You must determine on how to aggregate data to obtain the numerator and denominator and, where necessary, the time components over the observation period
 - For a binomial outcome, a Pchart is the appropriate SPC chart

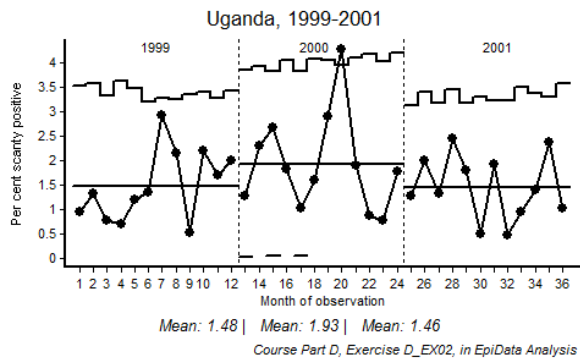
Task

- Produce five PCharts to display the proportion of 1) positive smears among all smears, 2) low-positive smears among all smears, 3) scanty positive smears among all smears, 4) low-positive smears among all positive smears, and 5) scanty positive smears among all positive smears.

These are the five PCharts:



Proportion scanty pos smears among all positive smears



This is the program `d_ex02.pgm` that produced them:

```
* 1) Determine the proportion of positive
* smears among all smears
* 2) Determine the proportion of low-positive
* smears among all positive smears
* 3) Determine the proportion of scanty positive
* smears among all positive smears
* Definition of positive: any quantified positive
* or any scanty (quantified scanty or unquantified scanty)
* Definition low-positive: any smear which is 1+ positive or scanty positive
```

```
*****
* Procedural steps
* 1) Make basic dataset
* 2) Start selection process
* 3) Aggregate data
* 4) Make SPC charts
*****
* 1) Prepare basic dataset

cls
close
logclose

read "mmuz.rec"

define regyear ####
regyear=year(regdate)

cls
tables country regyear

gen i mmseq=0
if year(regdate)=1999 then mmseq=month(regdate)
if year(regdate)=2000 then mmseq=month(regdate)+12
if year(regdate)=2001 then mmseq=month(regdate)+24
if year(regdate)=2002 then mmseq=month(regdate)+36
if year(regdate)=2003 then mmseq=month(regdate)+48
label mmseq "Sequential month"

select reason=0
select country=3
select mmseq>0 and mmseq<37
* The selection above is not necessary for Uganda alone
* as there are no examinees in 2003

drop age country reason sex laboratory regdate
savedata "temp_01.rec" /replace

*****
* 2) Count all smears, all positive, all scanty positive,
```

```

* all low positive smears

cls
close
read "temp_01.rec"

* Count all smears
* Note: non-sensical sequences removed, thus simply:
gen i allsmears=1
if result2<>9 then allsmears=2
if result3<>9 then allsmears=3

* Count all positive quantified smears
* (include scanty not quantified)
gen i allpos1=0
if result1>0 and result1<4 then allpos1=1
if result1=5 then allpos1=1
gen i allpos2=0
if result2>0 and result2<4 then allpos2=1
if result2=5 then allpos2=1
gen i allpos3=0
if result3>0 and result3<4 then allpos3=1
if result3=5 then allpos3=1
gen i allpos=allpos1+allpos2+allpos3

* Count all scanty smears
* (include scanty not quantified)
gen i scantpos1=0
* (include scanty not quantified)
if result1>0 and result1<1 then scantpos1=1
if result1=5 then scantpos1=1
gen i scantpos2=0
if result2>0 and result2<1 then scantpos2=1
if result2=5 then scantpos2=1
gen i scantpos3=0
if result3>0 and result3<1 then scantpos3=1
if result3=5 then scantpos3=1
gen i scantypos=scantpos1+scantpos2+scantpos3

* Count all low positive smears
* (include scanty not quantified)
gen i lowpos1=0
if result1>0 and result1<2 then lowpos1=1
if result1=5 then lowpos1=1
gen i lowpos2=0
if result2>0 and result2<2 then lowpos2=1
if result2=5 then lowpos2=1
gen i lowpos3=0
if result3>0 and result3<2 then lowpos3=1
if result3=5 then lowpos3=1
gen i lowpos=lowpos1+lowpos2+lowpos3

drop result1 result2 result3 \
    allpos1 allpos2 allpos3 \
    scantpos1 scantpos2 scantpos3 \
    lowpos1 lowpos2 lowpos3
savedata "temp_02.rec" /replace

*****
* 3) Aggregate data

cls
close
read "temp_02.rec"

agg mmseq /sum=allsmears /sum=allpos /sum=scantypos /sum=lowpos /close
drop n nallsm1 nallpos nscantpos nlowpos
rename sumallsm1 to allsmears
rename sumallpos to allpos
rename sumscant1 to scantpos
rename sumlowpos to lowpos
drop n nallsm1 nallpos nscant1 nlowpos
savedata "examinations.rec" /replace

```

```

*****
* 4) Make SPC charts

set option spc= /sizeX=500 /sizeY=350

cls
close
logclose

read "examinations.rec"

set echo=off
pchart allpos allsmears mmseq /xtext="Month of observation" /bw \
  /ytext="Per cent low positive" \
  /ti="Proportion positive smears among all smears" \
  /sub="Uganda, 1999-2001" \
  /fn="Course Part D, Exercise D_EX02, in EpiData Analysis" \
  /b=12 /b=24 \
  /xlined=12.5 /xlined=24.5 \
  /text="120,70,1999,0" \
  /text="260,70,2000,0" \
  /text="400,70,2001,0"

pchart scantpos allsmears mmseq /xtext="Month of observation" /bw \
  /ytext="Per cent scanty positive" \
  /ti="Proportion scanty pos smears among all smears" \
  /sub="Uganda, 1999-2001" \
  /fn="Course Part D, Exercise D_EX02, in EpiData Analysis" \
  /b=12 /b=24 \
  /xlined=12.5 /xlined=24.5 \
  /text="120,70,1999,0" \
  /text="250,70,2000,0" \
  /text="400,70,2001,0"

pchart lowpos allsmears mmseq /xtext="Month of observation" /bw \
  /ytext="Per cent scanty positive" \
  /ti="Proportion low positive smears among all smears" \
  /sub="Uganda, 1999-2001" \
  /fn="Course Part D, Exercise D_EX02, in EpiData Analysis" \
  /b=12 /b=24 \
  /xlined=12.5 /xlined=24.5 \
  /text="120,70,1999,0" \
  /text="260,70,2000,0" \
  /text="400,70,2001,0"

pchart lowpos allpos mmseq /xtext="Month of observation" /bw \
  /ytext="Per cent low positive" \
  /ti="Proportion low-pos smears among all positive smears" \
  /sub="Uganda, 1999-2001" \
  /fn="Course Part D, Exercise D_EX02, in EpiData Analysis" \
  /b=12 /b=24 \
  /xlined=12.5 /xlined=24.5 \
  /text="130,70,1999,0" \
  /text="260,70,2000,0" \
  /text="400,70,2001,0"

pchart scantpos allpos mmseq /xtext="Month of observation" /bw \
  /ytext="Per cent scanty positive" \
  /ti="Proportion scanty pos smears among all positive smears" \
  /sub="Uganda, 1999-2001" \
  /fn="Course Part D, Exercise D_EX02, in EpiData Analysis" \
  /b=12 /b=24 \
  /xlined=12.5 /xlined=24.5 \
  /text="120,70,1999,0" \
  /text="250,70,2000,0" \
  /text="400,70,2001,0"

set echo=on

*****
* Clean up

set echo=off
define yesno # global

```

```
yesno=?Cleaning up temporary files: 1=yes 0=no?
imif yesno=1 then
  erasepng /noconfirm /all
  erase "examinations.rec          "
  erase "examinations_allpos.rec"
  erase "examinations_lowpos.rec"
  erase "examinations_scanty.rec"
  erase "examinations_smears.rec"
  erase "temp_01.chk              "
  erase "temp_01.rec              "
  erase "temp_02.chk              "
  erase "temp_02.rec              "
  cls
  type "All temporary files erased" /h2
else
  type "All temporary files retained" /h2
endif
set echo=on
```

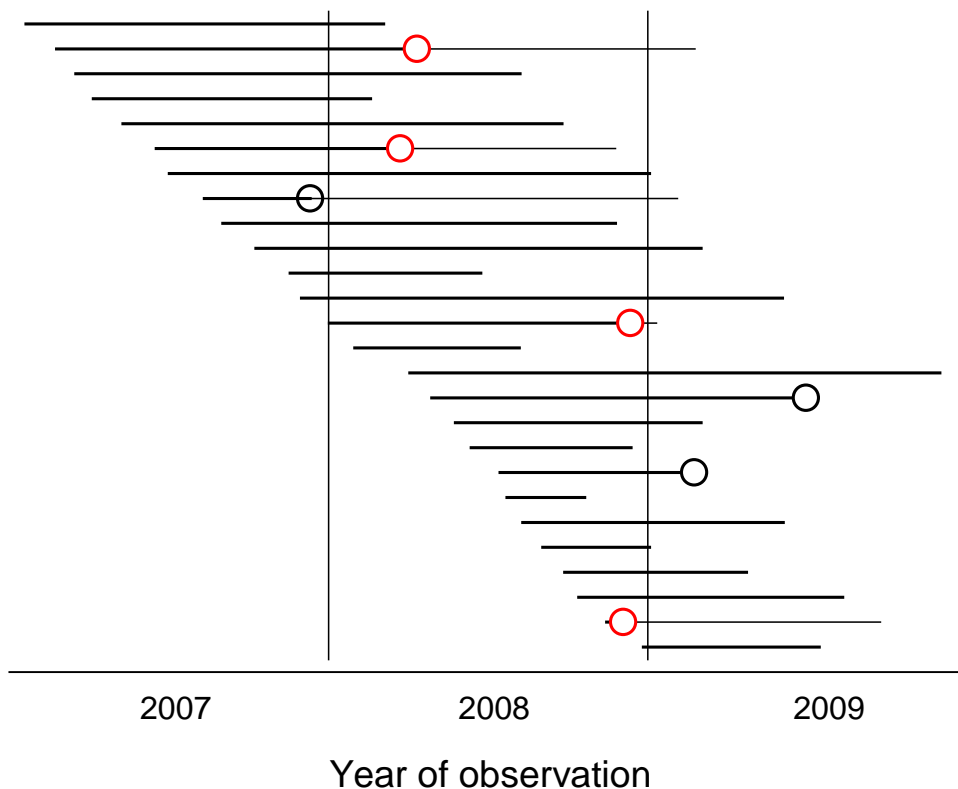
Exercise 3: Survival analysis

At the end of this exercise you should be able to:

- a. Understand the indications for survival analysis
- b. Be able to do a simple survival analysis in EpiData Analysis

An example of a survival analysis using the Kaplan-Meier method

We observe people over time, starting in 2007 into 2009 and note whether or not they develop an event (whatever it may be) during the observation time as shown for 26 individuals in the following graph:



We can think of this setting like of an institution, such as a prison, where inmates enter the prison and are discharged at some time. During incarceration some may develop tuberculosis. If our interest is focused on the year 2008, we have two measures of the magnitude of the problem.

We might calculate the incidence rate in the year 2008. The numerator is 4 cases. The denominator is person-time of observation. As the following table shows, we know the date of entry and exit from the institution, and the date the event occurs among those who had and event:

ID	Entry date	Exit date	Event	Event date	Obs start	Obs end	Obs days
A	20-01-2007	04-03-2008	No		01-01-2008	04-03-2008	63
B	24-02-2007	23-02-2009	Yes	11-04-2008	01-01-2008	11-04-2008	101

C	18-03-2007	07-08-2008	No		01-01-2008	07-08-2008	219
D	07-04-2007	18-02-2008	No		01-01-2008	18-02-2008	48
E	11-05-2007	24-09-2008	No		01-01-2008	24-09-2008	267
F	18-06-2007	24-11-2008	Yes	23-03-2008	--	--	
G	03-07-2007	02-01-2009	No		01-01-2008	31-12-2008	365
H	12-08-2007	03-02-2009	Yes	11-12-2007	01-01-2008	31-12-2008	365
I	02-09-2007	24-11-2008	No		01-01-2008	24-11-2008	328
J	10-10-2007	02-03-2009	No		01-01-2008	31-12-2008	365
K	18-11-2007	23-06-2008	No		01-01-2008	23-06-2008	174
L	01-12-2007	03-06-2009	No		01-01-2008	31-12-2008	365
M	02-01-2008	10-01-2009	Yes	11-12-2008	02-01-2008	11-12-2008	344
N	31-01-2008	06-08-2008	No		31-01-2008	06-08-2008	188
O	03-04-2008	30-11-2009	No		03-04-2008	31-12-2008	272
P	28-04-2008	05-07-2009	Yes	30-06-2009	28-04-2008	31-12-2008	247
Q	25-05-2008	02-03-2009	No		25-05-2008	31-12-2008	220
R	12-06-2008	12-12-2008	No		12-06-2008	12-12-2008	183
S	15-07-2008	02-03-2009	Yes	22-02-2009	15-07-2008	31-12-2008	169
T	23-07-2008	20-10-2008	No		23-07-2008	20-10-2008	89
U	10-08-2008	04-06-2009	No		10-08-2008	31-12-2008	143
V	02-09-2008	02-01-2009	No		02-09-2008	31-12-2008	120
W	27-09-2008	23-04-2009	No		27-09-2008	31-12-2008	95
X	13-10-2008	11-08-2009	No		13-10-2008	31-12-2008	79
Y	14-11-2008	23-09-2009	Yes	03-12-2008	14-11-2008	03-12-2008	19
Z	26-12-2008	15-07-2009	No		26-12-2008	31-12-2008	5

Days	4833
Cases	4
Cases/1000 person-days	0.828
Cases/100 person-years	30.2

Each person contributes person-time of observation, starting earliest from the beginning of the year 2008 or later if entry into the system was later. Person-time is contributed until exit from the institution or up to the point of the event if either happened in the year 2008, if later, observation time ends at the right-censoring point of 31 December 2008. Individual F developed the event in 2007 and although still in the system in 2008 does not contribute any person-time of observation in 2008. Individuals P and S developed the event only in 2009 and are thus not counted in 2008 and they contribute to person-time of observation through the end of the year 2008. Thus summed up, the 25 of the 26 inmates contributed 4,833 days of observation time which gives a case rate of 0.8 per 1,000 person-days of observation, or annualized, 30.2 cases per 100 observation years.

Survival analysis

The second way to look at the size of the problem is to ask what the probability for an individual is to “survive” the year 2008 without developing the event.

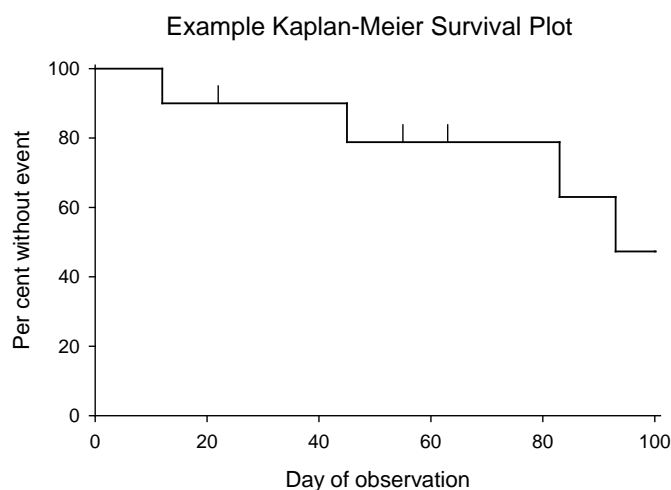
We exclude the person who had the event already in 2007 and sort the persons by the time of event or censoring (discharge or latest end of 2008). We note the number at risk of the beginning of the interval, then the number who got censored, then how many were left after censoring, and finally how many get an event during the interval:

At risk at beginning	Censored	At risk at end	Event at end	Proportion surviving	Survival
25	1	24	0		
24	0	24	1	23/24	0.958
23	1	22	0		0.958
22	1	21	0		0.958
21	1	20	0		0.958
20	0	20	1	(19/20)*0.958	0.910
19	1	18	0		0.910
18	1	17	0		0.910
17	0	17	1	(16/17)*0.910	0.857
16	1	15	0		0.857
15	1	14	0		0.857
14	1	13	0		0.857
13	1	12	0		0.857
12	1	11	0		0.857
11	1	10	0		0.857
10	1	9	0		0.857
9	1	8	0		0.857
8	1	7	0		0.857
7	1	6	0		0.857
6	1	5	0		0.857
5	1	4	0		0.857
4	0	4	1	(3/4)*0.857	0.643
3	3	0	0		0.643

At each point where an event (not censoring) happens, we calculate the survival probability by dividing the number “surviving” after the event by the number at risk at the beginning of the interval when the event occurred.

You may note that censoring during an interval is assumed not to affect survival probability during that interval, but censored individuals are also subtracted from those at risk for the next interval.¹⁻³ This assumption is a simplification because censoring may, under certain circumstances, indeed be affecting the survival probability during the remaining interval. In any case, however, taking both the occurrence of the event and censoring into account for each interval following an event is a much more appropriate way to calculate survival than the proportion with an event among all who entered the cohort or by removing those censored from the cohort.

Graphically, we summarize the Kaplan-Meier survival probability as a step graph, sometimes showing the points when an individual was censored:



1. Kaplan E L, Meier P. Nonparametric estimation from incomplete observations. J Am Stat Ass 1958;53:457-81.
2. Bland J M, Altman D G. Survival probabilities (the Kaplan-Meier method). (Statistics notes). BMJ 1998;317:1572.
3. Fink S A, Brown R S, Jr. Survival analysis. Gastroenterol Hepatol 2006;2:380-3.

Survival analysis in EpiData Analysis

The command for a Kaplan-Meier survival analysis in EpiData Analysis is:

```
lifetable outcome interval
```

or

```
lifetable outcome startdate enddate
```

There are a multitude of options available both lifetable-specific options and general graph options. Use the Help file to test out various options until you have the survival plot that suits your needs.

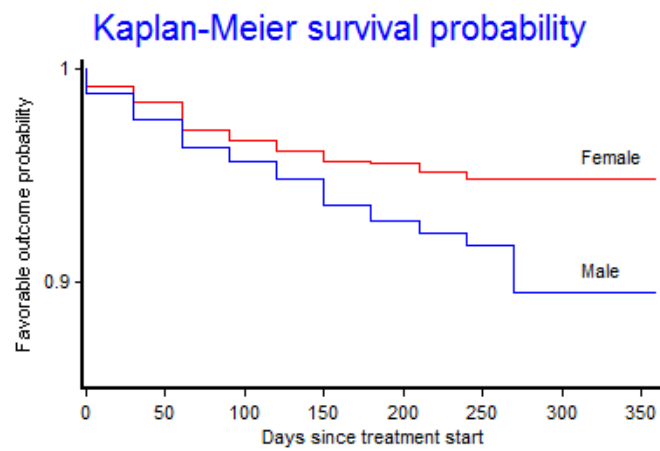
Tasks

The purpose of the exercise is to demonstrate quantitatively the probability of remaining without an event.

We use to this end a real dataset from a tuberculosis program, although we have removed any identifier and most variables, and retained to simplify your work only records of patients who have an exact date of treatment start and an exact date of treatment end. The dataset is provided as a supplementary file `d_ex03.rec`.

- 1) *Define a binomial outcome, where “favorable” is cured or treatment completed and all other outcomes are “unfavorable”*

2) *Do the lifetable analysis only for new sputum smear-positive cases and show the survival probability for females and males in the same graph*



EpiData course: Exercise D_EX03

Solution to Exercise 3: Survival analysis

Key points:

- In this simplified example, there was no censoring, people either had the event or they didn't
- Survival analysis requires two variables, the time when the event occurred or the endpoint of observation, and whether there was an event or not

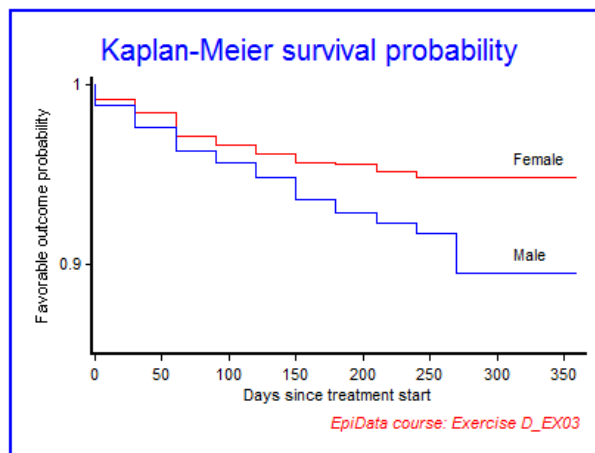
Tasks

The purpose of the exercise is to demonstrate quantitatively the probability of remaining without an event.

We use to this end a real dataset from a tuberculosis program, although we have removed any identifier and most variables, and retained to simplify your work only records of patients who have an exact date of treatment start and an exact date of treatment end. The dataset is provided as a supplementary file `d_ex03.rec`.

- 1) *Define a binomial outcome, where “favorable” is cured or treatment completed and all other outcomes are “unfavorable”*
- 2) *Do the lifetable analysis only for new sputum smear-positive cases and show the survival probability for females and males in the same graph*

The graphic output produced by the program `D_EX03.PGM`:



The program `D_EX03.PGM` to produce the above graphic output:

- * Exercise D_EX03
- * Survival Analysis with EpiData Analysis

```
cls
close
logclose
```

```

read "d_ex03.rec"

define advout #
advout=1
if outcome<=2 then advout=0
label advout "Adverse treatment outcome"
labelvalue advout /0="Favorable outcome"
labelvalue advout /1="Adverse outcome"

define case #
case=0
if sm00>0 and sm00<9 then case=1
label case "Microscopy-defined case"
labelvalue case /0="Non-case"
labelvalue case /1="Smear-pos case"

cls
select sex<>9
select case=1
select category=1
lifetable advout startext outext \
    /i=b30 /by=sex /adj \
    /nocl \
    /ymin=0.85 \
    /ymax=1 \
    /t \
*    /bw \
    /ti="Kaplan-Meier survival probability" \
    /fn="EpiData course: Exercise D_EX03" \
    /xtext="Days since treatment start" \
    /ytext="Favorable outcome probability" \
    /text="335,90,Female,0" \
    /text="335,155,Male,0"

select

*****
* Clean up
erasepng /all /noconfirm

```

Exercise 4: Creating a menu for standard reports

At the end of this exercise you should be able to:

- a. Write an HTML-based interface for a menu
- b. Writing programs with interactive prompting

This exercise involves working with HTML to make a user-friendly interface with menus and to prepare the EpiData Analysis programs that execute the production of standard reports with interactive prompting for choices.

The end product interface will appear as follows:



Quarterly reports on case finding and treatment outcome



Enter data



Access EpiData Entry

Create reports



A quarterly report on case-finding



A quarterly report on treatment outcome

Terminate session



Erase temporary session files



Exit the report menu

We will structure the exercise as follows:

Preparatory work

- Installing EpiData Entry and EpiData Analysis

- Delete an obsolete sub-folder and create new sub-folders

- Unzip the required files from the course website into the relevant sub-folder

Background how EpiData Analysis starts and how to shape its looks when opening

Making the interface in HTML

- Using an HTML editor to make the skeleton of the interface

- Editing the HTML file

Making the EpiData Analysis programs

- The program to produce the quarterly report on case finding

- Make the basic dataset

- Make the interactive selection process

- Make the charts side by side

The program to produce the quarterly report on treatment outcome

Make the basic dataset
Make the interactive selection process

Preparatory work

Installing EpiData Entry and EpiData Analysis

First make sure your “Normal EpiData Analysis” is updated to the newest version. The recommended location for both EpiData Entry and EpiData Analysis is C:\EPIDATA.

In addition, we will install EpiData Entry and EpiData Analysis into a separate folder (make it user-defined to keep control over what’s going to happen). When prompted for the path, put it into:

```
c:\epidata_report
```

You will get in this folder three sub-folders and 24 files:

```
docs\  
samples\  
TestData\  
English.ea.lang.txt  
Epdintro.pdf  
EPIDATA.CNT  
EpiData.exe  
EPIDATA.HLP  
epidata.ini  
EpiData.lbl  
epidatastat.10  
epidatastat.12  
epidatastat.15  
epidatastat.20  
EpiDataStat.exe  
epidatastat.ini  
epiout.css  
epiout_b.css  
epiout_w.css  
license.txt  
licensetest.txt  
preventdouble.ea  
readme.rtf  
unins000.dat  
unins000.exe  
unins001.dat  
unins001.exe
```

Delete unnecessary sub-folders and create new sub-folders

The TestData and samples sub-folders are superfluous for this exercise and can be deleted. Instead create **six new** sub-folders, so that you have a total of seven sub-folders:

```
data\  
docs\  
images\  
originals
```

pgm\
required\
temp\
English.ea.lang.txt
...

Unzip the required files from the course website into the relevant sub-folder

The course website contains a zip file with 30 required files:

choice_year.qes
choice_year.rec
epidata.ini
epidata.png
epidata_wikiL.png
epidatastat.ini
epidatastat.png
epidatastatsetup.ini
epiout_monospace.css
eraser.png
imif_example.pgm
insert_pgm_html.txt
next.gif
quit.jpg
sample_2004.chk
sample_2004.eix
sample_2004.qes
sample_2004.rec
sample_2005.chk
sample_2005.eix
sample_2005.qes
sample_2005.rec
sample_2006.chk
sample_2006.qes
sample_2006.rec
side_by_side_graphs.html
start_template.htm
uganda.chk
uganda.rec
union.jpg

Unzip all these 30 files into the EPIDATA_REPORT\REQUIRED sub-folder. Then move and over-write if necessary as follows:

From the EPIDATA_REPORT\REQUIRED sub-folder to the EPIDATA_REPORT\IMAGES sub-folder:

epidata.png
epidata_wikiL.png
epidatastat.png
eraser.png
next.gif
quit.jpg
union.jpg

From the EPIDATA_REPORT\REQUIRED sub-folder to the EPIDATA_REPORT\ORIGINALS sub-folder:

choice_year.qes
choice_year.rec
uganda.chk
uganda.rec
sample_2004.chk
sample_2004.eix
sample_2004.qes
sample_2004.rec
sample_2005.chk
sample_2005.eix
sample_2005.qes
sample_2005.rec
sample_2006.chk
sample_2006.qes
sample_2006.rec
side_by_side_graphs.html

From the EPIDATA_REPORT\REQUIRED sub-folder to the \EPIDATA_REPORT root folder:

Epidata.ini
epidatastat.ini
epidatastatsetup.ini

Note to the above three *.ini files: you could perfectly well edit all these three files (and we show below how to edit the epidatastat.ini to serve our purpose in some detail).

From the EPIDATA_REPORT\REQUIRED sub-folder to the EPIDATA_REPORT \DOCS\EN sub-folder:

start_template.htm
epiout_monospace.css

From the EPIDATA_REPORT\REQUIRED sub-folder to the EPIDATA_REPORT\PGM sub-folder:

imif_example.pgm

This should leave you with a single file in the EPIDATA_REPORT\REQUIRED sub-folder:

insert_pgm_html.txt

This is an example that we may make use of later when we insert HTML code to call a program.

Background how EpiData Analysis starts and how to shape its looks when opening

In any software you may use, there is an executable file that starts the process of opening the program and displaying the standard interface. In EpiData Analysis, this file is in the root of the folder in which you installed the program and its name is:

EpiDataStat.exe

This file contains all the essential code to direct EpiData Analysis to do what it is expected to do. Publishing this code will make EpiData Analysis what we call “open-source” software. While the designers and programmers of EpiData software are working on preparing this source code with sufficiently detailed documentation to ultimately make it open-source, the time is not yet quite mature to do so because the documentation must be un-ambiguous and clear for any other developer to derive usefulness for further development from it. As we are not software developer ourselves, we do not have any need to know the source code, the only thing we need to know is some very basic things that this executable file does, and how we can override certain things to meet our needs.

Among other files, the executable file looks first for an HTML file that is located in the EPIDATA_REPORT\DOCS\EN sub-folder:

start.htm

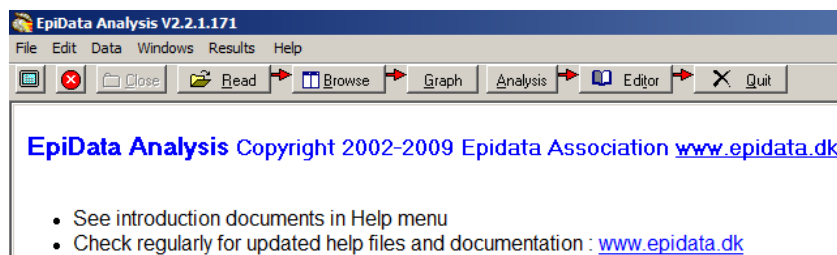
If we double-click this file it opens in our default browser and this is the display we get:

EpiData Analysis Copyright 2002-2009 Epidata Association www.epidata.dk

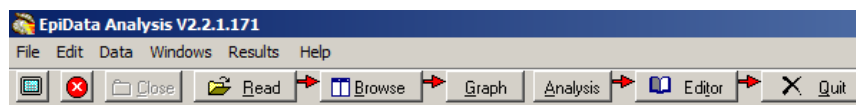
- See introduction documents in Help menu
- Check regularly for updated help files and documentation : www.epidata.dk

We can change this visualized part to our liking in the start.htm file which will be one of our tasks.

When we open EpiData Analysis, we see in addition the version display, the menu bar, and the process bar:

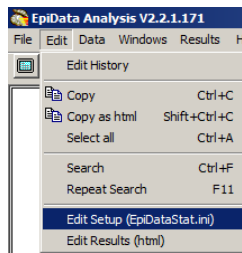


The upper component:



is part of the default defined in the executable file. It is, however, possible to suppress part or all of it by modifying the EpiDataStat.ini file.

With the opening of EpiData Analysis, a small program called EpiDataStat.ini is run. You can access it from the Edit menu:



If you choose to edit this Setup program, it will open in the Editor. Its initial default script is:

```
* EpiData Analysis default settings file
* Edit the next lines to change size or font
set echo=off

* Viewer font and size: (plus editor and help windows)
set browser font size =12
set graph font size =12
set viewer font size =12
set window font size =12
set editor font size =12
set viewer font name ="Verdana,Courier"

* uncomment next two lines to display Chinese Characters
*SET viewer font charset = "gb2312"
*set viewer font name="Arial Unicode MS"

*****
* set options defined during installation:
*****

* To see other set: issue "set" command or look in help file (F1)
*****
set display variables=ON
set display databrowser=OFF
set output open=ON
* set options defined during installation:
*****
* Default folder defined as:
cd c:\epidata_report\TestData
set output folder="c:\epidata_report\TestData"
set language=english
set echo=ON
```

If we strip it of all comments, what remains is:

```
set echo=off
set viewer font size =12
set window font size =12
set editor font size =12
set viewer font name ="Verdana,Courier"
set display variables=OFF
set display databrowser=OFF
set output open=ON
set language=english
set echo=ON
```

a series of SET commands that tells EpiData Analysis some defaults that are operative until we change. We asked you before to write over this file and that replacing file differs from the above (comments that remain are not shown) is shown in red font:

```
1 set echo=off
2 cd temp
3 set display mainmenu=off
4 set display command prompt=off
5 set display worktoolbar=off
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="..\docs\en\epiout_monospace.css"
11 set display variables=OFF
12 set display databrowser=OFF
13 set output open=OFF
14 set output folder="\EpiData_report\temp"
15 set language=english
16 set echo=on
17 set start page="..\docs\en\start_tb.htm"
```

If we rearrange a bit to put similar things together, we may summarize as:

```
1 set echo=off
2 cd temp
3 set start page="..\docs\en\start_tb.htm"
4 set viewer font size =12
5 set window font size =12
6 set editor font size =12
7 set viewer font name ="Verdana,Courier"
8 set language=english
9 set display variables=off
10 set display databrowser=off
11 set display mainmenu=off
12 set display worktoolbar=off
13 set display command prompt=off
14 set output open=off
15 set output folder="\EpiData_report\temp"
16 set style sheet="..\docs\en\epiout_monospace.css"
17 set echo=on
```

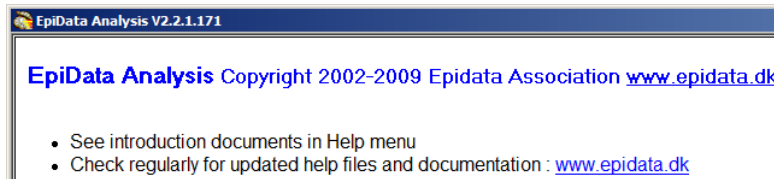
In Lines 1 and 16 we turn the echo (showing the `epidatastat.ini` file running) off and on respectively.

In Line 2, we direct EpiData Analysis to go to the (above created) sub-folder `EPIDATA_REPORT\TEMP`. EpiData Analysis will be in this place when you start with an analysis and if you need to access data files or a program file or any other file, you will need to tell it where these are located *relative to this location*.

In Line 3, we move from this location one step up (`..`), then two down to define the location of the “`start_tb.htm`” file, that is the file that EpiData Analysis should open, when it starts. This file does not yet exist at this point in time. If we put an asterisk in front:

```
* start_tb.htm
```

EpiData Analysis will use the default “start.htm” (it will actually do so without an asterisk as it will bypass it as non-existent) and the interface looks this way:



At the bottom, you see don't see the command line anymore, and only the sub-folder in which EpiData Analysis is after executing the first four lines:



The default style sheet that EpiData Analysis uses is the output.css file, a cascading style sheet. You could make the style in the start.htm file, the name of which we modify on line:

```
16 set start page="..\docs\en\start_tb.htm"
```

but the recommendation to the World Wide Web Consortium (W3C) is to refer in the main file to another specific (*.CSS) file that defines the style. This recommendation is for good reasons: you can always change the style sheet without changing the main HTML file. You copied the epiout_monospace.css file into the EPIDATA_REPORT\DOCS\EN and in Line 16 we instruct EpiData Analysis to use the latter style sheet instead of its default. Below are the results with the two respective style sheets. Do you note the difference?

Display with the default output.css file

<u>Study country</u>	
	<u>N</u>
Moldova	17725
Mongolia	22555
Uganda	54050
Zimbabwe	34478
Total	128808

Display with the epiout_monospace.css file

<u>Study country</u>	
	<u>N</u>
Moldova	17725
Mongolia	22555
Uganda	54050
Zimbabwe	34478
Total	128808

Of course, you can make your own style sheets but this will require learning a bit more on how to make one, but this not subject of this exercise.

EpiData Analysis writes log files and other stuff that are useful to review when something goes wrong. These files are not usually used when all goes as expected and to get them out of the way, the command in Line 15 redirects the output to be stored in the sub-folder TEMP.

With this brief and rather simplified introduction of how EpiData Analysis starts to work, you should now have an understanding on what the EpiDataStat.ini file does and how you can always access it and change it on the fly. In your “regular” EpiData Analysis program, it

will often prove useful to direct it with this file to a specific project folder on which you are currently working with the CD command in Line 1 and it will always be in the right place for the project until you change it.

After we are now done with the role of the `EpiDataStat.ini` file, we have to deal with the HTML file `start_tb.htm`, which gets us into learning some basics about the HTML language.

Making the interface in HTML

Using an HTML editor to make the skeleton of the interface

HTML is the language of the Internet which is interpreted by a browser such as the proprietary Internet Explorer™ or the open-source and free Firefox® to make it visually comprehensible and appealing for the user. A simple text that looks like:

This is a simple text to show the browser interpretation of HTML text and the actual underlying HTML.



We add above an icon for display.

has the following underlying HTML code in the browser:

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
2
3 <html>
4
5 <head>
6 <meta content="text/html; charset=ISO-8859-1" http-equiv="content-type"><title>temp</title>
7 </head>
8
9 <body>
10 <big><span style="font-family: Arial;">This is a simple text to show the browser</span>
11 <br style="font-family: Arial;"><span style="font-family: Arial;">interpretation of HTML text and the actual</span>
12 <br style="font-family: Arial;"><span style="font-family: Arial;">underlying HTML.</span></big><br>
13 <br>
14 <big><span style="font-family: Arial;">We add above an icon for display.</span></big>
15 </body>
16
17 </html>
```

This is complex at first look, but when we look at it carefully then we realize that it is a highly logical language and sequence of instructions to the browser.

In Line 1, information is provided that the language conforms with W3C standards and the version of HTML it is using and that you can find this confirmed at the W3C website.

Every component in an HTML document has the principle to define the beginning and the end of the component and the system is the same for all. In the above example we have:

Begin	End
<HTML>	</HTML>
<head>	</head>
<body>	</body>

The structure indicates that everything between the opening and ending HTML tags is in fact HTML. Embedded are two parts, the head and the body, each indicating where it starts and where it ends. Every HTML page is build around these key parts, and within these you may have other sub-components imbedded that follow the same principles such as here within the body:

Begin End

For the time being, we will leave it at that but will come back later to these principles when we work specifically on the `start_tb.htm` file.

Because of the complexity for the beginner, a multitude of software has been developed allowing the user to write normally as in a word processor to see what one actually wants to get. The software translates it into HTML and the page becomes interpretable by the browser. The advantages of such software are obvious but the downside is that it is often very expensive (hundreds of Euros perhaps), and not all is adhering strictly to W3C standards which will make it difficult for some browsers that require strict adherence to interpret the language properly.

You may have heard about the Mozilla Foundation which produces free and excellent open-source software, such as the browser Firefox, the email client Thunderbird, the calendar Sunbird and yes, the HTML editor Nvu. Nvu is still in its infancy and has some problems, some of which have been resolved with the HTML editor KompoZer which you find on this course web in the software section and that we will be using.

You do not need to install KompoZer, it is just a zip file. Unzip it into the root of your hard drive and you get a folder:

KompoZer 0.7.10\

You may make a shortcut to its executable file:

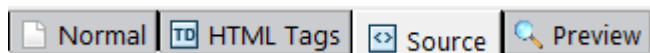
KompoZer.exe

to your desktop and if desired to the quick launch bar to have it accessible at your fingertip with one click away or simply look for it and double-click the `KompoZer.exe` file.

Access KompoZer and find the `start_template.htm` file in the `EPIDATA_REPORT\DOCS\EN` sub-folder of the project. It is an empty page with a title:



That it is not quite empty becomes clear if you tick at the bottom to "Source":



In fact, we have taken the normal `start.htm` file that comes with the installation of EpiData Analysis, have stripped it down and added a few essentials to prepare it for an interactive menu (template courtesy, Jens M Lauritsen, April 2008) and saved it under this `start_template.htm` file name.

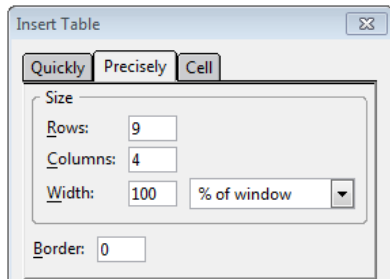
The first thing is to save it as:

`start_tb.htm`

Remember that we direct the EpiDataStat.ini file to go to this file when initiating:

```
set start page="..\docs\en\start_tb.htm"
```

You will be working in the “Normal” tab and the first thing we do is to insert a table with 4 columns and 9 rows (you can count them in the screenshot of the final interface shown at the beginning). Choose “Precisely”:

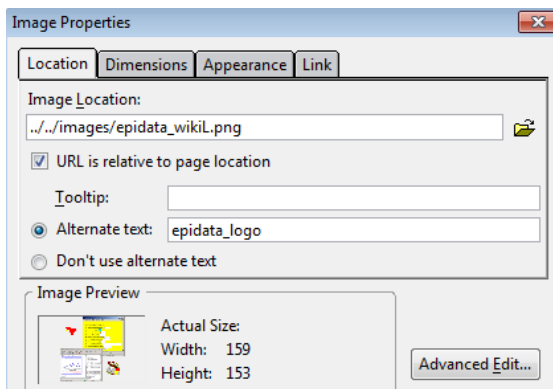


and set the Border to “0”. The default for “Border” is 1: a line around each cell is displayed in the browser. Setting it to “0” allows entering the information into cells instead of using the Tab key (which we cannot do properly in an HTML page), but the user does not see any line and remains unaware that we used a table.

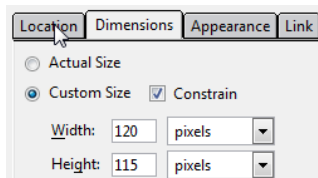
If the table cell background is not white, you may have to fiddle with the options for them to render them white (non-transparent). The best way is to change the background as follows:

1. Put the cursor into the top left cell
2. Right-click, choose “Table select” | “All cells”
3. Right-click again, choose “Table or Cell Background Color”
4. Pick the color (in our case “white”)

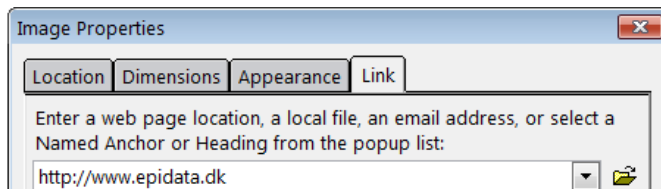
In the most upper left cell we insert one of the provided EpiData logos (the `epidata_wikiL.png` file from the `\IMAGES` sub-folder) using the “Insert” menu and also supply a (required) alternate text for the name:



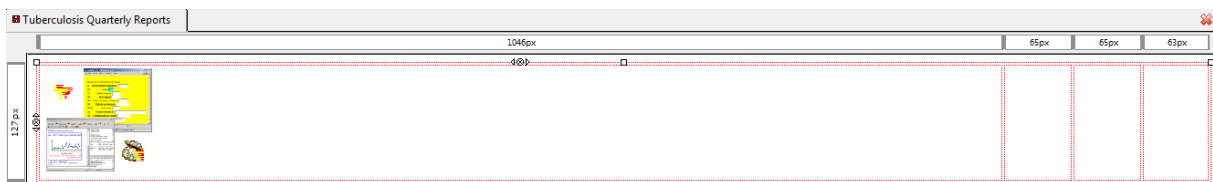
Then go to the tab “Dimensions” and decrease the current size of the width to 120 pixels:



In the “Link” tab add the URL of the EpiData website:



Then accept and you get:

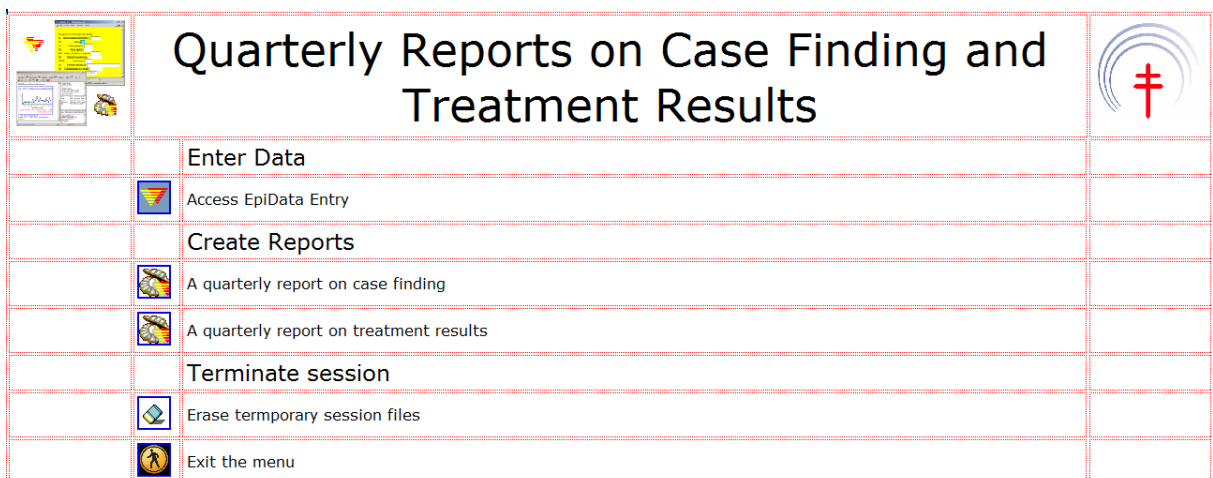


Never mind for now the distortion of the column width. Add the Union logo into the most upper right cell and make a link to the Union website (<http://www.theunion.org>) in the same manner.

Join the central two cells of the first row, add the text and format it (see beginning of this Exercise) to get:

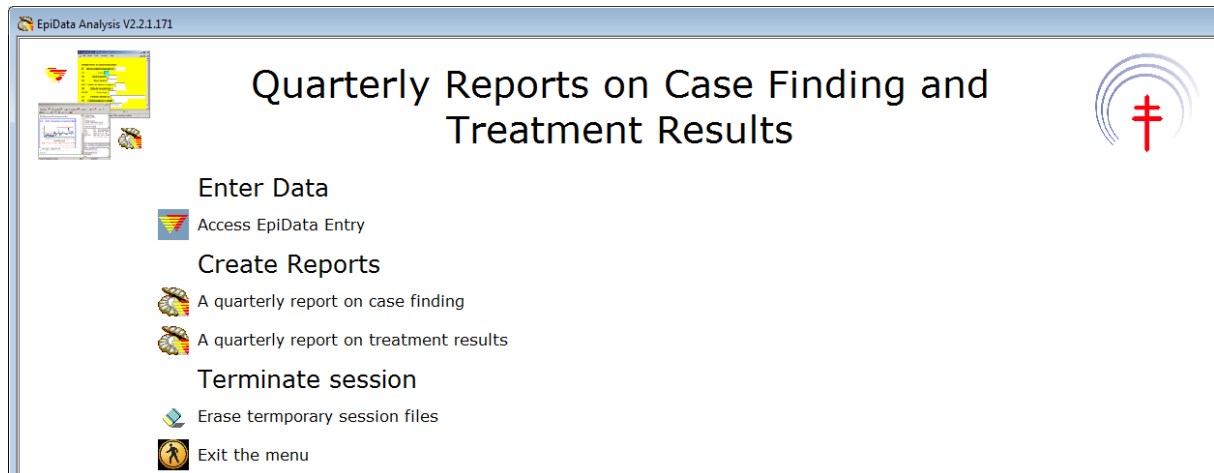


Continue adding text, formatting it, and add the appropriate icons into the appropriate places until you have the complete lay-out:



Make sure to save the file. Perhaps you want to look at the source code and see that a lot has been added. We will not further edit the source code here, we will do this in a text editor. Thus exit KompoZer, this is all we are going to do with it.

If you now click the `EpiDataStat.exe` file, you will get:



Of course, there is no functionality yet (except the EpiData and Union web site icons if you are on the internet). In the next step we add functionality to the other icons.

Editing the HTML file

Windows has an inbuilt text editor, NotePad™. There are several free text editors available that are more flexible and powerful than this one. We have selected Crimson Editor® as our choice for this course. This free text editor has several powerful features such as showing HTML code in colors and providing the very useful feature for rectangular selections.

Open now in this text editor the `start_tb.htm` file. What you see may seem a bit overwhelming and we will thus take it apart into manageable pieces to understand what we find there and to edit it where appropriate.

You have seen before the first few lines:

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
2 <html><head>
3 <meta content="text/html; charset=ISO-8859-1" http-equiv="Content-Type">
4 <meta name="copyright" content="EpiData Association">
5 <meta name="description" content="Introduction to EpiData reports"><title>Tuberculosis Quarterly Reports</title>
```

They inform us that we deal with an HTML file and that the head starts. If we collapse (and hide from view) lines 9 to 28, we see up to line 28:

```

1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
2 <html><head>
3 <meta content="text/html;charset=ISO-8859-1" http-equiv="Content-Type">
4 <meta name="copyright" content="EpiData Association">
5 <meta name="description" content="Introduction to EpiData reports"><title>Tuberculosis Quarterly Reports</title>
6
7 <style>
8 <!--
28 </style><!--<link href="epiout.css" rel="stylesheet" type="text/css" media="screen">--></head>

```

You note here the beginning and end tags for HEAD and those for STYLE embedded inside these:

```
<head><style>...</style></head>
```

Make two hard carriage returns after the closing `</head>` tag, so we see a bit better what we have next. Make another hard carriage return after the end of:

```
<body class="bodywhite">
```

so that we separate for visibility the beginning of the table definition:

```
33 <table style="text-align: left; width: 100%; border="0" cellpadding=.....
```

Can you see where our first row begins and where it ends?

```

<tr><td style="background-color: white;"><a href="http://www.epidata.dk"></a><td colspan="2" rowspan="1" style="background-
color: white; text-align: center; width: 957px;"><big><big><big><big>Quarterly Reports on Case
Finding and Treatment Results</big></big></big></big></td><td style="background-color:
white;"><a href="http://www.theunion.org"></a></td></tr>

```

The row begins with the opening tag `<tr>` and its ending tag `</tr>`. Each cell embedded in the row has its beginning and end tags `<td ...></td>`:

	Begin tag	End tag
Row	<code><tr></code>	<code></tr></code>
Cell	<code><td></code>	<code></td></code>

If we isolate the first cell with the logo of EpiData, we have thus:

```

<tr><td style="background-color: white;"><a href="http://www.epidata.dk"></a></td>

```

```
<a href ... </a>
```

is the opening about everything dealing with this logo: first the link to the web site, then information about the cell style, the dimensions of the logo, the alternate text, and finally the reference to the name of the image and the place where it is relative to the current position:

```
src="../../../images/epidata_wikiL.png"
```

Thus, in summary:

The definition of the image is defined as:

```
src=
```

and the image itself is given with its name relative to the location it had when you inserted it:

```
"../../../images/epidata.png"
```

We have no links to any of the programs that should be run when the user clicks on the EpiData Entry image defined as:

```
src="../../../images/epidata.png"
```

because there is not yet any instruction on what to do here. We are going to do that in the next step.

Opening EpiData Entry from within EpiData Analysis

The task is to open EpiData Entry from within EpiData Analysis. To this end, we write HTML code associated with the icon for EpiData Entry:



and add the code in the correct place. Locate the HTML code that gives information about the embedding of this icon:

```

```

(You may not have `border="0"`, this removes the blue lines around the icon that designates it as a hyperlink).

Actually, the above information about the icon is located in one single cell, within the tags `<td> ... </td>`:

```
<td style="background-color: white;"></td><td style="background-color: white; width: 42px;"></td>
```

The instruction to access EpiData Entry and, once there, to open a specific REC file is placed immediately before the definition of the icon ``. The following text is placed there:

```
<a href="../../../epidata.exe ..\originals\sample_2006.rec">
```

The `href` is a hyperlink to a reference, here to “`epidata.exe`”, the executable file that opens EpiData Entry. Why the “`../../../`”? The `start_tb.htm` is located two levels down (`\en\docs`), therefore the instruction must be to go two levels up to be in the root of the “`epidata_report`” as the `epidata.exe` file is in the root. As you have observed on the internet, HTML uses forward slashes to designate paths and locations.

Once EpiData Entry is opened, it requires “normal PC” behavior in designating paths, thus we need backward slashes. The file we are requesting to be opened, `sample_2006.rec`, is located one level down in the `EPIDATA_REPORT\originals` folder.

After we insert the above code, we must also place an ending `` tag to close the opening `<a href` before the cell closes with `</td>`:

```
<a href="../../../epidata.exe ..\originals\sample_2006.rec"></a></td>
```

Test functionality after saving.

Writing HTML code to invoke an EpiData Analysis program when clicking on the icon

Quite similar to the above, we locate now the HTML code for the EpiData Analysis icon:

```

```

Again as above, we need to insert HTML code immediately preceding the `<img src=...`. What is different is that we are already in EpiData Analysis, so it is not necessary to execute it (it would actually give an error if the default is set to run only one copy of EpiData Analysis at a time). All we need to do is to make a hyperlink to an EpiData Analysis program, specifically here to the program that should produce the first quarterly report on case finding. We will call the program `sample_1_quarterly_report.pgm` and it will be located in the folder PGM. The command to run a program is "run". We would thus think that:

```
href=run "..\pgm\sample_1_quarterly_report.pgm"
```

should do it. It is, however, not quite sufficient as the interplay between HTML and EpiData Analysis requires two additional things:

- 1) an opening command "epi:" to give the indicate that EpiData Analysis instructions will follow
- 2) everything related to EpiData Analysis must be placed between single quotes.

Accordingly, the above becomes:

```
href='epi: run "..\pgm\sample_1_quarterly_report.pgm"'
```

We will add more EpiData Analysis commands, each separated by a semi-colon, like:

```
href='epi: CLS; set echo=off; run "..\pgm\sample_1_quarterly_report.pgm"'
```

Once the program has run and produced the output, the user should get an instruction to press F8 which refreshes the `start_tb.htm` file (thus gets the user back to the main menu interface):

```
href='epi: CLS; set echo=off; run "..\pgm\sample_1_quarterly_report.pgm" type "F8: Go back to menu"'
```

Including the opening `<a href` and the closing `` tag and showing the entire content of the cell, we then get:

```
<td style="background-color: white;"></td></tr><tr><td style="background-color: white; width: 42px;"><a href='epi:CLS; set echo=off; run "../../../pgm/sample_1_quarterly_report.pgm"; type "F8: Go back to menu"'></a></td>
```

Summarizing some of the specific EpiData Analysis commands that we may use in HTML:

Action	EpiData Analysis command
Clear the screen and memory	<code>epi:CLS</code>
Show only output, not the running of the program	<code>set echo=off</code>
Do not display the main menu	<code>set display mainmenu=off</code>
Do not show the process bar	<code>set display worktoolbar=off</code>
Do not show the command line	<code>set display command prompt=off</code>
Run the program	<code>run "../pgm/run_entry.pgm"</code>
Tell the user how to return to the main menu after the program has completed	<code>type "F8: go back to the main menu"</code>

This summarizes the principles we use to connect (making a link to) the execution of an EpiData Analysis program to an icon. While the HTML code is now correct, nothing will happen yet because the program file does not yet exist.

Preparing the EpiData Analysis program

Reading the data file

We remember that the starting default folder is the `\epidata_report\temp` folder. After the standard closing of any open file, the path must thus be changed to the folder in which the data files are located, the `\epidata_report\originals` folder:

```
cls
close
logclose

cd ..
cd \originals

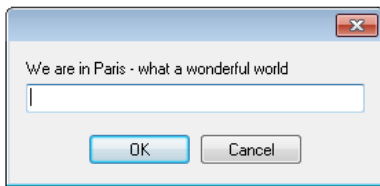
read "x.rec"
```

Interactive prompting

During the running of a program in a menu, there should be options to choose from. Depending on the selection the user makes, the program will then continue one or the other way(s). The program is thus to stop at a certain point and prompt the user for input. This is accomplished by typing something between two question marks. The following:

```
?We are in Paris - what a wonderful world?
```

gives you a pop-up box:



There is no option here, obviously, it is just a statement. To expand interactive prompting to include an option, we make use of a constant (global “variable”). To illustrate it, we are using the Uganda laboratory dataset (`uganda.rec`, included in the `originals` folder), collected over three years with a variable `REGDATE` denoting the registration date. We want to give the user the option to make a frequency of the field `SEX` for a selected year:

```
cls
close
cd ..
cd originals
read "uganda.rec"

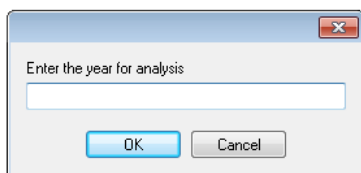
set echo=off
gen i regyy=year(regdate)
define yr ##### global
cls

set echo=on
freq regyy
  yr=?Enter the year for analysis?
set echo=off
  select if regyy=@yr
  cls
  freq sex
```

We first extract the registration year `REGYY` from the registration date. Then we create a constant `YR`. Then we display the frequency (`FREQ REGYY`) of the registration year and get:

regyy	
	N
1999	17300
2000	18662
2001	18088
Total	54050

The user is then immediately prompted to enter the year:



If, say, the year 2000 is entered, then the output is:

Sex of examinee	
	N
Female	7745
Male	9326
Missing	1591
Total	18662

Note the:

```
select if regyy=@yr
```

where we make use of the constant YR.

We can use the same principles, but expand it to provide an option to do either one thing or another:

```
define yesno # global

yesno=?Do this or that: 1=Do this - 2=Do that?
imif yesno=1 then
  (do something)
else
  (do something else)
endif
```

The user then enters a “1” or a “2” into the box.

We can use a nested sequence of events. If we take the Uganda dataset again and we want to give the choice to first select the year, and then have the option of analyzing the entire selected year or only a quarter of it, we write (this sample program `imif_example.pgm` is available in your PGM folder):

```
set echo=off
gen i regyy=year(regdate)
label regyy "Registration year"
gen i quarter=month(regdate)
label quarter "Quarter of the year"
recode quarter 1-3=1 4-6=2 7-9=3 10-12=4
labelvalue quarter /1="Jan-Mar"
labelvalue quarter /2="Apr-Jun"
labelvalue quarter /3="Jul-Sep"
labelvalue quarter /4="Oct-Dec"

define yr #### global
define yesno # global
define q # global
cls

set echo=on
freq regyy
  yr=?Enter year for analysis?
set echo=off
  select if regyy=@yr
```

```

cls
yesno=?Analyze: 1: Whole year; 2: One quarter only?
imif yesno=1 then
  cls
  type "You selected: Registration year @yr" /h2
  freq sex
else
  freq quarter /v1
  q=?Select quarter?
  select if quarter=@q
  cls
  type "You selected: Registration year @yr and Quarter @q" /h2
  freq sex
endif
set echo=off

```

In the first block, we extract registration year and registration month and recode the latter into quarters.

In the second block, we define the three constants.

In the third (last) block, we allow selection of the registration year:

```

freq regyy
  yr=?Enter year for analysis?
  select if regyy=@yr

```

Then based on the selected year, we ask to choose the quarter

```

cls
yesno=?Analyze: 1: Whole year; 2: One quarter only?

```

After choosing between “Whole year” and “One quarter only”, we make use of the `imif` command which is closed with an `endif` command:

```

imif yesno=1 then
  cls
  type "You selected: Registration year @yr" /h2
  freq sex
else
  freq quarter /v1
  q=?Select quarter?
  select if quarter=@q
  cls
  type "You selected: Registration year @yr and Quarter @q" /h2
  freq sex
endif

```

IMIF is used to divert course in a `pgm` file depending on parameters which can, as done here, be acquired by “? ... ?”.

We use the `type` command to inform the user of the selection made. The `/h2` is HTML language for header size.

We will store all EpiData Analysis programs in the `\epidata_report\pgm` sub-folder. Again, the present position is a sub-folder of `EPIDATA_REPORT` and we always want it to be in a sub-folder at the beginning, so that the command:

```
cd ..
```

gets us one level up to the \EPIDATA_REPORT but never out of that “container” (thus the relative path will never be violated and get into conflict with the drive or folder). As the data are all located in the \EPIDATA_REPORT\ORIGINALS subfolder, yet we are currently in the EPIDATA_REPORT\TEMP subfolder, before reading a file, we must thus always change the folder accordingly:

```
cd ..  
cd originals
```

This get's us from \EPIDATA_REPORT\TEMP one level up to \EPIDATA_REPORT in the first line, and then one level down to \EPIDATA_REPORT\ORIGINALS in the second line.

Making the EpiData Analysis program to produce the quarterly report on case finding

You will make a total of three programs:

```
sample_1_quarterly_report.pgm  
sample_2_quarterly_report.pgm  
cleanup.pgm
```

We have two actual datasets of a random selection of tuberculosis case registers from the years 2004 and 2005 from Viet Nam. They were provided by courtesy of Dr Nguyen Binh Hoa from the National Tuberculosis Program Viet Nam. They were slightly edited from the original case registers in that the original identifier was removed and replaced by an artificial one. The names of the 30 units were also changed to have non-reality units. Several variables that were collected were also removed and some other minor modifications were made. However, overall, these data are real.

There are three data files:

```
sample_2004.rec  
sample_2005.rec  
sample_2006.rec
```

The last file (sample_2006.rec) contains no records and is used for data entry only. To facilitate selection, we added one more file:

```
choice_year.rec
```

containing just three records with the three registration years respectively. This can be used to prompt the users for the selection of the analysis year.

The task is to produce a quarterly report on case finding. We follow here the guidelines of The Union for the quarterly report. It requires that all notifiable cases in the quarter are reported. We will get back to this shortly.

Because we set `echo=off`, the screen will stay blank during the running of the program. As this may confuse the reader, it will be useful to insert after any CLS command a line that informs the user that despite the blank screen something is happening, like:

```
cls  
type "Be patient...files are identified and prepared" /h2
```

The following two parts are required for the quarterly report as recommended by The Union:

ALL CASES REGISTERED IN THE QUARTER

SMEAR-POSITIVE				SMEAR-NEGATIVE		EXTRA-PULMONARY	TOTAL
New cases	Relapses	Treatment after failure	Treatment after default	< 15 yrs	15 + yrs		

NEW SMEAR-POSITIVE CASES ONLY

Age group (years)													TOTAL			
0-14		15-24		25-34		35-44		45-54		55-64		65 +		Male	Female	Total
M	F	M	F	M	F	M	F	M	F	M	F	M	F			

A note on the notifiable cases: Patient categories “Transfer in” and “Other” must not be reported.

When you define your variables, note that not all registered cases may always have the required information. For instance, you need three variables to determine “SMEAR-NEGATIVE, <15 yrs”: The case must be pulmonary, you must know the case is smear-negative, and you must know the age. Take this into account when deciding what you are going to present the analysis.

If you want to produce a graphics output of two graphs and show them side by side, you must first save the graphs, e.g.:

```
bar repdef2 \
    /ti="All cases" \
    /save="all_cases_1.png" /replace
cls

select repdef2<>0
select repdef2<>9
bar repdef2 \
    /save="all_cases_2.png" /replace
cls
```

Among the required files was the HTML file “side_by_side_graphs.HTML”:

```
echo <table><tr><td colspan=2 border align=center >
<font color=black face="Arial" size="4">Some title that crosses over both
graphs</font></td>
<tr><td></td><td>
</td></table>
```

Replace the example names with the actual names of your graph files and then the simple command in EpiData Analysis:

```
show "side_by_side_graphs.html"
```

will show them side-by-side.

Other programs

The process is analogous for the quarterly report on treatment results. You can use the same basic program up to the point where you produce the actual output. You may consider two outputs, one for all cases, and one for sputum smear-positive cases only.

It might be useful to have a separate program to erase all files created during the session, so that only the files remain that are needed to start anew.

Command to exit the program

To exit the program, you do not need a special program. The EpiData Analysis command for the HTML file to quit the program without asking for conformation is:

```
<a href='epi:exit'> ... </a>
```

Opening the menu

To make things as simple as possible for the user, not requiring search for the `epidatastat.exe` file, you may add a batch file (that might be named `start.bat`, the `*.bat` extension being mandatory) at the same level as the `\EPIDATA_REPORT` folder. This batch file follows simple conventions (see internet for more on writing batch files):

```
cd epidata_report
start epidatastat.exe
```

Zip both the `\EPIDATA_REPORT` folder and the `start.bat` file into one single zip file (any name like `anyname.zip` will do of course), send it by email to the user with the instructions:

- 1) Unzip the `anyname.zip` file to any place on your computer
- 2) Double-click the `start.bat` file to open the menu
- 3) Start working

Task

Produce a single folder containing the EpiData Entry and Analysis programs, including the database of with an interface that permits by simple clicking to enter data or to run two standard reports, giving the user the option of choosing which country, year, laboratory, etc to analyze and is transferable to any drive or folder, with a total zipped file size of just 3.81 MB.

A user will be able to unzip this file onto any drive or folder, double-click the `start.bat` file, and be in the menu interface. Test it out by zipping it and then unzipping it onto a flash USB stick.

Solution to Exercise 4: Creating a menu for standard reports

Key points:

- The EpiData Analysis interface is largely based on HTML which gives the user great flexibility in customizing it to the needs
- A menu can be written for a user that requires not skills in EpiData Analysis and allows producing standard summary reports by simply clicking like on any web site

Task

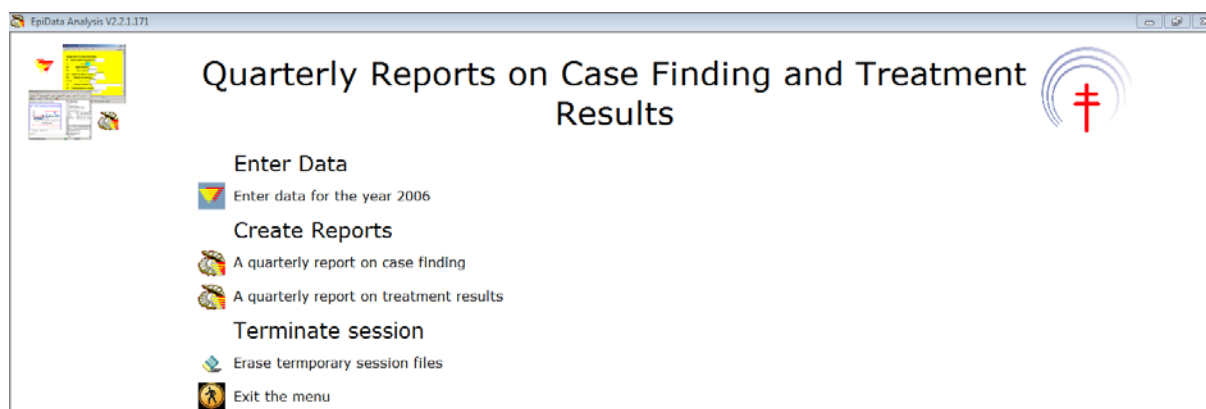
Produce a single folder containing the EpiData Entry and Analysis programs, including the database of with an interface that permits by simple clicking to enter data or to run two standard reports, giving the user the option of choosing which country, year, laboratory, etc to analyze and is transferable to any drive or folder, with a total zipped file size of just 3.81 MB.

A user will be able to unzip this file onto any drive or folder, double-click the `start.bat` file, and be in the menu interface. Test it out by zipping it and then unzipping it onto a flash USB stick.

Solution

The entire folder is available on the course web site as a zip file.

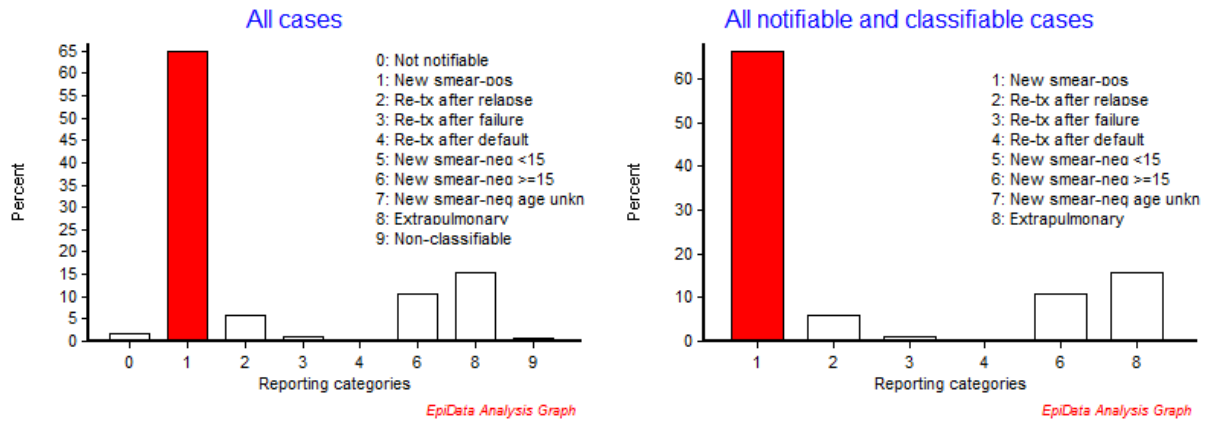
The interface the user sees:



In the following just select output graphs are shown.

The output graphs from the first program:

Quarterly report on case finding



The output graph from the second program:

Quarterly report on treatment outcome

