Report 1 :

How to add data in MODE-LD database?

Author :
Christophe Moreels

March 2014

Contents

1 Introduction

The goal of this document is to explain in a very practical way how to add information in the database of MODE (LD project).
It is particularly useful for MODE-LD staff, but can also be read by any person who would be involved in this database.
This document will be very precise: it often uses examples, and a lot of screen-shots are included, so that you can compare your screen with what you should have if everything was OK.

Let's insist at this point that the database can look very difficult at a first point, but when reading the document and practising a little bit, it is at the end very easy to use.

Before starting the explanations, let's note different important global remarks:

- For security reasons, it is not possible for MODE staff to remove or to modify data in the database. This means that you should be very careful when adding data in the database. Don't go too fast, and read your data a second time before validating the forms. If you did a big mistake, you can ask the project manager, LD, or myself to correct the data.
- If the forms don't react dynamically (example: the list of communes doesn't update when you select the district), it should be because of a *Javascript* error. Javascript is a program that is used to make dynamic forms. If you think javascript is correctly installed on your computer, try to use another web browser. *Mozilla Firefox* should work fine, while *Google Chrome* has some problems.
- The database is a tool that can help MODE to manage their data, but it will not solve all the problems. In particular, if MODE staff doesn't complete the database regularly and correctly, the data will be wrong. In that case, the database is not useful anymore...
- People who add information in the database are not necessarily the same as the people who search information in the database, for example to write the reports. People who add the information should do it regularly and in a correct way, so that the people who write the reports can rely on good data.

This document focuses on how to add data correctly in the database. A second document will explain in detail how to search useful information in the database, in order to add interesting data in the reports of MODE-LD.

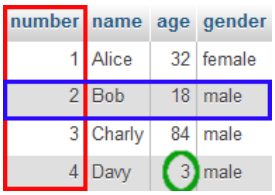
2 Overview

Before we start with MODE's database, it can be useful to quickly see what is a database, and why it is really interesting to use it.

2.1 What is a database?

A database is an organized collection of data. It can be viewed as a table, where each line corresponds to an *instance*, and each column is a *label*. Let's consider an easy example in the figure 1. This example stores instances of clients, each consisting in 4 labels:

- A unique number to identify each client
- The name of the Client
- His age
- His gender



number	name	age	gender
1	Alice	32	female
2	Bob	18	male
3	Charly	84	male
4	Davy	3	male

Figure 1: Example of a simple database

This example is a database containing 4 *instances* (blue), and 4 *labels* (red). A *data* is a single element in the table, like shown in green.

2.2 Interest

An interesting question is to ask: *Why is it useful to use a database?*
Indeed: data could also be stored on paper forms, or in an simple excel table. The answer is that with a database, you can easily sort the data, and view only specific information.

As an example, imagine you have more than 1000 instances of clients, and you would like to know how many are males, and how many are females.
With paper forms, the only way to proceed would be to count each form 1 by 1, which takes a lot of time...

With an excel table you could do it more efficiently, but it requires to know which *command* to write, which can be complicated to people who don't have knowledge in computer science.

With a database, all you have to do is to specify what information you want to display (which is very easy), and the output appears automatically for you. No computer science is needed, the user should only be able to complete a *form*.
To make things clear, let's take our previous example: the database of clients. Imagine that this time the table contains not 4, but 1000 instances of data, which make it very difficult to count 1 by 1 the lines in the table.
Figure 2 shows how a user can easily get only the clients that are *males* (so not the females).
Let's briefly explain the 3 steps (shown in colour pink) of this diagram.

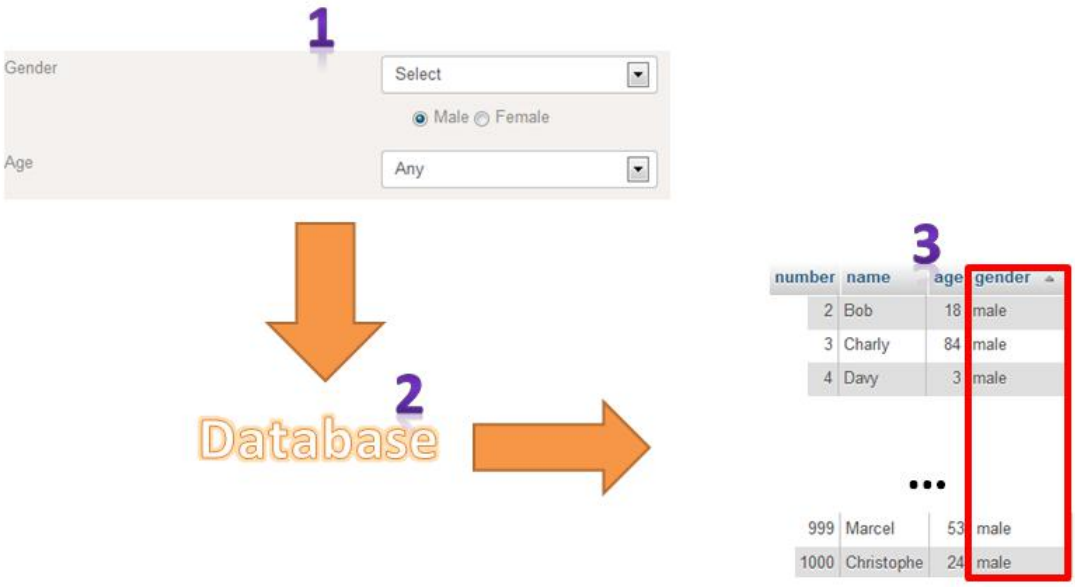


Figure 2: Example of search in database: only displays males

1. The first step is done by the user. He has to say to the database that he only wants to see the male clients. To do so, he doesn't need to know a lot about computer science: he just complete a form. In this case, he clicks on *male*, and put *Age* field to *Any*, because he doesn't care about how old the beneficiaries are (he wants to see all the ages, given that the clients are males). When he has complete the form, the user just click on a *Send* button, to send the information to the database.

2. The second step concerns the internal program of the database. It will receive the information (here: *The user only wants to see the males clients*). The internal program will do a lot of work to sort the data, but the user doesn't care about how that work. All what is important for him is to get the result.

3. After the program finishes running, it send the useful information to the computer of the user. Here he will receive the list of clients containing only the males.

Like you can see in this example, using a database is very easy (no strong skills are required), and it is very powerful since it can adapt all the information you need for you. The process is also very fast: the program runs in a few seconds. If the user had to count the data 1 by 1, it would take him hours or days and would probably do errors wile counting...

2.3 What the user has to do

The database can do a lot of things alone, thanks to a program that controls it. For example, when a new beneficiary is added, the database generates a unique code number, which will be associated to that beneficiary. The user doesn't need to manage those codes, which ensures that every code is unique.
But the program cannot work completely alone: it needs a user (you), to know what to do. In particular, the user will have 2 big missions:

- Add new data in the database
- Search particular data in the database

The first point consists in describing new data, in order to add new instances in the database. This will be done using a form (see section 2.4).
The second points consists in asking for particular instances of the database. Like we saw in figure 2, we can for example ask to see only the male clients. This is called a research with *constraints*, and is also done by completing a form.

This report focuses on the first point: how to correctly add information in the database. To learn how to display useful data from MODE's database, please read the second report, which focuses on that second point.

2.4 Forms

This section will explain in detail what is a form. In particular, it explains how a user can use it to give different kind of information to the database.
There exist a lot of different kind of information that the user can send to the database. In our example of clients database (see figure 1) there are 3 kinds:

1. Numbers are used for the age of the clients, and to identify them (label: number).
2. Text: used for the name of the clients
3. Finite values: used for the gender of the clients. In this case there are only 2 possible values: *Male* or *Female*.

Depending on the kind of information, you will use different ways to input the data.

2.4.1 Text inputs

The first is to enter a text. This should be done using the keyboard. It is useful for example to enter names.

Figure 3: Example of text input to input the name of beneficiaries.

2.4.2 Radio buttons

A radio button is a little circle that can be selected or not (if the circle is filled, it means it is selected). The big particularity of radio buttons, is that you can only select 1 value. In the example of gender, the user can only select *Male* or *Female*, but not both. If you first select male, and after you select female, the male button will be unselected again. Note also that you **must** select 1 information. If you don't select any option, the form will return an error.

Figure 4: Example ofradio button to indicate the gender of the beneficiaries.

2.4.3 Check boxes

Check boxes work like radio buttons, in the difference that you can select multiple data, or no data at all. For example: the beneficiaries of MODE can have different activities. It is so a good idea to put check boxes to select the different activities done by the beneficiaries of the project. If the beneficiary does multiple activities, you can check multiple check boxes. If he does no activity at all, you can leave all the check boxes empty.

Figure 5: Example of check boxes to select the health desease of the beneficiaries.

2.4.4 Datepickers

Another information that is often used in databases are dates. Datepickers seem like text inputs, but when you click on it, a calendar automatically displays. The user should then click on the date of the calendar, in order to select the date. Using the mouse, it is easy to switch to another month or another year.

Figure 6: Example of date picker to input the date of selection of a new beneficiary.

Important: The user should never type the date on the keyboard. The incode format can be different depending on the form. For example: 4th January 2014 can be: 04-01-2014 or 1/4/2014. The user don't know the encoding, and if he choses a wrong one, the database could not understant and make big errors. By clicking on the date with the mouse, the encoding will automatically be the good one.

2.4.5 Text areas

Text areas are like text inputs, but are larger. They are useful to write big sentences. For MODE project for example, they are used to input remarks (which could be a few sentences long).

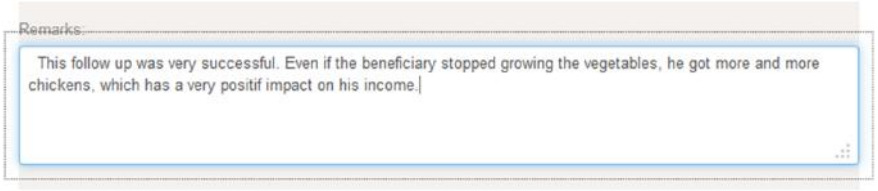


Figure 7: Example of text area to input a remark concerning a follow up.

2.4.6 Select lists

The last kind of information can be selection lists. Simply click on the arrow, and the list will drop down. By clicking on an item, you will then select it. This is useful when there are only a finite number of possible values that could be entered. In MODE's project for example, lists are useful to enter the income of the beneficiaries. This make sure that the data will be a number, and not a text for example.

Figure 8: Select list to put the age of the beneficiaries.

2.5 Danger: the *Other* field

When I created the database, I could not imagine how to project would evolve in the future. As an example, when you add a new beneficiary, you has to select which activities the beneficiary does by checking the activities. But what happens if the activity is not in the list?
If you could not add new activities, the database could quickly become up to date, and no more useful. For that reason, I decided to add a *Other* check box. When selecting it, you can specify a new activity. This is depicted on figure 9. When you add a new activity in this way, it will automatically be added in the

Figure 9: When the *Other* check box is checked, you can add a new activity in the list.

list for the next time, so that you don't need to add it each time a beneficiary does this new activity.
Important: If you add too many activities, the list will become very long. This is not acceptable. You should only add new data that are general, that means that can be applied for a lot of beneficiaries. Be very careful before adding a new data, and discuss it with MODE team before. If you want to remove an added data, ask the project manager, LD or Myself.

2.6 Where is the data stored?

Now that you know how to use forms, and so send information to the database, it can be interesting to learn where the data is stored. The are 2 different possibilities.

2.6.1 Storing data on a computer

The first method is very simple: all the data is stored in the memory of a computer. This is called a *local* database. A big advantage is that it is very easy to implement (programs like *Access* can make easy local databases).
A local database has also big limitations. First of all, if the computer is broken or stolen, all the data would be lost. Another technical limitation, is that only 1 person can work on the time on the database...

2.6.2 Storing data on an online server

The second possibility is to store the data on a server. First, let's remind what is a server.
A server is a computer with a lot of memory (thousands of Giga Bytes), which is accessible all the time using the internet. For example, the server used by MODE contains MODE's website and MODE's database, and it is located in Europe (not in MODE's office).
To reach the server, all the information go through the internet.
The big advantage of using a server, is that different users can access the data on the same time. For example: 3 staff of MODE can add new beneficiaries, each on their own computer. Another advantage is that the data is accessible from anywhere, not only from MODE's office.
There are also drawbacks in using a server: to access the data, an internet connection is required. If there is no electricity for example, it will not be possible to work on the database.
Another problem is that if there is no security, everybody can access the database, and so the confidentiality of the data is not ensured. That is why you will have to *Login* if you want to use MODE's database (see section 3).

Figure 10: Example of a server, which is accessed by 5 computers on the samed time

3 MODE's database

Now that we know what is a database and why it is useful to use it, let's have a look at MODE's database. In particular, this report explains in detail how to add information in it. A second report will explain how to use the information stored in the database, in order to use the data in reports for example.
This section will explain you step by step how to proceed to add information.

3.1 Connection to the server

MODE's database stores his data on a server (see section 2.6.2 for explanation on servers). The connection with the server is done on MODE's website:

www.modecambodia.org

When you are on the website, the first step is to login. This is done by clicking on the login link (see image 11).

The link brings to a connection page shown in figure 12. Simply write your username and password, and click on *Log in*. The login and password are not given in this document for security reasons. If you don't know the user/password, ask the staff of MODE.

Note that once you put the good login/password, your are connected for a duration of 1 hour. This means that after 1 hour, you will have to login again. If there is an electrical cut, you will have to login again afterwards.

If you are correctly logged in, you should now see a database menu on the website, like shown in red on figure 13.

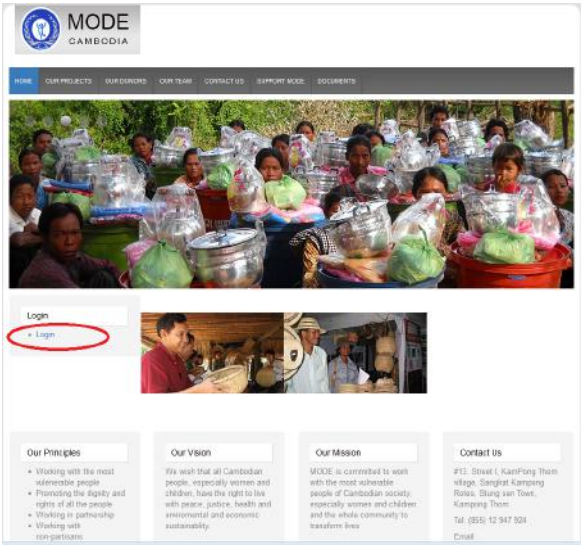


Figure 11: Login on the website: step 1

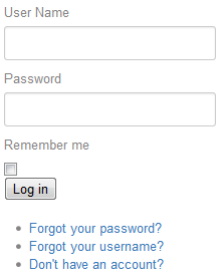


Figure 12: Login on the website: step 2

3.2 4 types of data

Now that you are logged in, it is very easy to add new information in the database.
Just click on the link *Add in database*, shown in green on figure 13. This brings you on a page containing 4 buttons, shown on figure 14. Each button brings you to a different page, used respectfully to add:

- A new beneficiary
- A new Follow up
- A new training
- A new self help group

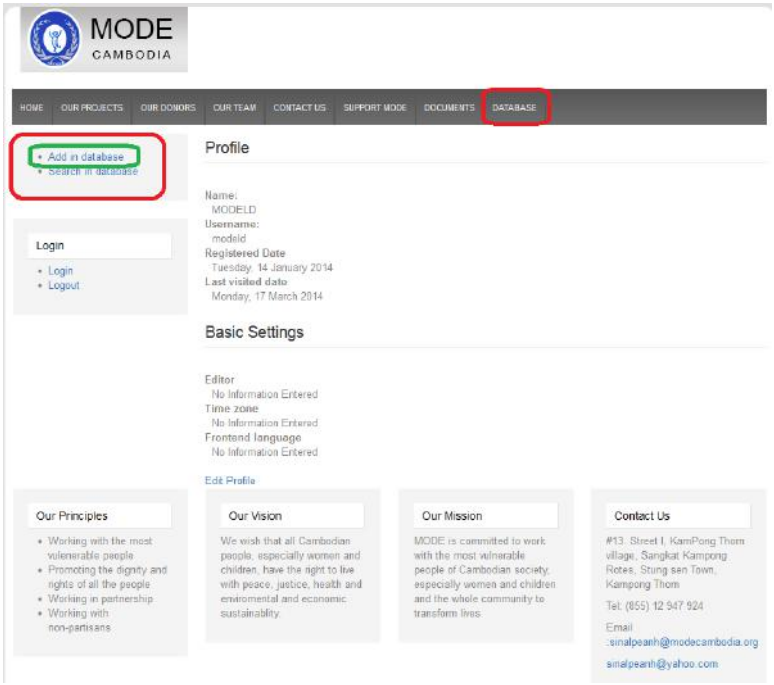


Figure 13: Website view when correctly logged in: database menu should appear.

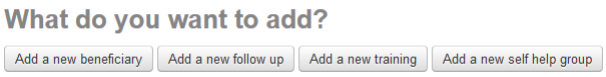


Figure 14: Main page to add data in the database

3.3 Adding a new beneficiary

To add a new beneficiary, first click on *Add a new beneficiary* button. This will bring you to a form that you need to complete. The form is divided in 4 pages.

3.3.1 Page 1

The first page is shown on figure 15.

The page contains the following labels to complete:

- **Type**: select the project of the beneficiary. IFS (Integrated Farming System) is for the agriculture project, IGA (Income Generating Activities) is for the selling project.
- **Name**: write the name of the beneficiary.

Figure 15: First page to add a new beneficiary

- **Gender**: select the gender of the beneficiary: Male or Female.
- **Age**: select the age of the beneficiary
- **District**: select the district where the beneficiary lives. After selecting the district, the commune list will be updated in such a way that only the communes of the selected district appears. This refresh can take a few seconds, please wait...
- **Commune**: After selecting the district, the list of communes should only contain the communes of the selected district. If this is not the case, please select the district again. When selected the commune of the beneficiary, the list of villages will update, in order to contain only the villages located in the selected commune. Again, the update of the village list can take a few seconds.
- **Village**: After selecting the commune, the village list should contain only the list of villages located in the selected commune. Please select the village of the beneficiary.

Remarks:

1. For the address: it is important to respect the order: first select the district, then the commune, and finally the village.
2. All the labels are required. If you let a label empty, you will not be able to go to page 2. Please correctly fill in all the fields!

Once you are finished, click on the button *Go to step 2*, which will bring you to the second page of adding a beneficiary.

3.3.2 Page 2

The second part of the form is depicted on figure 16.
There are here 5 information to complete:

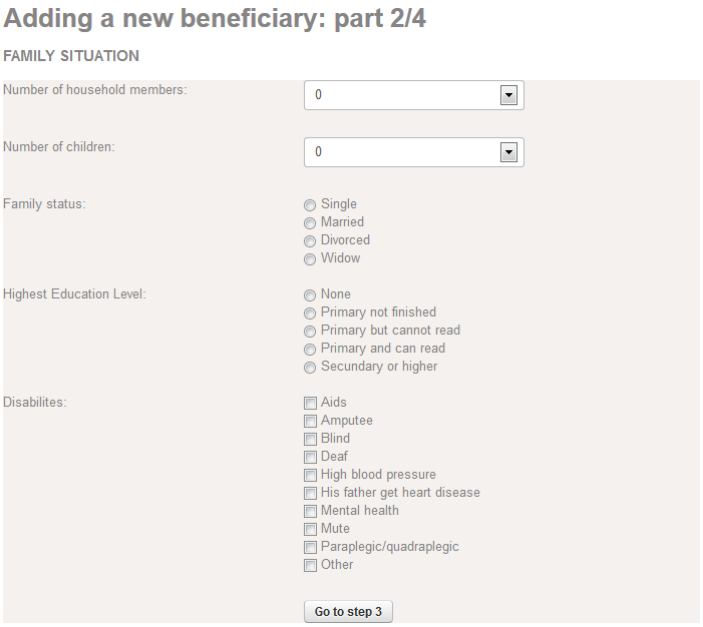


Figure 16: Second page to add a new beneficiary

- **Number of household members**: select the number of people living in the house of the beneficiary.
- **Number of children**: select the number of children of the beneficiary (0 if no children).
- **Family status**: Select the status of the beneficiary. Be careful not to confuse *Widow* and *Divorced*. Widow means that the husband (or wife) died, whereas divorced means that they voluntarily don't live together anymore (but they are still alive).
- **Highest education level**: Select the highest eduction level followed by the beneficiary.
- **Disabilities**: Select the health problems that the beneficiary could have. If the beneficiary doesn't have any health problem, don't select anything. If the beneficiary suffers from a disability that is not in the list, you can add a new disability by selecting *Other*. But please read section 2.5 before doing this.

Remark: Like for the first page, all the data must be completed. If you forget to complete a field, you will not be able to go to page 3.

Once you are finished, click on the button *Go to step 3*, which will bring you to the third page of adding a beneficiary.

3.3.3 Page 3

The third part of the form is depicted on figure 17.
There are here 3 information to complete:

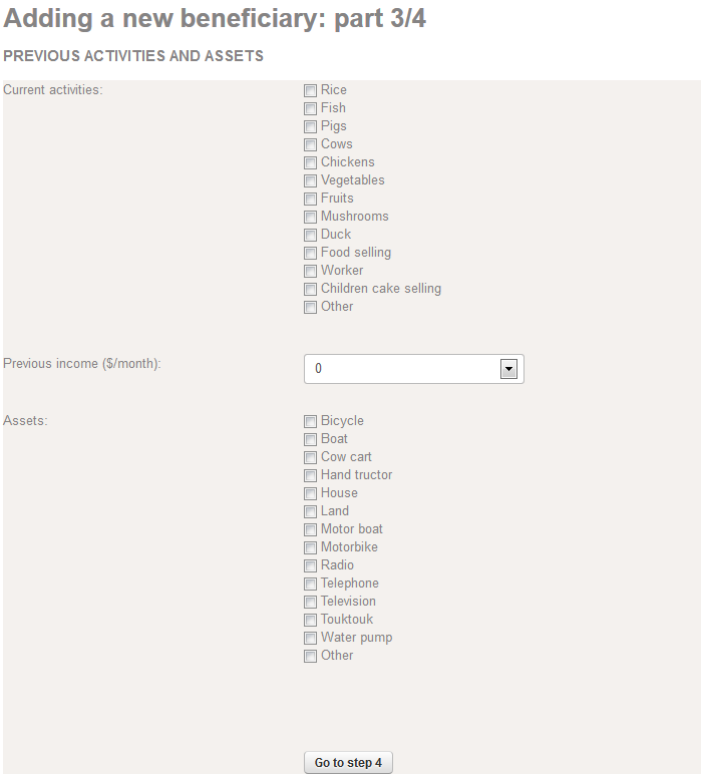


Figure 17: Third page to add a new beneficiary

- **Current activity**: select the activities that the beneficiary is doing when he is selected. The list of activities will be different for IFS and IGA types (see page 1). If the beneficiary is doing an activity that is not in the list, you can add a new one by selecting *Other*. But please read section 2.5 before doing this.
- **Previous income**: select the income (in $ per month) that the beneficiary earned before being selected.
- **Assets**: Select the assets that the beneficiary has. If an asset is not in the list, you can add a new one by selecting *Other*. But please read section 2.5 before doing this.

Remark: Like for the first 2 pages, all the data of this page must be completed. If you forget to complete a field, you will not be able to go to page 4.

Once you are finished, click on the button *Go to step 4*, which will bring you to the last page of adding a beneficiary.

3.3.4 Page 4

The fourth (and last) part of the form is depicted on figure 18.
There are here 6 information to complete:

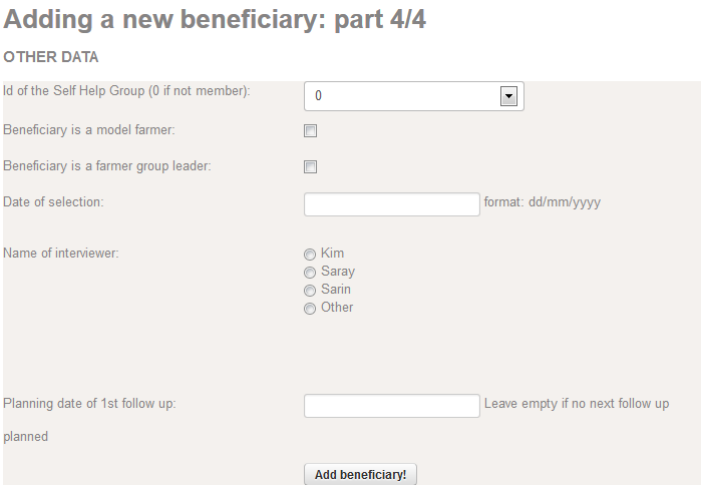


Figure 18: Last page to add a new beneficiary

- **Id of the self help group**: if the beneficiary belongs to a self help group when he is selected, please put the Id of his self help group. To know the id of the self help group, you should *Search* a self help group in the database. Searching data is explained in the second report. If the beneficiary doesn't belong to a self help group when he is selected, just put 0.
- **Model farmer**: check only if the beneficiary is a model farmer when he is selected (only for IFS beneficiaries).
- **Farmer group leader**: check only if the beneficiary is a farmer group leader when he is selected (only for IFS beneficiaries).
- **Date of selection**: select the date of selection. Please just click on the calendar, don't write anything on the keyboard to respect the correct date format.
- **Name of interviewer**: select the interviewer which selected the beneficiary. You can add a new interviewer if he is not in the list, by selecting *Other*. But please read section 2.5 before doing this.

- **Planning date of the 1st follow up**: This tool can help you to know which beneficiary you should follow-up. If you want to use it, select the data on which you plan to do the 1st follow up for the beneficiary (it doesn't need to be very precise, if it is in the same month it is enough). To know what beneficiary you should follow up, do a research on the beneficiaries, and put a filter on the next follow up date. For more information, read the second report, which explains how to search data in the database. If you don't want to use the tool, just let the field empty.

Remark: All the fields are required, except the last one: *planning date of the first follow up*, which can stay empty.

Once you are finished, click on the button *Add beneficiary!*, which will add all the information in the database.

3.3.5 Success

If everything happened correctly, you should get the page depicted on figure 19. If you want to add another beneficiary, you can quickly go to the 1st part of the form, by clicking on *Add a new beneficiary*.

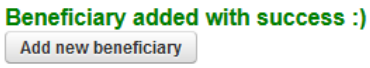


Figure 19: Page that confirms the beneficiary has correctly been added in the database

remark: Please note that all the information is added on the same time in the database. This means that when going for example from page 1 to page 2, the information of page 1 is not yet added in the database. All the information is added together, after clicking on *Add beneficiary!*.
As a consequence: if an electrical cut happens when you are for example on page 3, you will have to restart everything from page 1 to be sure all the data is correctly insert in the database.

3.4 Adding a new follow up

To add a new follow up, first click on *Add a new follow up* button (see figure 14). This will bring you to a form that you need to complete. The form is only 1 page long.
There are 13 informations to add:

- **Beneficiary code number**: This field is very important! When you add a new beneficiary in the database, a unique code number is automatically created by the database. The code has the form: IFSx or IGAx, where x will be replaced by a counter number. For example: if the database contains already 32 IFS beneficiaries, and you add a new one, the code

number given to this new beneficiary will be: IFS33. To know the code number of the beneficiary that you followed up, you have to search in the database. To do this, click on *Search in database* and then *Search beneficiaries*. Add a constraint for example on the name of the beneficiary you want to find, to make the research easier. Please read the second report, which focuses on *How to search information in the database?* if you need more information on finding the codenumber of a given beneficiary.
To make sure you selected the good code number, a blue text will display the name of the beneficiary corresponding to the code number you entered. An example is shown on figure 20. Note that when changing the

Figure 20: The blue text shows that *Chan Reth* is the name of beneficiary number IGA8

code, the page will refresh, whish can take a few seconds. Be patient...
A good idea would be to include a field: *CodeNumber* on the paper form of the follow ups. In that way, you can look for the codenumber before the follow up, and when you complete the database you just need to copy the code from the paper form.

Remark: All the following information is initialized to the value of the most recent follow up, or on the selection data if it is the first follow up for this beneficiary. This is done to make it faster for you to complete the follow up forms, but please be careful: don't forget to update the data that may have change since the previous follow up. If you change nothing, the form will be valid and sent to the database.

- **Current situation**: select if the beneficiary is still active, if he stopped temporarily or if he failed.
- **Current activities**: Select the activities that the beneficiary is doing (note: the list is different for IFS or IGA beneficiaries).
- **Current assets**: Select the assets that the beneficiary has
- **Id of the self help group**: If the beneficiary is member of a self help group, precise the id of the self help group (see section 3.3.4 for more information).
- **Model farmer**: Select only if the beneficiary is a model farmer (for IFS only).
- **Farmer group leader**: Select only if the beneficiary is a farmer group leader (for IFS only).
- **Income**: Select the income (in $ per month) of the beneficiary.

- **Date**: Select the date of the follow up. Please just click on the calendar, don't write with the keyboard to ensure the encoding is correct.
- **Facilitator**: Who did the follow up?
- **Planning date of next follow up**: This is a tool to help you knowing which beneficiary you have to follow. For more information, see section 3.3.4. If you don't want to use it, you can leave it empty.
- **Remarks**: If you want, you can add a remark about the follow up. Let empty if you don't have any remark.

When you are finished, click on: *add follow up!*, to add all the entered data in the database. The page of figure 21 confirms everything happened correctly. If you want to add another followup, you can quickly get on the form by clicking on *Add new follow up*.

Figure 21: Page proving all the data has correctly been added to the database

3.5 Beneficiaries VS follow up: How data is structured

Now that you can add new beneficiaries and new follow ups, it is important that you understand the link between these 2 adding process, and in particular how the data is structures.
The form of the beneficiaries and of the follow ups have a lot of information in common. For example, you have to specify the activities, the assets and the income of the beneficiary in the selection form as well as in the follow up form. The question is: what does the database do with that information?

3.5.1 Only keep the most recent data

The easiest way to proceed, would be to replace the oldest data by the new one. Let's consider an example that focuses on a beneficiary called *Alice*, and that only considers her income. The figure 22 illustrates this example.
During the selection, Alice earns 75$ per month. So when you add Alice (new beneficiary) in the database, you write 75 in the income field. This is represented in green on the figure.
After, during the first follow up, the income of Alice increased to 100$ per month. So when you add the follow up in the database, you write 100 in the income field. The database would then remove 75 (since it is an old data), and replace it by 100. The same would happen when you add a second follow up: 100 would in the example be replaced by 150$ per month.

The big advantage of this method is that it is very easy. When you will do a research on the beneficiaries, you will always see the more recent data (so the

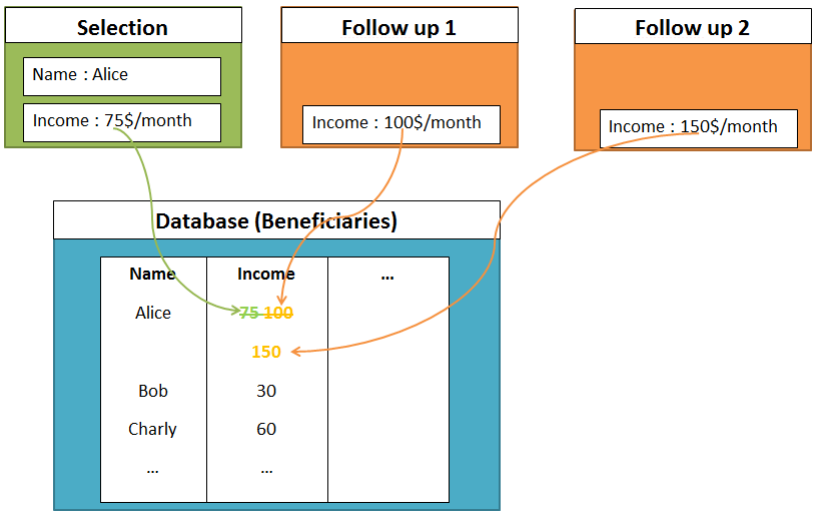


Figure 22: Example of database which only keeps the most recent data.

database is always up to date).
But there are also 2 big problems with this method.
First, you don't know how many follow ups you did for the beneficiary. If we take the previous example and see *150* in the income, we don't know if this corresponds to the initial value (during the selection) or to a follow up value.
Second, because you remove the old information, you can not follow the evolution of the income of a beneficiary. In the previous example, it could be interesting to keep all the data (75, 100 and 150), to observe that Alice's income is increasing. This shows that she is a success case.

The solution is to add a separate database for the follow ups, like explained in the next section.

3.5.2 Keep all the data, by separating beneficiaries and follow up databases

In this section we will see how to keep track of all the data, without removing the oldest ones. We will again consider Alice's example, to make things very precise.
Figure 23 can seems a little complicated and needs some explanations. During the selection (add new beneficiary form), the data is stored in a database called *Beneficiaries* (left blue box). This database contains only the data of the selection.
The results of the follow up forms are added in a separate database, called *Follow ups* (right blue box). With this method, each time you add a new follow up, a new line is added in the follow up database. You never replace the older data. In our example, there are 2 follow ups done for Alice. So 2 lines will be added in the follow up database: 1 per follow up.

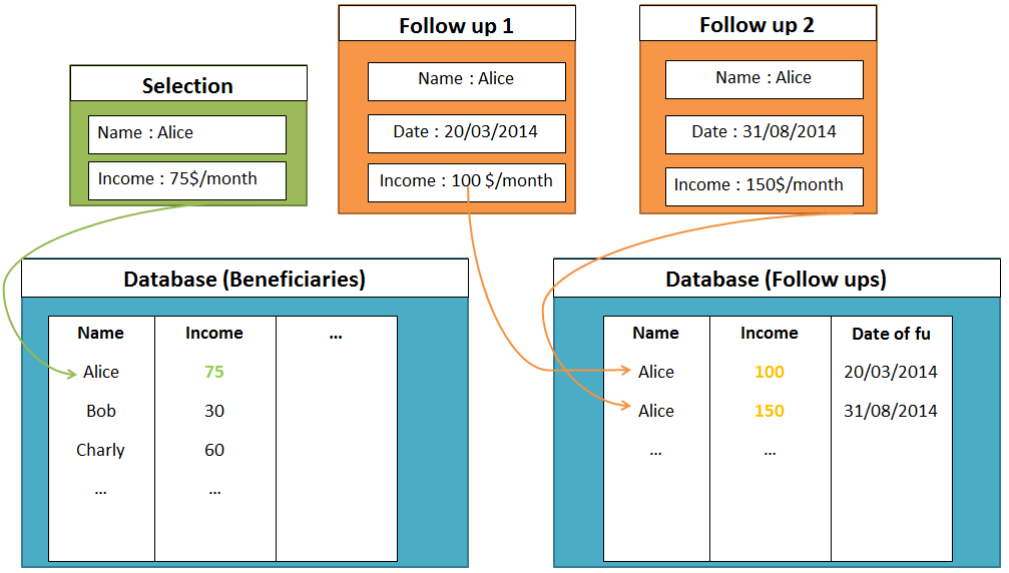


Figure 23: Example of database which keeps all the data (old and new).

The big advantage of this, is that it solves the 2 problems we had before. In clear, we can now follow the evolution of the data of a beneficiary. For example, we can see that Alice's income increased from 75 to 150 after 2 follow ups.

Remark: this method is a little bit more complicated than the previous one, and it is very important to understand it. In particular, when you want to look for the most recent data of a beneficiary, you should first have a look in the follow up database (click on *search in database* then search follow ups). If you look in the beneficiary database (Click on *search in database*, then search beneficiaries), you will find the data that was valid **during the selection only.**
To make sure this is clear, let's consider the following example: you want to know the amount of the richest beneficiary. How would you do?
The natural answer would be: Search in database, search Beneficiaries. Then look for the beneficiary with the biggest income. In many cases, this answer would be wrong. Indeed: we hope that after a few follow ups, every beneficiary will become richer. So, there is a high probability that the richest beneficiary is in the *follow up* database, and not in the *beneficiaries* database (which only contains the data from the selection).
The answer would so be: Search in database, search follow ups. In that table, you should look for the biggest income field, which can be done by sorting the data according to the *income* columns (read the second report for more information about how to search data in the database).

There is 1 exception: the next follow up planning date. In this case, the data is modified directly in the *Beneficiaries* database. The reason is that it is not

interesting to keep track of the old data. Only the most recent date is important for the facilitators. So, the next follow up planning date can only be seen in the *Beneficiaries* database. This is shown on figure **??**.

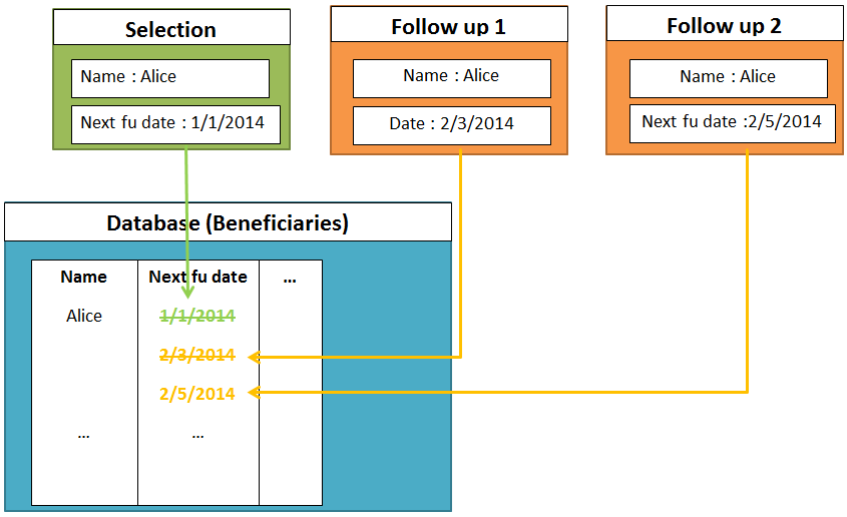


Figure 24: The *Next follow up date* is updated directly in *Beneficiaries* table.

3.6 Adding a new training

To add a new training, first click on *Add a new training* button (see figure 14). This will bring you to a form that you need to complete. The form is divided in 2 pages.

3.6.1 Page 1

There are 6 informations to add on the first page (shown on figure 24).
Here are the explenations of the different fields:

- **Type of training**: Select IFS or IGA. The list of train topics will be updated depending on this selection.
- **Topics of training**: Select the topics that were explained during the training.
- **Date of the training**: Select the date of the training. Please just click on the calendar dates, don't use the keyboard. This makes sure the format of the date is correct. If the training last more than 1 day, enter the first day of the training.

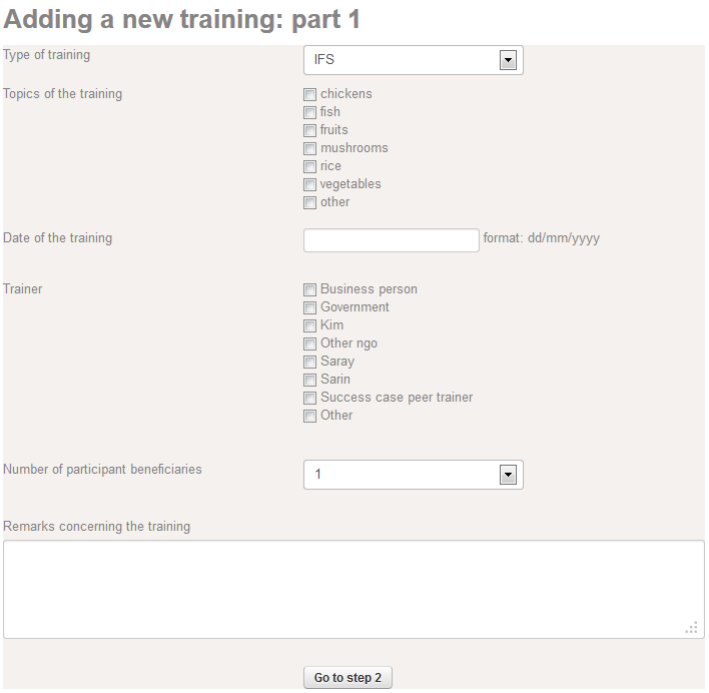


Figure 25: Page proving all the data has correctly been added to the database

- **Trainers**: Select the trainers. You can select multiple trainers. If the trainer is not in the list, you can add a new trainer by selecting *Other*. But please read section 2.5 before doing this.
- **Number of participant beneficiaries**: Select the number of beneficiaries that took part in the training. This is important for the second page of the form (see section 3.6.2).
- **Remarks**: If you want, you can add any remark about the training. If not, let the field empty.

Finally, click on *Go to step 2*.

3.6.2 Page 2

The objective of the second page is to know which beneficiary took part in the training and what kits they received. An example is shown on figure 25.
For each beneficiary, you need to fill 2 data:

- **Code number of the beneficiary**: This is exactly the same mechanism as the 1st field of a follow up. Please read section 3.4 for more information.
- **Kits provided**: If kits are given to the beneficiary, please check the corresponding kits.

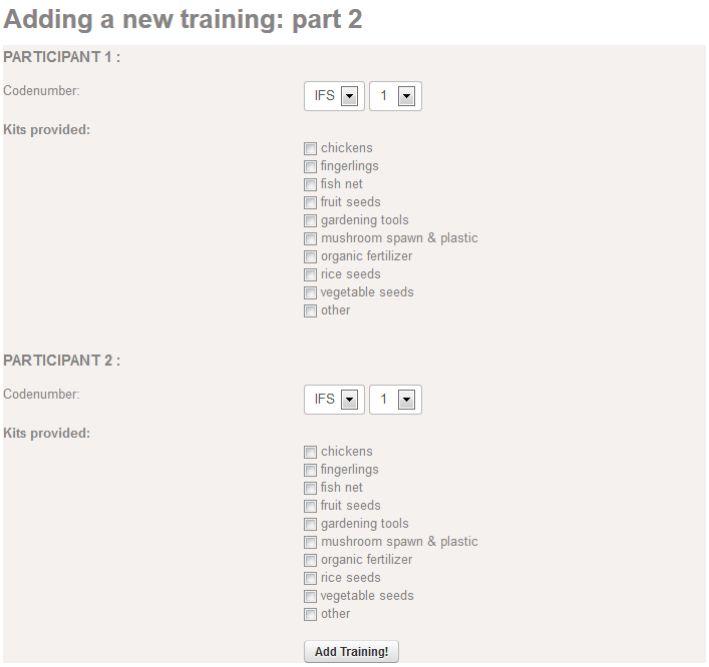


Figure 26: This page is an example with 2 participant beneficairies

Remark: because you need to input all the codes of the participants, you should first check search all the codes in the database before starting to complete the form of adding the training. Like for follow ups, a good idea should be to write all the beneficiary codes of the participants on a paper form, so that you only need to copy the codes when you add a new training in the database.

When you are done, click on the button *Add training!* to add the training in the database. Like usual, a confirmation page will appear, and you will be able to add another training using the button *add a new training*.

3.7 Adding a new self help group

In the database, a self help group (called SHG in what follows) is a group of beneficiaries. Beneficiaries who belong to the same SHG, share a same *SHG id* number. There are 2 different possibilities concerning adding Self Help Groups : add a new self help group and add a new member to an existing self help group. This 2 things are explained separately in what follows.

3.7.1 Adding a new self help group

If you want to add a new SHG (which means it is the first time it will contain beneficiaries), first click on *Add a new self help group* button (see figure 14). This will (as usual) bring you to a form that you need to complete. The form

is only in 1 page.
The fields you need to complete are the following:

- **Date of creation**: Simply select the date when the SHG was created. You don't need to use the keyboard, just click on the date on the calender.
- **Number of members**: Write the numbers of members in the group. The form will then display selection fields for the codenumbers of the members.
- **Code number of the members**: Write the code number of every member in the self help group. If you are not clear with code numbers, please refer to section 3.4. When a beneficiary is selected in a new self help group, his *self help group id* will automatically be modified. This id will be modified only in 1 database:

  – In the *follow up* database if the beneficiary has had at least 1 follow up. If he has had multiple follow ups, only the last instance will be modified.
  – In the *beneficiaries* database if the beneficiary has never had a follow up.

  Let's consider an example on figure 26 to make it clear. In this example,

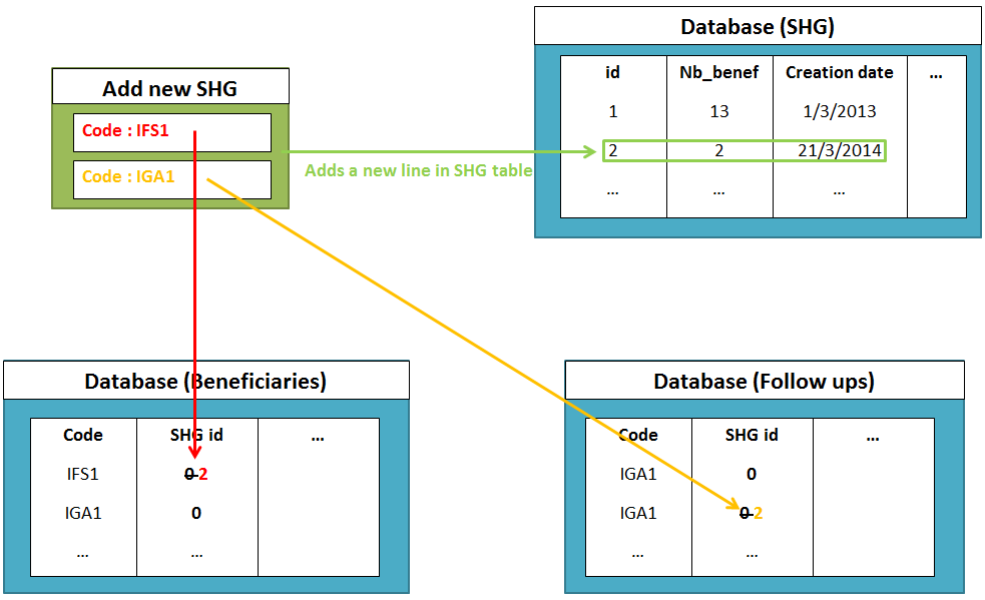


Figure 27: What happens when adding a new SHG?

we add a new SHG containing 2 beneficiaries: IFS1 and IGA1 (see green box). The first thing that happens, is that a new instance is created in the table *SHG*. As you can see, it has the id number 2 (automatically

generated), nb_benef = 2, and the creation date is filled in.
The next thing to do, is to modify the beneficiaries IFS1 and IGA1, so that we can know that they belong to the self help group with id 2.
Like you can see, IFS1 has never had a follow up (there is no line in the follow up table containing code IFS1). So the most recent information about IFS1 is the information of the selection. This is in the *Beneficiaries* table.
For IGA1, the situation is different. Like you can see, this beneficiary had 2 follow ups (the 2 lines of the table *follow ups*). In this case, only the most recent follow up is modified, and the *Beneficiaries* table is not modified.

If you are not clear with the difference of the *beneficiaries* and *follow up* databases, please refer to section 3.5.

- **Remarks**: If you want, you can write any remark to give more information about the self help group. An example of correctly filled form is given in figure 27.

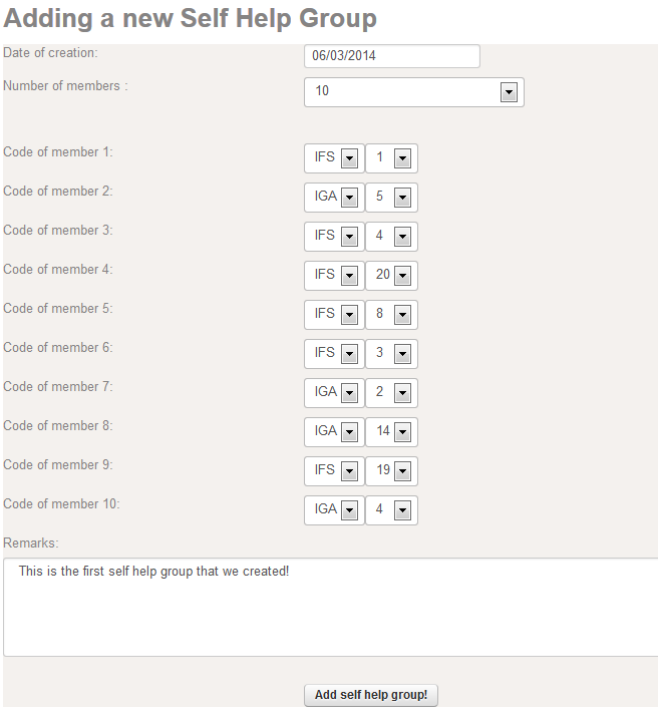


Figure 28: Example of correctly filled up of SHG form.

When you are done, click on the button *Add self help group!*. You will arrive on a page confirming the self help group was correctly added. You can then easily add a new self help group by clicking on the button: *Add new self help group*, which will bring you again on the form. The confirming window is

depicted on figure 28.

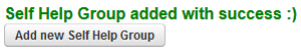


Figure 29: Page that confirms you filled up the shg form correctly.

3.7.2 Adding a new member in an existing Self Help Group

Another situation, is when a self help group has already been created, but you want to add new members in this self help group. This can be done in 2 ways:

- If the beneficiary has not yet been added in the database: add the new beneficiary, and on page 4: select the id of the self help group the beneficiary belongs to. If you don't know the id, you should do a search on all the beneficiaries that belong to a SHG. You can than observe the different ids of the SHG the beneficiaries belong to (for more detail, read the second report, that focuses on how to search information in the database). When the id of the SHG is put, the field *nb_members* of the self help group will automatically be increased by 1.
- If the beneficiary who joins the SHG is already in the database, just add a new follow up. Indeed, in the follow up, you can input the id of the SHG that the beneficiary belongs to (see previous point if you don't know the id of the SHG). Again, the field *nb_members* of the SHG will automatically be increased by 1 when the beneficiary is added.

3.7.3 What if a beneficiary leaves a SHG?

It is also easy to remove a beneficiary from a SHG. To do this, just add a new follow up, select the corresponding beneficiary (using his unique codenumber), and put 0 as SHG id. The *nb_members* field of the SHG will than automatically be decreased by 1.

4 Conclusion

I hope that after reading this document, you will be able to add data in the database in an efficient way. If you have any problems with the database, or any question, please ask your project manager, LD, or myself.
I spent 6 month making the database. I prefer spending a few hours more to solve problems, than to hear that MODE stops using the database.
I join you my email address: don't hesitate to send me any question about the database, I will really look forward to helping you managing MODE's data :)

moreelsc@gmail.com