Exercise 7: A relational database

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

a. Understand when a single data entry form and when a relational database is the preferred data collection instrument

b. Create a relational database for a varying number of observations.

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	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	Cohabitating	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+

This type of a register requires a different approach to data entry 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

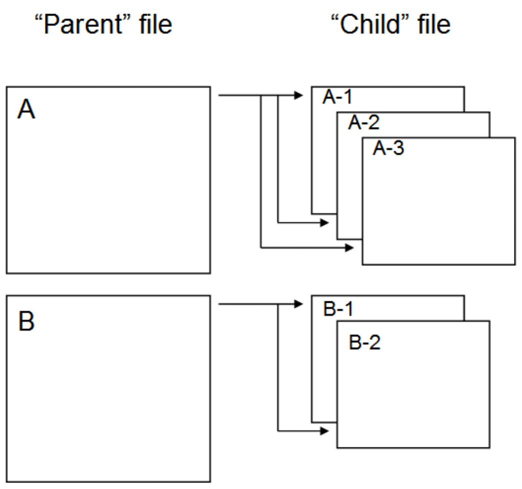
Some patient characteristics do not change over time such as, in this example, the identity of the patient, sex, and marital status (well, perhaps not during these short intervals). Others do change, such as the date of examination, blood sugar, the aspect of the sputum and the sputum smear examination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

To capture such information in a single data entry form as was done in the previous exercises 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 usually be Must Enter fields.

Rule: *If an individual has a fixed number of observations for each variable, then a single EpiData form 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 deciding which information is static for an individual (during the observation period) and which information changes over repeated observations. We will illustrate a relational database with a very limited set of variables.

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, we may have different visits to a clinic where each time the date of the visit is recorded, and blood pressure blood sugar 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 variables in 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 records and those at Level 2 child records.

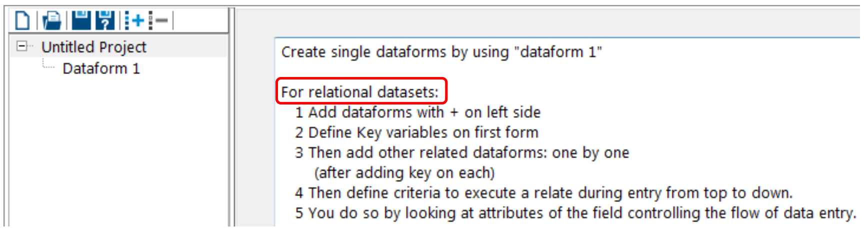
To build the relational database, we collect different information at each level and link the two levels, so that at some point during entering parent information will trigger the opening of the child form to enter one or more child records. One or more child records are added until one wishes to return back to the upper level of parent records.

Important components for the structuring of the relation between the two data forms are:

- The two forms have a common identifier field. This identifier might be called `idparent`.
- The child form must have its own unique identifier, which might be called `idchild`.

Illustrating it practically in EpiData Manager

We start a New Project. We note the brief information on the relational database in the opening window:



Create single dataforms by using "dataform 1"

For relational datasets:
1 Add dataforms with + on left side
2 Define Key variables on first form
3 Then add other related dataforms: one by one
(after adding key on each)
4 Then define criteria to execute a relate during entry from top to down.
5 You do so by looking at attributes of the field controlling the flow of data entry.

Change the name of the Untitled Project to Exercise 7 and Dataform 1 to 1_parent and save the project as `a_ex07.epx`.

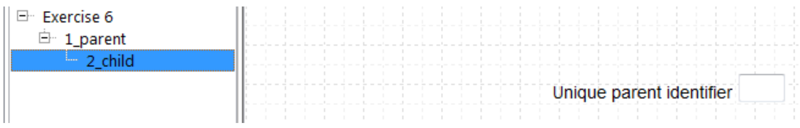
We add as first field the string variable `idparent`, the `Unique parent identifier` of length 2 as a Must Enter field. We add only two fields: `age`, an integer of length 2 for `Patient's age in years` and `sex` for `Patient's sex`, an integer of length 1. Both are Must Enter fields. For `age` as a continuous variable, we might write in Notes:

```
Enter 0 to 97 for exact age
Enter 98 if 98 or older
Enter 99 if not recorded
```

Finally, we define `idparent` as Key field. Next we make the 2_child Dataform (rename it using **F2**) by clicking on the:

:+:

And we note that EpiData opens the new data form with the `idparent` identifier already written onto the canvass:



If you click on it, you see that in the Field Properties menu everything is grayed out and in its Extended tab we also see the grayed out setting to No Enter, all courtesy of EpiData Manager!

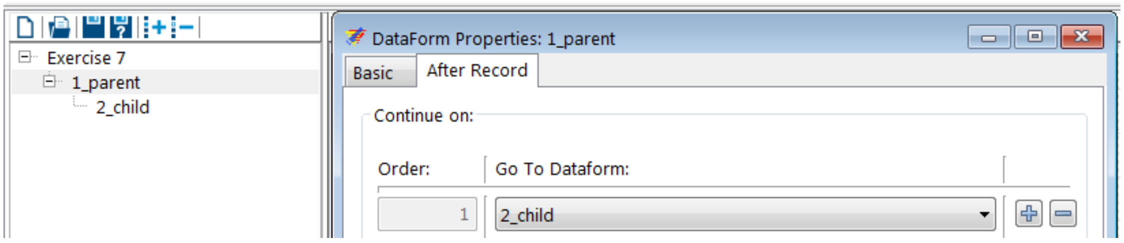
While we are in the `2_child` data form, we add the four variables `visit`, `syst`, `diast`, and `bs`, for `Visit date`, `Systolic blood pressure`, `Diastolic blood pressure`, and `Plasma glucose in mMol` respectively, a date, two integer and one float field, all Must Enter (and define some range, but assume no missing values). Preceding

Visit date, add idchild, the Unique child identifier, a No Enter string field of length 13, resulting from the combination of the values of idparent and visit.

This is all there is to it: the flow of the related forms is such that once the 1_parent is completed, the flow leads right over to the 2_child data form, where you can complete from one to many records before you move back up to a new 1_parent data form.

Other options

You can add more related forms. If there is more than one data form, you can choose under Properties of the 1_parent data form (right-click its name) in the After Record tab the order of flow which is to be followed after the parent record:



For a single related form as in our example, there is only one possibility, and this is pre-recorded as default.

You can also set conditions in a field of the 1_parent form, i.e. which value in the field should trigger access to the related field. As of now, there is still some bug that should be fixed shortly that prevents this extended functionality to work as intended.

Task:

o ***Complete the related data set and check it for functionality in the EpiData EntryClient***