Friends in Village Development Bangladesh (FIVDB)

Efficient Survey Data Entry

A template for development NGOs

[This page deliberately left blank]

Efficient Survey Data Entry

A template for development NGOs

Abu Saeem Arif
Aldo Benini
Hasan Ahmed Chowdhury
Wasima Samad Chowdhury
Saiful Hasan
Md. Yasin Mazumder

Friends in Village Development Bangladesh

FIVDB

November 2010 / March 2011

Photos by FIVDB staff

Suggested citation:

Arif, A.S., A. Benini, et al. (2010). *"Efficient Survey Data Entry. A template for development NGOs"*. Sylhet and Washington DC: FIVDB [Version March 2011].

Table of Contents

List of Figures

Preface

Friends in Village Development Bangladesh (FIVDB) is a mid-sized development NGO, with a thirty-year tradition of multisectoral rural development support in the northeastern region of the country and, since recently, with an additional focus on an ambitious popular education and community organization program that is expanding into other regions as well. The information requirements in coordinating and reporting on this work are considerable.

Surveys contribute, in important if not always prominent ways, to the information flow. They supply information that guides a variety of deliberations and decisions, from the selection of program sites, through the recruitment of participants, the enhancements of program reports beyond financial and administrative data, and ultimately to the measurement of outcomes and impact.

While the FIVDB management is committed to quality information, my colleagues and I at the directors' level keep its arcane technical aspects, such as survey data entry, at arm's length. We have a monitoring unit; we trust it to take good care of survey processes. Instead I wish to emphasize two larger considerations:

First, although rarely discussed in these terms, there is a market for data in FIVDB and similarly, I believe, in other development NGOs. Data is continuously being produced and exchanged. Survey data is produced, not by one department only, but in collaborative efforts. Our monitoring associates may be the ones committing the bulk of the data to computers, but in the first place the interviewing in the villages is done by frontline workers of other departments. These units give time and effort because they hope that the surveys will benefit their work too, in the short run by accessing potential clients, in the longer term by demonstrating its impact. The market for data is not only internal. Volunteers in Community Learning Centers create village maps, with numbered households that are the basis for sampling later during outcome measurement. They do this because they expect FIVDB to return information of practical value.

Second, FIVDB has been using computers for the management of survey data since 1987. Our command of this technology has grown but slowly and with occasional setbacks. We are still struggling with hardware problems in field offices, but even more so with the turnover of trained personnel and with the conceptual challenges of building a consistent survey tradition. At the same time, the qualitative demands for data have grown, in a competitive aid climate that wants programs to demonstrate impact. While our donors are free to commission external evaluations, FIVDB itself aspires to monitor its work using data that I like to call, for lack of a better term, "near research-grade". Valid definitions and reliable processing become imperative if we are to be credible.

It is in those two contexts - collaboration and quality - that data entry technology matters. The better it helps to reduce errors, the freer will our statistics be of noise and bias. The faster it does the job, the sooner will FIVDB, its donors and its partners in villages and government have access to survey data, survey findings and their implications. The easier it is on the monitoring associates, the more time and energy will they save to do other important work, away from their desks and in the true field.

The new data entry template that one of our large baseline surveys is experimenting with is a step in that direction. This paper details its technicalities and makes its core elements available for others. Like other technologies, it will be effective if it is "always at our side, and never in our way". Data entry is an important subject for survey practitioners - so that the rest of us need not bother about it.

Bazle M. Razee
Associate Director, Program & Planning

Acronyms and abbreviations

CAPI	Computer-assisted personal interviewing
CLC	Community Learning Center
FIVDB	Friends in Village Development Bangladesh
ICT	Information Communication Technology Center
NGO	Non-governmental organization
PPR	Planning, Policy and Research
SPSS	Statistical Package for the Social Sciences
STATA	Stata - A statistical software package
UNESCO	United Nations Educational Scientific and Cultural Organization
VBA	Visual Basic for Applications, the programming language of the macros controlling the data entry template

Acknowledgements

This study would not have been possible without the work of these colleagues in the FIVDB Policy, Planning and Research Unit:

Arif Azad Khan, Arif Mohammad Shakil, Hasena Begum, Md. Abdus Salam, Md. Yeasin Mazumder, Muhammad Al Amin, Muhammad Saddam Hossain, Rakshit Bhattacharjee, Rashida Jahan Qureshi, Samaresh Talukder, Syeda Shahina Akther Ruba.

Dipok Roy, of the FIVDB head office staff, took several of the photos used in this document.

Summary

In poor countries, surveys of households and communities are conducted by research institutes and statistical bureaus as well as, significantly, by NGOs. Some of their surveys are large. Baseline surveys sometimes involve full enumerations, to serve multiple purposes of rapport building, beneficiary targeting and listings for later sample surveys. In large surveys, data collection and data entry take much time, compressing design, pre-tests, analysis and dissemination. By and large, this has been the experience also of surveys that the FIVDB Policy, Planning and Research Unit has coordinated.

Data entry in particular is the least loved of all survey phases, feared as a source of cost and error, a necessity without learning value. Survey leaders, worried about cost, time and reliability, are tempted to invest the bare minimum in data entry operators. Greater productivity is expected chiefly from technical improvements, often designed as closed applications that the NGO subsequently cannot adapt to changing needs.

This approach is short-sighted. Appropriate data entry tools, together with skills training and encouragement to rapidly analyze data in field offices, can do more. They are empowering, for direct survey workers as well as for contributing program staff and grassroots organizations.

In the wake of two rounds of a large baseline survey in northeastern Bangladesh, we present a novel data entry template. It combines ergonomic benefits and data validation features with the flexibility to adjust category sets and response codes locally. It is easily adaptable; with appropriate training, field-based monitoring staff can produce tabulations from local data sets. They need not wait for the central monitoring unit to create master tables from the entire survey dataset.

This paper presents the rationale and key elements of the template. It comes as a workbook in Microsoft Excel, an application widely used among NGO workers. We describe most elements in a non-technical language. For technically inclined readers, we

detail how a special Excel feature - event macros - is harnessed to check entries for valid values and to scroll to the next entry cell.

Figure 1: A segment of a data entry screen

	3	4	30	31	32
1	SerialNo	Hhno	LoanCurrentAny	LoanSource1	LoanBalance1
9	8	9	No		
10	9	10	Yes	Krishi Bank	20000
11	10	11	No		
12	11	12	Yes	Moneylender	30000
13	12	13	Yes	Krishi Bank	20000
14	13	14	Yes	Krishi Bank	40000
15	14	15	No		
16	15	16	No		
17	16	17	Yes	Entry error	15000
18	17	18	Yes	ASA	Use number!
19	18	19	No		
20	19	20	No		
21	20	21	Yes	Grameen Bank	20000

Note: Data shown here is simulated. Serial numbers are automatically inserted. A macro checks whether household numbers (Hhno) are unique. Errors due to illegal values - categories not included in defined sets; text entries in a numeric field - are flagged.

We document the code and supply a demonstration file. Specifically, the template may appeal to NGOs which

- do surveys in-house, but do not have professional research units
- have monitoring units, with head office and field workers trained (or trainable) in basic data management and analysis
- record interview data on paper questionnaires (as opposed to computer- or mobile phone-assisted capture)
- can benefit from decentralized data entry and rapid data use in field units.

The experience that FIVDB has so far made with the tool suggests that data entry is considerably faster and less prone to produce spelling or data type errors. Also, due to

the protected variable sequence, the process of assembling master tables from numerous contributing tables (for different villages, and from several field offices) is safer.

As a side benefit, this template has enabled FIVDB to outsource a large part of survey data entry to a computer center run by a village grassroots association. Efficient prompts for categorical variables and automated error flagging ensure good quality with minimal outside supervision, setting monitoring staff free to attend to other work.

Adapting the template to the needs of other surveys involves the definition of data tables, variable codes and categories, and macro segments as exemplified in the text as well as in the demonstration Excel workbook. The workbook can be downloaded via this link.

This tool can be further improved. Also, we are convinced that "out there" in the wider NGO world many other solutions have been applied to data entry challenges. Over the next years, survey data may increasingly be captured in programmable mobile phones. Many of the required conceptual skills, however, will remain the same.

Improvements in survey data entry are only a small part of an evolving toolbox. FIVDB recognizes that better data management and analysis skills are needed in monitoring staff as well as in their close program staff counterparts. Increasingly, NGOs are under pressure to demonstrate impact. The capacity for self-observation is critical in competitive funding markets. More efficient surveys add to it.

Introduction

FIVDB, like many other development NGOs, has been conducting numerous surveys over the years. The nature of surveys has varied. Some were undertaken to satisfy project reporting; others looked at impact out of a concern that FIVDB and its donors shared; yet others were done for our own learning. A few followed templates prescribed by specific donors, sometimes without any role left for FIVDB to take part in analysis and discussion. In most surveys, however, FIVDB designed and executed all the phases, from concepts to dissemination of findings.

That has included data entry, which we have, with rare exceptions, done in-house. In 2008, FIVDB added a new project, Jonoshilon (FIVDB 2008), with educational, social organization and livelihoods interventions planned in 850 village communities. A dozen monitoring staff were hired to better observe the expanding project. They have been engaged in large baseline surveys. Despite the staff reinforcements, time and resource constraints made themselves felt particularly in the data collection and entry phases.

In order to ensure data reliability while at the same time easing the strain on field monitors, we have rationalized data entry over successive survey rounds. At present, an innovative data entry template is being used. This study details its architecture and documents macro code that other survey designers may wish to adapt to their own needs. We trust that there are other solutions and insights "out there" in the large world of NGO-driven surveys and are eager to learn from them for FIVDB's own productivity.

Data entry under time pressure

In survey research, "data entry" denotes the process of transferring information from the medium that records the response (traditionally answers written on printed questionnaires) to a computer application. "Data capture" is an almost synonymous term.

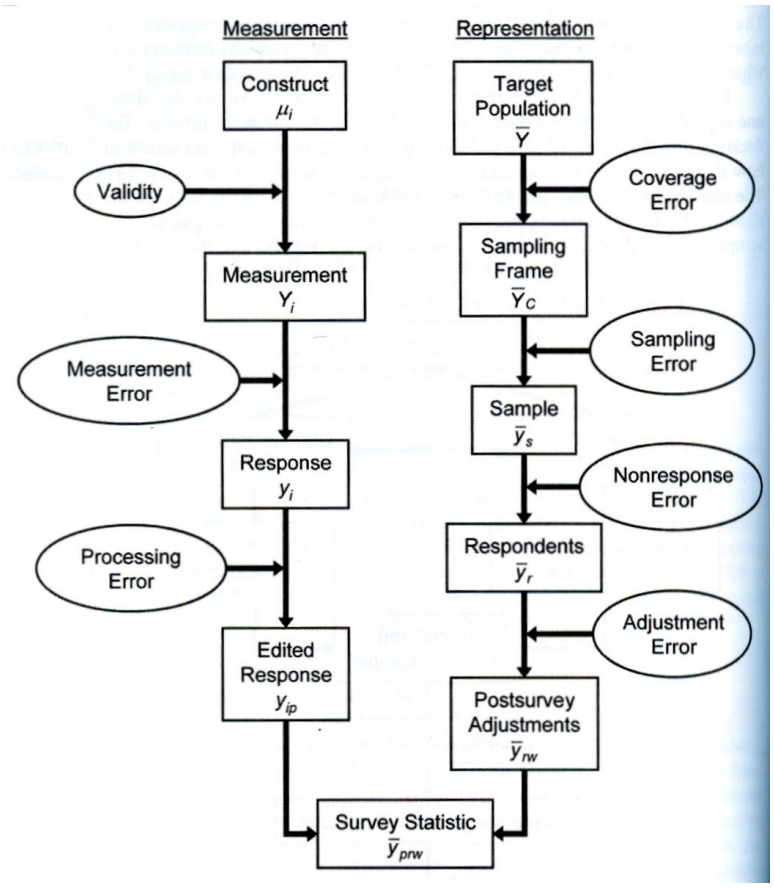
Data entry does not command much prestige. It is accepted as a necessity and feared as a source of cost and of error. Except in rare circumstances - qualitative research projects that combine coding with data entry come to mind -, the learning value of this survey phase is low. Often the work is contracted out or assigned to minimally skilled workers who otherwise have no stake in the research[1].

Under time pressure, or for lack of proper supervision, entry errors can be substantial. In a survey quality perspective, these errors are a subtype of processing errors. They appear on the left side of the survey lifecycle diagram in the classic book on survey methodology by Groves et al. (2004: 48). It may be significant that the authors do not even consider data entry a subject worthy of discussion. In practice, data entry challenges may bedevil also the representational qualities of surveys (the right branch of

[1] These and related complaints have been detailed for large-scale epidemiological studies, which also rely, in considerable part, on questionnaire surveys. For example, Ali et al. (2006: 1) note that an "efficient data management system is often not available", and "surprisingly little attention is paid to the computerization of data".

the diagram); under time pressure, only part of the questionnaires may be processed, causing the effective sample to fall short of targeted precision.

Figure 2: Survey lifecycle and error sources (Groves et al.)



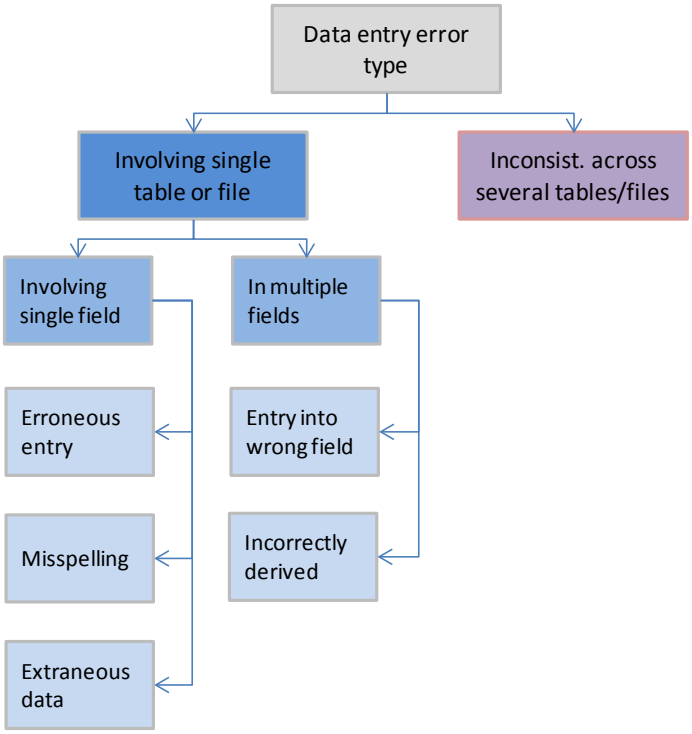
Barchard and Pace write: *"Data entry errors can be disastrous. In many industries, data entry errors are minimised by paying highly skilled data entry personnel or by using expensive optical scanning technologies. However, researchers, small businesses, and non-profit organisations cannot afford highly skilled data entry personnel or the alternative technologies, and instead rely upon minimally trained staff and volunteers for manual data entry. Obtaining accurate data entry is a challenge in these circumstances."* (2008: 359).

"Disaster" may sound overly alarmist; but their description of the issue outside well-endowed research agencies is pertinent; it leads us closer to the world of NGO-led surveys, with its reliance on manual data entry.

[Sidebar:] Types of entry errors

While data entry errors form a subgroup within processing errors (which some authors subsume under measurement errors while others treat them separately), it is useful also to clarify differences within entry errors. In their "A Taxonomy of Dirty Data", Kim et al. (2003: 84) list several types, which we have arranged in this diagram.

Figure 3: A typology of entry errors



They give examples of instances in types that are not self-explanatory: An erroneous entry happens if, e.g., age is mistyped as 26 instead of 25. Extraneous entries add correct, but unwanted information, e.g. name and title in a name-only field. An entry of an incorrectly derived value occurs when a function was incorrectly calculated for a derived field (we might think of a data entry person who incorrectly transforms British into continental measures). Inconsistencies across tables or files occur e.g. when the number of employees in the employee table and the number of employees in the department table do not match.

These authors do not include so-called dangling data under entry errors, i.e. records in one table (e.g. household members) for which at least one corresponding record in some other table is required, but is missing (e.g. household). To the extent that the corresponding records were all present in the source document, one is inclined to treat the loss as a data entry error.

We believe that another, rarely discussed important distinction concerns whether and how errors can be recognized and corrected:

1. by pre-programmed validation rules (e.g. Sex = male and Pregnant = yes → Error!)
2. by comparing with the source document or by double entry
3. by outlier and plausibility tests, with possibility to re-measure in the field
4. as in 3., but without possibility to re-measure.
5. without any basis to determine implausible values

In cases 3 - 5, computer entry and source document may agree. If so, these errors are not entry errors.

Obviously, there can be other error typologies, based on other distinctions; there will also be types of errors detected in data management practice that overlap several types.

Survey quality in development NGOs

Rural development NGOs conduct surveys in large number and variety. Often design and supervision are in the hands of small monitoring units, with limited research skills. In the field, data may be collected by program staff outside of the monitoring line. Interviewing, editing and transmission compete with their other duties. Data entry may take place in the central office far from the field; even where field office workers enter the data, they may have scant contact with the actual interviewers.

Professional research institutes continuously control survey quality through a number of devices, such as teaming up their field supervisors with field editors (Sana and Weinreb 2008). By contrast, NGO-led surveys tend to face tighter resource constraints, in the quality control chain as well as in terms of computer resources. Poorer NGOs often lack the expertise for sample surveys. Full enumerations create large data volumes; it is not unheard of that surveys never made it beyond data collection. Thus, while unit costs may be lower in NGOs, the tendency to over-collect is liable to inflate the cost, particularly the opportunity cost, of surveys and to depress data quality.

These differences are not immutable. Technology has changed the survey environment. Mobile phones and e-mail have made survey management easier, in terms of faster coordination and document transmission. Technological advances have facilitated data entry as well.

There are a number of computer applications meant to reduce the data entry burden as well as entry errors. Among relief and public health personnel, for example, Epi Info is popular (CDC 2005). At the high end of data management, the US Census Bureau has made available a programmable application - CSPro (U.S. Census Bureau 2009). In development NGOs, it is not uncommon to call on external consultants to create dedicated applications using Microsoft Access, for data entry as well as for standard reports. The statistical application SPSS enjoys widespread popularity, particularly when survey operations are shared between NGO staff and consulting firms.

Some of these applications work together with modern technologies that capture data electronically at the point of response. These - known by acronyms such as CAPI (computer-assisted personal interviewing) [2] - have made headway in relief and professional survey organizations. As mobile telephones become cheaper and more

[2] This is a fast-moving field (See e.g., Cork, Cohen et al. 2003: probably obsolete by now), with interesting connections also to mobile telephony (Tomlinson, Solomon et al. 2009) and GPS devices.

widespread (phenomenally so in Bangladesh), we will over the next years see more data entry tools developed for these devices - for interviewers capturing the response in programmable phones and transmitting it to a remote database, as well as for sample of respondents who will be given such phones to communicate repeated measurements (Couper 2005).

The adoption of these technologies by national NGOs in poorer countries has been slow. FIVDB's surveys are using, and the rest of this paper assumes data entry off, traditional paper-and-pencil questionnaires.

For the build-up and maintenance of survey competency in development NGOs, data entry relying on EpiInfo, SPSS and Access-programmed special applications suffers from two serious drawbacks. First, these applications are centralizing in the sense that not only the design, but also the operation of the specific data entry modules remains in the hand of few persons - those who can realistically be trained and supervised for the purpose. They generate little or no skill transfer to other persons or for other tasks. They are often not sustainable because the expertise for needed design changes is lost, too slow or prohibitively expensive. Typical manifestations are abandoned database applications [3] and socially isolated analysts in backoffices who produce one-time statistical analyses for reports that will never be used in the field.

Second, they presume a degree of conceptual fore-knowledge of the social field to be surveyed that may result in bias or inefficiency at the analysis stage. The application causes inflexibility or loss of valuable information that does not conform to the script. This is a particular problem with categorical variables. Survey designers assume that they know the relevant response categories and fix category sets in advance. Substantive response outside the pre-defined options is handled through the option "Other". The specifics are recorded in a text or memo field. This information may remain unreported, or unused or accessible only after time-consuming manipulation[4].

---

[3] In the extreme, they are created by consultants who subsequently emigrate together with the password, which they never shared with their principals, as we found in a NGO in Nepal.

[4] Survey researchers have long recognized this danger. One of the remedies recommended consists of "post-coding" unanticipated categorical and other qualitative information, the operation of assigning codes to recorded (textual, verbal, pictorial, etc.) information *after* the interview and, ideally by persons other than the interviewers, before or during computer entry. Some code-category pairs may be created only then.
However, the skill requirements may be considerable (de Silva and Gunetilleke 2008). The logical challenges of coding have prompted in-depth studies in other fields as well, notably in health care informatics, particularly in processing information on (partly patient-described) disease symptoms, which may defy clear diagnoses. The two approaches to coding in this field are called "pre-coordination" and "post-coordination", but the rationale is the same as with pre-coding/post-coding: *"Pre-coordination and post-coordination can complement each other, with pre-coordination providing logic and intricacy and post-coordination allowing expressivity and more complete domain coverage"* (Rosenbloom, Miller et al. 2006: 280).
This is a thorny issue and one sufficiently important in our view to devote a sidebar to the topic of recoding, which we consider a desirable skill needed in the transition zone between data entry and initial analysis. See further below, page 41. The need arises regardless of what entry template one uses.

Data entry and empowerment

We offer elements of a more flexible data entry arrangement, with built-in controls that help to reduce errors and at the same time speed up the entry work. We use an application - Microsoft Excel - that is widely known by development NGO personnel. We exploit features of Excel that allow field personnel to add or modify category sets while making the coding easier. Our template, built with these elements, encourages several desirable practices:

- decentralized data entry with built-in error checks
- modifiable category sets to capture local specifics
- local copies of data tables for immediate analysis in field offices, and
- assembly, at the central office, of the various local data sets into one master file.

These characteristics empower field personnel, encourage individuals of different skill levels to work together and accelerate feedback to grassroots organizations of the communities in which the data was generated.

[Sidebar:] Rapid utilization in the field

The analysis and reporting of large baseline surveys can be a time-consuming affair. These phases are often centralized. By the time the head office releases a report, life and program priorities in the field have moved on. If at all final data sets are copied to field offices, guidance to produce lists and statistics tailored to local needs may be insufficient or outdated. Monitors and program supervisors may take cursory note of major baseline findings, but they may not have the time or incentive to translate them in staff meetings, and even less so for the individuals and grassroots organizations that collected the data in the first place. The participatory element of the NGO culture may be strong, but in itself does not provide the tools to retranslate program-wide results into bits and pieces of specific local interest.

The opportunities for creative feedback often go unnoticed. NGOs do go out of their way to build analysis skills in the groups they organize - in literacy, accounting, resource mapping and in a proliferating variety of participatory research and reflection tools (Chambers 2008). In most events, the information used is on one village community or small grassroots organization. Comparative perspectives on groups that do not regularly meet are not prominent.

It takes relatively little investment in Excel skills training for area-based monitors to produce simple tables that speak to a number of neighboring communities or program groups. Our template encourages the practice of making local copies of data tables and to analyze them long before the overall survey report is available. Appending tables from several villages and extracting comparative statistics via Pivot table require skills that are easily teachable.

Monitoring associates have produced such tables, in the Bangla script and on a large sheet that neo-literates could read, for a number of Community Learning Center meetings. For example, they offered three posters with tabulations for discussion when they met with some 35 people of Pirergaon village in Sylhet District in August 2010. Two of the posters spoke to the situation of the village. The first table compared literacy levels by gender. The second summarized the adoption of hygienic latrines by four wealth ranks. The ranks, from rich to ultra-poor, had been assigned to households by the Center volunteers themselves, at the time of creating a village map. Literacy and sanitation data were elicited during an FIVDB survey, which then also noted wealth ranks.

The third table contrasted Pirergaon with its neighboring village of Foringura, as regards the school enrollment of children age 5 - 12. This data had also been collected in the survey and broken down by the wealth ranks of the children's parental households. In total, 134 of Pirergaon's 196 children in that age bracket attended school, and 249 out of the 309 in Foringura.

Figure 4 a and b: A village meeting energized by poster-size data extracts

This manner of presenting information in tables and in comparison with some other village was new for many, and time was spent to explain and clarify. The discussion was lively. Questions ranged from curiosity ("Who is the child that you say is the one child from rich families in our village that is not going to school?") to appeals for help to continue one's children's education, to the expression of local pride ("Our situation is better than Foringura's in all respects, except in education" [Never mind that on literacy and sanitation the meeting did not have comparative information!]).

Requests for action were also advanced. FIVDB was requested to provide a print-out of those fellow villagers who were not fully literate (for literacy training). But most speakers singled out sanitation problems - the need to install hygienic latrines in all households and the inability of the 14 ultra-poor households to pay for them from their own means. Again, a request for a printed list was made, of those 14 households, for whose benefit the committee wanted to collect money. The meeting ended with a request that the monitoring associates come back with other comparative statistics that would let the villagers understand their situation in a new light.

Our impression is that the skills level of monitors rarely is the most limiting factor on the use of survey data in field offices and among program participants. Role ambivalence - other field staff treat monitors as data entry clerks and reporting assistants -, excessive interviewing and data entry burdens as well as difficult coordination with other program lines are more deterring. These conditions can be improved through supportive policies from the center. Chances will then be better that local participatory forms - in the recruitment of program participants or in grassroots planning events - benefit from computerized survey data. To the extent that data entry takes less time, and information of local value can be produced faster, survey practices and participatory culture can support each other in a virtuous cycle.

Ergonomic and data quality motivations

The motivation for this instrument has sprung from practical experience of a kind that other NGO-led surveys are likely to make in similar forms. The motives that drove the process are universal: workload reduction, convenience and productivity in the survey workforce; and concern for speed and quality of the product.

Between summer 2009 and spring 2010, FIVDB conducted a baseline survey of all households in 98 villages of four districts in northeastern Bangladesh. Data was collected and processed on over 19,000 households and over 110,000 residents. The data was entered by eleven field office-based monitoring associates. Most of them had undergone data management training in Excel in two workshops in 2009 totaling eleven training days. It had been collected by program staff trained by the monitors and supported by grassroots committees. The committees - so-called Community Learning Centers - shared maps of dwellings that they had drawn during participatory reflection

events in their village communities. While the village maps were important for collective learning, the baseline survey was able to access them for elements of the unique household and person identifiers that will be needed in later follow-up surveys.

In retrospect, the baseline data entry effort is difficult to quantify, but it is safe to say that it took several months of each monitor's working capacity.

The monitors entering the data had undergone more intensive data management training than what workers in similar positions in other Bangladeshi NGOs are typically afforded. Yet, there were two problems that impaired productivity and reliability:

- The template made extensive use of Excel's data validation feature. While reducing errors, it also obliged the monitors to enter categorical data by multiple mouse click-and-scroll operations for every active data cell. These were time-consuming and tiring.

- The data sheets were not protected. For workflow convenience or out of local analytical interest, several monitors took to rearranging the order of columns in data sheets. This complicated the table appending operation while creating master tables for the 98-village database. Some of the more subtle reordering was not detected until the analysis stage; amending the tables and re-doing analyses lost considerable time.

It was primarily the drive to mitigate those problems that led to a modified data entry template, including features that the following sections present. The monitors requested a template that would require less frequent switching between key strokes and mouse operation. The organization wanted to minimize errors in data entry and in appending tables for the master file.

In addition, the ability to locally add categories to the set of legal values seemed desirable. Such a feature would avoid the dilemma between provoking error messages or relegating novel categories to supplementary text fields. For example, when asked about membership in organizations, interviewees reported some that are strictly local and were unheard of in the FIVDB head office. Given their local importance, they should appear in tabulations and, for this to happen, need to be in the set of organizations, with codes prompting for them and with (at least locally) uniform spellings.

The template was to find the right balance between central control and local autonomy in other ways too. It was felt important to create a system of unique record identifiers. These needed to respect centrally prescribed rules; they needed to be created in the field offices before the master file of all survey records would be ready while incorporating elements of village maps kept in the Community Learning Centers.

Organization of this study

We proceed as follows. We spell out the assumptions, particularly regarding staff skills. We describe precautions taken against potential errors and for better productivity. We describe the basic mechanics of the data entry workbooks. Since surveys may collect data on more than one entity, we open a sideline to illustrate a two-level data structure. For our household and household member data, we describe a system of unique identifiers that facilitates linkage between tables and with follow-up surveys. We then proceed to the more technical part.

This part is again sub-divided. We describe devices that our template uses without the need for Visual Basic for Applications (VBA) code. Separately, we present sample VBA code used in macros that monitor and speed up the data entry.

We then report on benefits as well as problems encountered during the use of the template for the current baseline data entry round. We hope that readers may contact us with improvements. We assume that, "out there" in the networked audience, there are individuals and groups savvier than we are in survey support, and that we can learn from them to improve this template or to altogether replace it with something better.

Mechanics of data entry

Assumptions and precautions

We assume a set-up for an NGO-led survey inspired by the current FIVDB processes, but plausibly similar in other agencies as well:

- A monitoring unit, attached to the NGO central office, designs most or all of the survey tools. It trains the field staff in its direct line in their use. These workers learn how to train and supervise interviewers (recruited among program staff and grassroots volunteers) and how to manage rapport with village communities and with others whose cooperation is beneficial (such as local government officials).

- In the head office, some monitoring staff have an advanced command of MS Excel. They have at least limited VBA skills and can adjust macro code by imitating exemplary code, or else have access to someone who can do it for them.

- Head and field offices are equipped with computers in which a macro-enabled version of Excel has been installed. The head office shares data entry templates with the field offices; the field-based monitors have been trained in their use (and ideally were part of the pre-test and revisions). File naming, sharing and back-up protocols are in place.

- Field-based monitors possess a medium level of Excel skills. They know how to link, append and merge tables, understand identifier concepts, and can make

two-way tables off data tables. They are motivated and able to do simple analyses of the local data.

- Program and monitoring staff share a concern for reliable data. They expect to feed back some of it to the grassroots organizations who helped in the collection, for local targeting and planning purposes.

Experience with an earlier template that FIVDB used in 2009 recommended a number of precautions. These were built into the next data collection and entry round in 2009 - 10. Some of them may be of general value:

- Field offices receive data entry templates. The monitors, who install them in machines with MS Excel with macros enabled, have previously been trained in their use as well as in basic descriptive data analysis.

- The templates have data entry sheets with protected areas. These are passworded at the central office. Similarly, the macro code is protected.

- Additional variables can be added locally in unprotected areas. The definitions of standard variables and the category sets are shown in the same workbook, in sheets separate from the data sheets.

- Data entry personnel can add to or modify category sets. Traditional questionnaire coding is not necessary; at data entry, macros fill in fully spelled-out values. These can be prompted by any of several codes or abbreviations[5].

- Macros supervise whether values entered into fields are legal. Illegal entries trigger red cell background and a warning text. Both can be removed by the data entry operator, whose corrections will not be visible to others.

- For each village, data is entered in a separate workbook, in order to safeguard the uniqueness of the identifiers. Field offices send village files to the central office (in large programs, they may first send them to area offices). Off the village-wise data tables, the central office assembles a master file for the entire survey data. Copies will later be shared with the field offices.

Meanwhile, individual field workers or area offices are free to assemble copies of village-wise data sets for their immediate local analysis needs. These assemblies will keep the same record identifiers; calculated or additional variables can thus later be merged.

[5] E.g., "Grameen Bank" by "g", "5", "Grameen", "Gramin", or by whatever the laid-down set for this value. Monitors may also add their own codes as long as they are unique in the category set, e.g. "gram".

[Sidebar:] Spreadsheets for data management - pros and cons

Our choice of a spreadsheet template for survey data management needs justification. In professional research, the gold standard remains the relational database. Even in less sophisticated survey environments, there are serious objections to this use of spreadsheets. This bullet list of pros and cons is eclectic; readers may add more compelling points.

Objections

- Spreadsheets as such do not reinforce the culture of documenting data whereas database applications offer data dictionary features. Knowledge of what the data means is often limited to those currently working on a survey and dies at the end of the project.

- Spreadsheet applications do not provide data entry forms that closely imitate the layout of questionnaire and data form pages. The differences in format make data entry more strenuous and error-prone.

- Database applications have higher standards of data safety, both technically and in the culture of the users. Spreadsheet data are vulnerable, particularly in multi-user projects.

- Complex multi-entity structures are difficult to model in spreadsheets. Even for relatively simple multi-table databases, relational queries take extra skills, effort and devices.

Although this is a specific consideration in poorer countries, it is one more point in favor of database applications: With the spread of Internet access also to remote field offices, decentralized data entry can take advantage of the fact that Microsoft Access or OpenOffice Base files can simply be attached to e-mail messages. Collaborative database use does not depend on an integrated corporate network.

Points in favor

- Database applications depend to a greater degree on external expertise and support. In less well-endowed survey environments, this often does not come forth. Collapse, abandonment or invalid results are frequent consequences of survey data management programs created with such tools.

- Spreadsheets are extremely popular; significant numbers of NGO workers, inside and outside units conducting surveys, use them. Skill levels within units and organizations vary widely; stronger users are available as informal trainers. Inter-departmental coalitions can be built around managers and, below them, around spreadsheet users working on a common task such as the entry, analysis and use of survey data. Spreadsheet skills acquired during surveys are transferrable to other tasks.

- Some spreadsheet applications, including Excel, offer data validation features. Data entry templates can incorporate them in order to force legal entries.

Error control in the NGO culture

Data validation features recognize the fact that by far the largest number of computer applications are created, not by professional programmers, but by end users. Spreadsheet applications have been particularly empowering in this regard, but also particularly error-prone (Burnett, Cook et al. 2002). While "end user programming" is pervasive in the whole computing world, additional considerations apply to survey data, particularly those managed by NGOs in under-resourced situations.

- First, the concept of "total survey quality management" will likely be absent or relegated to something desirable, yet infeasible. In the culture of NGOs that are not chiefly devoted to research, even the more restricted notion of "measurement error" (and its absence in unbiased and reliable measurements) is outside the daily concerns of the majority of workers.

Notions of "processing error" are stronger in units tasked with financial transactions and contracts, such as Accounts, Micro-finance and Human Resources. The mental models to avoid and correct them are borrowed from audits - adherence to manuals, balancing and visual double-checking by other persons. These departments are used to complementing their dedicated computer applications with spreadsheet applications of which they are the end-user programmers.

This discipline is helpful, but not sufficient for quality survey data. By itself, the audit model does not protect from practices that compromise survey data validity. For example, NGO monitors often do not see the need to distinguish between missing data and genuine zeros, on the rationale that sums and counts of positive values are not affected.

- Second, NGO staff working on a survey often have other tasks as well. They may be dynamically assigned to this survey, other surveys and to non-survey work in response to seasons, transport and electricity shortages, organizational priorities and staff changes. Concept consistency (among workers) and concept permanence (over time) can therefore be a challenge in data entry as well. Specifically, and in contrast to professional research agencies, survey data use in development NGOs cannot always wait for the final, "locked and frozen" version of the database. Part of the data may have to be used on an urgent basis, for programming purposes. Additional record entry, corrections, reformatting may go on for considerable time after a first partial analysis. The co-existence of multiple versions of data tables creates its own dangers and burdens.

These observations converge to a larger one as regards NGO-led surveys: it is a reasonable assumption that non-sampling errors are larger than the sampling error. The latter results primarily from sample design and sample size and usually can be estimated. Non-sampling errors are more complex and more difficult to gauge. In NGOs with low data management skills, high work pressure and multi-tasked monitoring staff, it can be extreme.

Ultimately, staff quality decides

Non-sampling error is usually sub-divided into coverage error, non-response error and measurement error (Nicholls, Baker et al. 1997: 224). Data entry errors form part of this third category. It follows that investments in reliable data entry may pay higher dividends than cost-equivalent sample enlargements. Yet data entry does not account for all non-sampling errors.

Therefore, technical training of those collaborating in various survey phases does more for quality than applications focusing on data entry by low-skilled workers do. Since spreadsheet applications are socially more inclusive than database applications, we opt for the former, with all due anxiety about the things that may still go wrong.

Demo file

The template being used in the current (summer 2010) round of the FIVDB Jonoshilon survey accommodates 139 household-level and 31 individual-level variables. A segment of the template has been made into a downloadable demonstration workbook (link). The

file name is "FIVDB_DataEntryTemplate_100820.xlsm", the password is "password" (all lowercase). It will be needed to change protected cells and to access hidden columns in the data sheets. The same password makes the macro code in the VBA editor (press Alt + F11) accessible.

For this purpose, the household-level variables have been reduced to 40, and the individual level ones to 23. Data is included from one village on 20 anonymized households and on the 150 corresponding household members. As many as 14, resp. 15 of the variables, including household head and member names, serve identification purposes. The demonstration workbook retains them all in order to show how the unique record identifiers are formed. The substantive variables in the demonstration workbook were selected chiefly so as to illustrate various types of macro code useful when adapting the tool for other surveys.

We advise readers to have a first look at the workbook before reading on. Following the description of the template's key elements below, some may want to add a few fictitious records to the household and household member data sheets, perhaps even modifying some category sets in the list sheets. Accessing the protected area may be instructive for those closely looking at the macro code.

Key elements

In the basic mechanics of the template, four key elements work together. These are: the sheet structure of the workbook, the protection of sensitive cells and areas, macros attached to the data sheets, and flexible coding of categorical variables.

We describe each element in non-technical language. Subsequent sections will detail the technicalities. Readers not concerned with this detail may then skip forward to the sections commenting on our practical experience with the template.

Workbook set-up

For each surveyed village, monitors save a separate workbook based on the data entry template. The limitation to one village is motivated by concern for the uniqueness of household numbers (taken from the village map). Also it makes supervision easier; all the data on a given village is usually entered by one or two persons.

The template is organized by entity levels, such as for

- village-level data
- household-level data
- household-member level data.

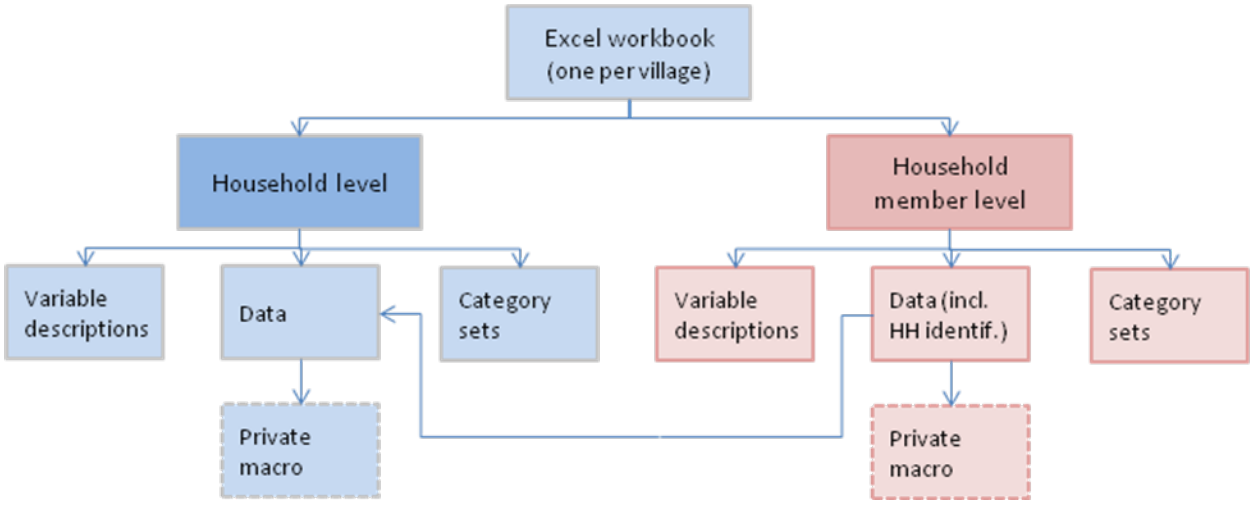
For simplicity, the demonstration workbook operates at two levels, the household and household members.

For each level, three worksheets and a private VBA code section are provided:

- The LevelX_**Vars** sheet holds metadata; it enumerates the positions (column numbers), names and labels (= full names, meanings) of the variables that hold the survey response at this level.
- The LevelX_**Data** sheet is the one in which the data will be entered. Copies of it will be used to assemble the master file in the central office or to do local data analyses in a field office.
- The LevelX_**Lists** sheet holds category sets for categorical variables (e.g., occupation) and the admissible prompts (codes, abbreviations, alternative spellings) for each included category (e.g., typing "Bee" would be enough for "Bee keeping").
- VBA code is attached to each LevelX_Data sheet.

Figure 4 exemplifies the structure of the workbook for the two-level data situation. The arrow between the two data worksheets indicates that household member records are linkable to the household records through the household identifier. The macros are private to each data sheet; they are not in a workbook-level module.

Figure 5: Basic workbook structure for a two-level data situation



Sheet protection

In the data sheets, the pre-defined variable names (column headers) cannot be changed by field personnel. In addition, some columns of a technical nature remain invisible. Field workers can, however, add variables of local interest in the unprotected area of a data sheet.

Figure 6: Structure of a protected data sheet (household level)

RecNo	RunNo	Serial No *[unique within village file]*	Household number *[unique within village]*	*[Other identifier fields, incl.:]*	HHCode *globally unique]*	Name of HH head	*[Substant - ive data fields - standard]*	*[Locally defined data fields]*
		Automatic	Data	Data	Formula	Data	Data	Data
		Automatic	Data	Data	Formula	Data	Data	Data
		Automatic	Data	Data	Formula	Data	Data	Data
		Automatic	Data	Data	Formula	Data	Data	Data
		Etc.						

= Hidden columns; record and running numbers assigned during master file assembly
= Protected field header cells
= Data entry area. Legal values supervised by background macro
= Locally defined variables. Not protected. Data not supervised

The sheet protection facilitates these behaviors:

- Hidden columns are for operations by the central office. In the FIVDB baseline survey, the central office assigns two record identifiers while appending all the village-wise data tables into one master workbook: a running number within each district, and the continuous record number for the full data set.

- The data entry operator in the field can access the red and green areas.

- In the red area, the field names (= the cells in the top row) are locked. They, together with the rest of the table, can be copied.

- The column order cannot be changed. Rows cannot be inserted or deleted (the contents of cells, though, can be removed). Data is entered starting in row 2.

Record-identifying variables are on the left side of the red area. Although the rules of forming unique identifiers do not directly affect data entry (and vice versa), the template can include elements that calculate and control identifiers. Record identifiers are the subject of a sidebar beginning on page 26.

Variables of substantive interest are on the right of the identifier variables, followed by authoring information (interviewer name, date of the interview).

The green area to the right is unlocked and unsupervised. It can be used to accommodate additional variables, some of common use, others of a purely local interest.

What the macros do

Each data sheet has an invisible macro attached to it. The macro kicks in whenever the content of a supervised cell is changed. It does not need to be triggered by a user command. It

- supervises whether cell values are legal in a given variable and flags errors
- checks whether household numbers are unique
- activates the next cell to the right when a legal value is entered
- depending on the response, skips several columns and activates the cell in the next relevant variable
- activates, and scrolls to, the cell at the beginning of the next record when data entry in the current record is complete, automatically creating a new serial number.

The last two bullet points imply that the macro can mimic interview filters and do a "carriage return to the next line".

Modifiable category sets

In the category sets sheets, field workers can modify or add categories in any given set. The changed or new categories will automatically be treated as legal values in the variables using the set[6]. These sheets are unprotected.

Figure 7: Example of a modifiable category set

HHIncomeSource_prompt	HHIncomeSource_value
1	01-Labor sale
01-Labor sale	01-Labor sale
Labor	01-Labor sale
Labor sale	01-Labor sale
4	04-Home gardening
04-Home gardening	04-Home gardening
Garden	04-Home gardening
Home gardening	04-Home gardening
veg	04-Home gardening
Vegetable production	04-Home gardening
Vegetables	04-Home gardening
13	13-Other income
13-Other income	13-Other income
Other	13-Other income
Other income	13-Other income
26	26-Bee keeping
26-Bee keeping	26-Bee keeping
Bee	26-Bee keeping
Bee keeping	26-Bee keeping
Bees	26-Bee keeping

The figure below illustrates the function of a locally modifiable category set with an example of income source options. For simplicity, the number of sources has been greatly reduced. Households earn from the sale of labor and from home gardening. They also draw other income.

Each option can be entered using different codes, abbreviations or related concepts. Thus, for example, "4" is a code for home gardening, "Garden" is an abbreviation, and "Vegetable production" is a related concept. Collectively, we call these "prompts". Entering *"4"*, *"Garden"*, *"veg"*, *"Vegetable production"*, *"Vegetables"*, the full *"Home gardening"* or the pre-fixed *"04-Home gardening"* gives the same result; Excel writes *"04-Home gardening"* into the active

[6] Note that the records using the initial category definitions will not be updated when a set is changed after the start of data entry in the concerned table.

cell in any of several fields dedicated to hold the multiple-response information.

Why the prefixes such as "04-"? By assigning numeric (or alphanumeric) prefixes, users obtain a desired sort order when they later tabulate data. A table containing this variable will thus return "Labor sale" in the first row or first column and "Other income" in the last.

Assume now that a monitor receives several questionnaires noting "bee keeping" as a significant source of income. He wishes to add this as a distinct option to the income sources that the workbook will recognize. All he has to do is to decide on prompts (codes, abbreviations) and on a prefix that assigns the option a particular place in the sort order of future tabulations. The additional pairs of prompts and values are simply written to the bottom of the set area, without any blank rows. In this example, "26", "Bee", "Bees", "Bee keeping" as well as "26-Bee keeping" will all return the same string - "26-Bee keeping".

We will now describe Excel features that were used as essential building blocks for this data entry template. This section is technical. Readers not interested in this detail may fast-forward to page 38. They may still benefit from browsing the demonstration workbook and from practicing data entry in a few fictitious records.

At first we will describe some features that use Excel menu options and in-cell formulas. These are more widely understood than VBA code. Subsequently we will detail the macro structure and the detailed code.

Technicalities: Excel building blocks

Outside VBA

Category sets

The category sets in the lists sheets come in cell ranges with two columns. In the right column, they contain all the legal options; on the left side we find, for each category, all its effective prompts. Technically, this arrangement creates lookup tables that come into play when a cell in the data tables changes content. The sheet-specific macro determines which category set, if any, is to be used for the variable in point. The macro reads the prompt that the user entered (e.g., "26" for "Bee keeping" in an income-source variable) and calls the function VLOOKUP to return the corresponding category in its full spelling.

Every cell range holding a category set is a named range. This named range is dynamic. It expands and contracts automatically if and when elements are added or eliminated[7]. The dynamic ranges in the category set sheet use the functions OFFSET and COUNTA.

[7] Dynamic ranges do not appear in the drop-down name list to the left of the function bar. They do appear in the Name Manager (under Formulas in Excel 2007). We use the formula supplied by OzGrid (OzGrid Undated).

For example, if the set has been placed with its left-column title in cell R1C44, then the formula

```
=OFFSET(Hlists!R2C44,0,0,COUNTA(Hlists!C44) - 1,2)
```

will work as follows: COUNTA(Hlists!C44) returns the number of numeric and text entries in column 44 of the sheet named "Hlists". Since the lookup table must not include the title row, we subtract 1. This is the number of prompt-and-category pairs included in the table. The function OFFSET with the arguments given here returns a reference to a range. This range begins at R2C44 as it upper left cell [the two arguments after R2C44 both are zero, meaning no offset from the start cell]. It is [COUNTA(Hlists!C44) - 1] cells' high (downward from, but including R2C44), and two cells wide (rightside from, but including R2C44).

In general terms, in order to create such a dynamic named range, template designers go to "Formulas" → "Defined Names" → "Define Name" in Excel. Here they enter a name for the range and type in the field "Refers to:", using a suitable instance of

```
=OFFSET([sheet name]!R2C[number of left column], 0,0,
COUNTA([sheet name]!C[number of left column]) - 1,2)
```

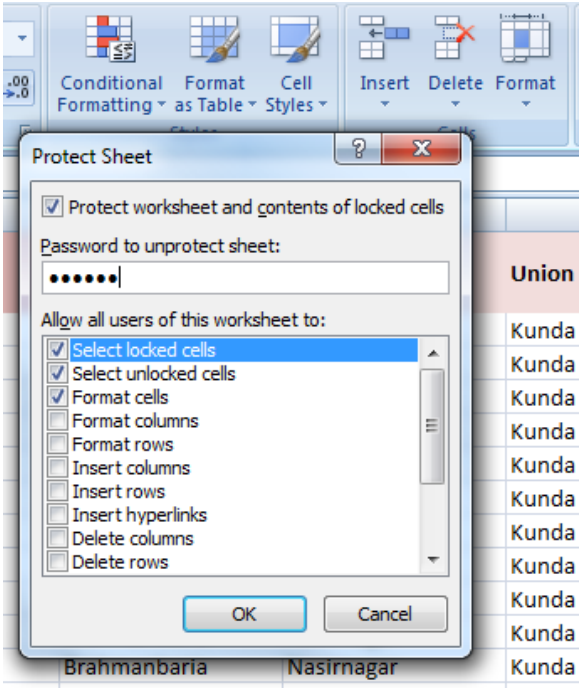
All named ranges in the category set sheets are dynamic. If items are added, eliminated or modified, VLOOKUP will find lookup tables already updated accordingly. Note that cell entries made with earlier versions of the category set will not be updated; they produced static values, not linked formulas.

Several users suggested including also the full spelling of every category in the prompt column. In other words, in the earlier example of "26-Bee keeping" among income sources, a pair *"26-Bee keeping"* [prompt] - *"26-Bee keeping"* [category] would be part of the concerned list. The function of such fully spelled-out prompts is to prevent error messages from actions that, we believe, are rarely taken. That is, an error is flagged when the user returns to a correctly filled data cell, clicks the formula bar (edit mode), does not make any changes, and yet clicks the Enter button (or hits the Enter key). To satisfy this demand, we have included such prompt-category pairs for all categories in our template lists. But it is not obvious why users will want to go into edit mode in a cell that they will not change; thus template designers should feel free to adopt this precaution - or not. If taken, it will indeed forestall an error in this type of action.

Locked cells and sheet protection

Technically, locking field-name cells and protecting the data sheets is not necessary. The macros and adjustable category sets will work without these precautions as long as users only enter data in the assigned sheet areas. However, as mentioned, in earlier survey rounds, we found tampering with the data entry framework - e.g., the order of columns - to be a serious problem and sufficient reason to graduate access to sensitive template elements.

Figure 8: Excel sheet protection menu (under Cells - Format)



Access to cell locking and sheet protection differs among Excel versions. Figure 7 is a screenshot of the Excel 2007 menu command *Home - Cells - Format - Protect Sheet*. We found it expedient to allow field personnel to select both locked and unlocked cells, format cells (to take off the red flag color after corrections), and use AutoFilter[8]. We disallowed the other options. For example, sorting records would thwart the automatic serial number generation; thus the option "Sort" is not offered.

Formulas for household and person identifiers

The populations covered by surveys are members of tiered social and administrative units. In the FIVDB baseline survey, six levels are involved: individual - household - village - Union (the lowest tier of local government) - sub-district, and district. The formulas used for these codes are spelled out in a sidebar since this is not a genuine aspect of data entry. However, the codes combine elements pre-defined by the central office with local ones generated during interviews and data entry. This procedure allows field offices to use data early, before a master file is sent back to them, and keeps households and residents traceable for program participation and follow-up surveys.

[Sidebar:] A way to construct unique identifiers

Instances of essential survey entities (persons, households, etc.) that will be re-checked, linked among tables, or re-visited at later points in time need to be uniquely identified. Names are not unique; moreover, often the same name is spelt differently. Numbers assigned by survey staff need to be reproducible; mere serial numbers correspond to nothing in the outside world. Not all persons have numbered ID documents; even if they did, NGO-led surveys could not use them for privacy as well as practical reasons. Higher-level entities, starting at the village or Union, may have been assigned unique codes in administrative gazetteers. These are available in census reports, but they are not widely used, and not at all by village communities or grassroots organizations.

In the FIVDB baseline survey, unique household codes are formed by concatenating district, sub-district (Upazila), commune (Union), village and household information. In general, it is sufficient to concatenate the first four letters of the administrative unit names. However, in the first survey round, using first-four-letter codes for villages led to errors. Neighboring villages in the same Union might be called X-village-aaaa and X-village-bbb, for which mechanical extraction produced duplicates. Not uncommonly, East-X-village and East-Y-Village types of

[8] The AutoFilter option appears when scrolling down in "Allow all users .. to:".

village names would occur, also prone to confusion. Village codes thus have to be selected manually, to make sure that each be unique.

For the household element, the concatenated identifier uses the dwelling number on the maps that Community Learning Center volunteers had drawn of their villages. These numbers were unique (outlines of buildings with several households were divided by lines). For uniformity, each of the concatenated identifiers consists of four digits. In this anonymized village in Brahmanbaria District, Nasirnagar Upazila and Kunda Union, sample household identifiers are shown in the figure below. Note that serial and household numbers are not identical; dwelling no. 8 was not inhabited, or its residents were not interviewed.

Figure 9: Concatenated household identifiers

	3	4	5	6	7	8	9
1	SerialNo	Hhno	DistrictCode	UpazilaCode	UnionCode	VillageCode	HHCode
7	6	6	Brah	Nasi	Kund	Xxxx	Brah-Nasi-Kund-Xxxx-0006
8	7	7	Brah	Nasi	Kund	Xxxx	Brah-Nasi-Kund-Xxxx-0007
9	8	9	Brah	Nasi	Kund	Xxxx	Brah-Nasi-Kund-Xxxx-0009
10	9	10	Brah	Nasi	Kund	Xxxx	Brah-Nasi-Kund-Xxxx-0010
11	10	11	Brah	Nasi	Kund	Xxxx	Brah-Nasi-Kund-Xxxx-0011

The concatenation uses combined Excel functions. Given the above column numbers, the household identifier is formed by:

```
=CONCATENATE(RC[-4],"-",RC[-3],"-",RC[-2],"-",RC[-1],"-
",TEXT(RC[-5],"0000"))
```

The household member table includes the household identifier, using the same formula. The individual identifiers are formed by concatenating household identifier and serial number. This is a convenience arrangement; there is no natural numbering of members in the households interviewed. Some may prefer to use the serial number of the demographic sheet in the questionnaire, but it is unlikely that this paper document will be at hand in future references. Figure 9 shows the identifiers of a few members of the first two households.

Figure 10: Concatenated household member identifiers

	3	4	9	10
1	SerialNo	Hhno	HHCode	PersonCode
14	13	1	Brah-Nasi-Kund-Xxxx-0001	Brah-Nasi-Kund-Xxxx-0001-0013
15	14	1	Brah-Nasi-Kund-Xxxx-0001	Brah-Nasi-Kund-Xxxx-0001-0014
16	15	2	Brah-Nasi-Kund-Xxxx-0002	Brah-Nasi-Kund-Xxxx-0002-0015
17	16	2	Brah-Nasi-Kund-Xxxx-0002	Brah-Nasi-Kund-Xxxx-0002-0016
18	17	2	Brah-Nasi-Kund-Xxxx-0002	Brah-Nasi-Kund-Xxxx-0002-0017

The household member identifiers, given column numbers, are produced by

```
=CONCATENATE(RC[-1],"-",TEXT(RC[-7],"0000"))
```

In both formulas, the function TEXT with the optional argument "0000" forces the numeric elements to be included with four digits. This is necessary for proper sorting.

The unique identifiers are basic preconditions for linking household and person tables. Survey analysts may want to attach a household variable (e.g. wealth rank) to household member records. In the opposite direction, they may want to attach a function of the household member set to the household table (e.g., the age of the household head). In Excel, this is easily done with the help of the function VLOOKUP and Pivot tables.

The mechanics of identifier production are simple. The template provides the formulas in the first record. Data entry persons copy them downwards. There are, however, motivational challenges. It is essential to explain the function of these constructs to data entry and analysis personnel, both from the head office and field-based. Our training offered intensive practice on how VLOOKUP and Pivot tables serve to link tables. Care for arcane requirements like composite identifiers can be expected of workers who are encouraged to use the data in their local work environments. In other words, workers who feel they have an intrinsic stake in the data, and who understand that the tedium of data entry ultimately serves an intelligible and personal aim.

Entering multiple-choice response data

Experience shows that NGO monitoring staff often enter multiple choice data in formats that make later analysis cumbersome or nearly impossible. In subsequent FIVDB survey rounds, this was a problem which we gradually overcame. The necessary training had to address both conceptual and technical questions[9].

At first glance, this problem seems to have nothing to do with data entry as such. However, the formatting of multiple-choice data that is the most convenient for the analyst may not be the most efficient for the entry operator, and vice versa. Some statistical programs have created special procedures (e.g. for STATA, Jann 2005), reason for us to look at the problem more closely in a sidebar. For data entry in Excel, we favor a particular approach. In a second step, at the time of analysis, the original variables can be transformed into a format more amenable to tabulation and modeling.

[Sidebar:] Treatment of multiple-choice response data

In a typical example, a household may report loans taken from one or several microfinance providers. These provider instances (or for that matter any property of the loans) can be handled differently in data entry. If the distinct providers are few and known in advance, the

[9] We found that not all field monitors would immediately grasp the difference between "the percentage of households that have a loan from the Grameen Bank" and "the percentage of current loans that are from Grameen".

template may provide each of them with his own field. Technically, for each provider a binary (or, if you like, indicator) variable (yes/no, 1 or 0, and the like) is created; at data entry each is filled with the positive or negative value (unless there was no response to the question, in which case they are left blank). This representation is known as the indicator mode (Jann, op.cit., p.93).

Figure 11: Sample records with multiple-choice data in indicator mode

	3	4	88	89	90	91	92
1	SerialNo	HHno	ASA	Grameen	BRAC	FIVDB	MF_other
13	12	12	1	0	1	0	0
14	13	13	1	0	1	0	0
15	14	14	1	0	0	0	0
16	15	15	1	0	0	0	0
17	16	16	1	0	0	0	0
18	17	17	0	0	1	0	0
19	18	18	0	0	0	0	0

If the choices are numerous, or if some of interest are not known in advance, it may be easier on the data entry persons as well as analytically beneficial to define multiple generic variables like "Provider1", "Provider2", etc. For this solution, adaptable category sets are part of the technical answer. At data entry, monitors add providers not yet included in the default list. The macros then ensure that the correctly spelt provider name is inserted into the field. Entry continues for as many providers as were reported by the household. The additional generic fields remain blank. This is called the polytomous mode, meaning: with more than two choices.

Figure 12: Sample records with multiple-choice data in polytomous mode

	3	4	18	19	20	21	22	23
1	SerialNo	HHno	NGOmemb	NGO1	NGO2	NGO3	NGO4	NGO5
13	12	12	1	BRAC	ASA			
14	13	13	1	ASA	BRAC			
15	14	14	1	ASA				
16	15	15	1	ASA				
17	16	16	1	ASA				
18	17	17	1	BRAC				
19	18	18	0					

Polytomous multiple-choice formats are analytically cumbersome. Their translation into a set of indicator variables is easily accomplished with a formula that combines the functions IF, MATCH, and ISERROR, and makes use of mixed cell references. Given that the polytomous data was entered in columns 19 - 23, it is extracted into indicators by

```
=IF(ISERROR(MATCH(R1C,RC19:RC23,0)),0,1)
```

as in this screenshot. The formula is identical for all indicators of loan providers.

Figure 13: Transforming polytomous variables into indicators

	19	20	21	88	89	90
1	NGO1	NGO2	NGO3	ASA	Grameen	BRAC
13	BRAC	ASA		=IF(ISERROR(MATCH(R1C,RC19:RC23,0)),0,1)	0	1
14	ASA	BRAC		1	0	1
15	ASA			1	0	0
16	ASA			1	0	0
17	ASA			1	0	0
18	BRAC			0	0	1
19				0	0	0

For smaller microfinance providers that are rarely found, specific indicator variables may not be useful. Their presence in the survey households can be lumped into a residual count variable, such as MF_other in column 92 above. The variable is calculated as the difference between the polytomous entries and the sum of the indicator variables, i.e.:

```
= COUNTA(RC19:RC23) - SUM(RC88:RC91).
```

Later analysis may then proceed through Pivot tables or through named ranges used in functions. In the example, let us assume that we calculate the number of loan providers to whom a household is indebted as "NoProviders", and thus name the data vector as well[10]. Assume that HHcode, the household code, is also the name of the range holding all the codes in the table, and that the maximum number of current loans from the same provider to a household is one.

These conventions let us conveniently calculate subject-based as well as instance-based quantities. For example, the percentage of households that currently have loans from any provider can be calculated as

```
= COUNTIF(NoProviders, ">0") / COUNTA(HHcode)
```

The average number of loans obtained by survey households is

```
= SUM(NoProviders) / COUNTA(HHcode).
```

Once monitors understand the polytomous and indicator modes, both entry and analysis of multiple choice data can proceed efficiently, and with few additional devices needed. It is obvious that this approach has its weaknesses. Notably, the control of missing values is not well handled.

In the VBA code

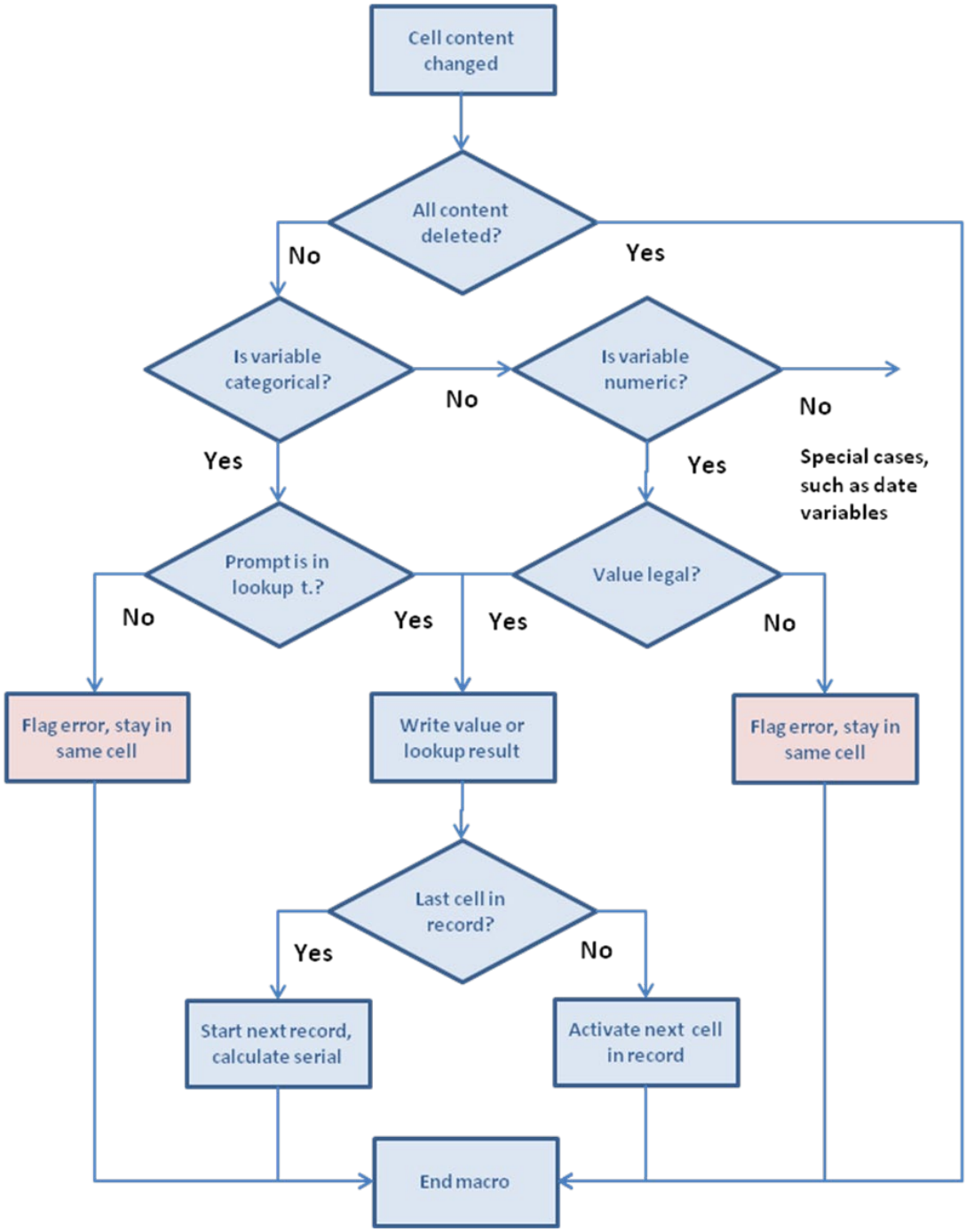
The major functions of the macros attached to the data sheets were enumerated in non-technical language (page 23). Whenever the content of a data sheet cell is changed, Excel performs a series of instructions that are not visible in any of the worksheets. They are instances of so-called event macros[11] triggered by a worksheet change. In fact, the template has only two such macros, one attached to the household level data sheet,

[10] In our example, COUNTA(RC19:RC23) is one option to calculate this number. Excel 2007 permits efficient data column naming via the menu command sequence Formulas - Create from selection - [check] Top row. Earlier versions work through Insert - Names - Create, etc.

[11] For a superficial introduction, but with many code examples, see McRitchie (2008). Pearson (2010) provides a more systematic introduction, with a steep learning curve.

the other to the household member sheet. The macros are tailored to respond to changes in the values individual cells (not to those affecting a range of cells). They both follow the course of action abbreviated in the flow diagram below.

Figure 14: Flow chart of basic event macro action



The detailed code that executes action in this flow diagram is given in the appendix. Blocks of code lines that detail the response to particular subsets of variables are contained between lines and start with appropriate comments. This should enable readers with basic VBA knowledge to adapt the macros intuitively to the needs of their particular data entry templates.

Obviously, the question of more general interest is how the macros recognize types and subtypes of variables and supervises cell entries appropriately to each.

Before writing any macro code, the template designer determines what treatment each variable should receive. Treatments must determine the response to these questions:

- What entries are to be considered errors?
- What error message(s) to print?
- For categorical variables, which lookup table to use?
- Which next cell to activate if no error occurs?

Some variables call for the same treatment. For example, the fields that hold loan providers (such as our NGO1, NGO2) justify uniform treatment. Other variables demand a special treatment each for itself alone. For example, the serial number at the beginning of the visible record must be consecutive and unique integers. For the household numbers, we expect unique integers, but gaps are allowed (dwellings without an interviewed household).

At this point, we will only exemplify each of various ways of letting the macro know whether a variable is to be treated individually or as a member of a group of variables all calling for the same treatment. We give brief examples and hyperlink them to the beginning of the relevant code block in the appendix.

As the event macro executes, it compares the name of the variable in which the change occurred to a sequence of instructions. Code for specific variables, or for groups of variables all to be treated the same way, is written in blocks separated by dash lines. If the variable belongs in a particular block, it executes the instructions and (after some internal house-keeping) terminates. If the name is not found yet, it continues evaluating membership block by block. If the variable belongs nowhere, the macro terminates without visible action. This is the case for some standard variables (e.g., the four-letter codes for administrative units) as well as for any local variables created outside the protected data sheet area.

In event macros, the range being changed is known as the "target". This holds also for single-cell changes. The macro acquires the name of the concerned variable through the expression "Cells(1, Target.Column).FormulaR1C1". It is used numerous times; for brevity, a convention is made before the first use, defining

```
t = Cells(1, Target.Column).FormulaR1C1.
```

Henceforward, the macro "understands" that t refers to the target variable name.

Treatment of a single variable

To provide a treatment for one variable only, the macro is made to look for its exact name. If it finds the name, the treatment code is executed, and the macro ends. Else, the macro moves on to the next block. In the example of the serial number, the instructions begin with an if-clause:

```
If t = "SerialNo" Then
      [execute treatment]
      [move to end of macro code]
Else
      [do nothing]
End If
      [move to next block]
```

The code block for this variable can be inspected through this link.

Treatment of multiple variables

For efficient programming and execution, variables that receive the same treatment are addressed in the same code block. Two procedures are used in our template - including all the concerned column head cells in a named range, and exploiting common strings that occur in all variable names to group, and only in them.

Grouping by defined range

We begin by defining a named range holding the first-row cells of all the variables to be grouped, and no other cells. For example, by selecting the cells that hold the names of the multiple income sources, we can define the name "FieldsIncomeSource", referring to

```
=HHlevelData!R1C15, HHlevelData!R1C17, HHlevelData!R1C19,
HHlevelData!R1C21, HHlevelData!R1C23, HHlevelData!R1C23
```

This is an example of a multi-area range. We then use an Excel method known as "Application.Intersect", which returns a range of cells that occur in all of the intersecting ranges. In our case, we let two ranges intersect - the just now defined "FieldsIncomeSource" and the first-row cell of our target variable. If the latter is indeed an income source variable, then the intersection is not empty. This makes it possible to treat each of the grouped variables the same way:

```
Set isect = Application.Intersect(Range( _
      "FieldsIncomeSource"), Cells(1, Target.Column))
'Note: The choice of this range name as isect is arbitrary.

If Not isect Is Nothing Then
      [execute treatment]
      [move to end of macro code]
Else
      [do nothing]
End If
      [move to next block]
```

The code block for this variable can be inspected through this link. The intersect method has the advantage that variables can be added to or removed from the group by simply changing a name definition.

Using common strings in variable names

Variable names may have text elements in common, such as in "LoanSource1", LoanSource2", etc. The VBA function "InStr" can be used to determine whether the relevant text element occurs in the target variable name. If it is not found, InStr returns 0. Remember that "t" was defined to abbreviate the target variable name. Thus, this code is fit for all variables whose names begin either in "Organisation" or in "LoanSource", regardless of how many they are:

```
If InStr(1, t, "Organisation", 1) > 0 Or _
           InStr(1, t, "LoanSource", 1) > 0 Then
      [execute treatment]
      [move to end of macro code]
Else
      [do nothing]
End If
      [move to next block]
```

The code block for this variable can be inspected through this link. The InStr function is useful if a text string, or several text strings, are used in multiple variable names.

Other elements of the macro architecture

Temporary worksheet to transfer categorical values

The treatment of categorical variables in the macros is not straightforward. The new target cell content - what we call the "prompt" for a specific category - is used as the lookup value in the concerned category lookup table. These tables are held outside the data sheet, in the lists sheets. The function VLOOKUP then returns the corresponding fully spelt-out category or the error message #N/A.

However, the target cell cannot be updated with a self-referential formula such as

```
Target.FormulaR1C1 = _
      "=vlookup(Target.FormulaR1C1, [lookup table], 2, false)"
```

This produces an error. Instead, the result first has to be stored in a named cell. In earlier versions of the macro code, we designated a cell in the data sheet itself to be the transfer point. However, this solution was cumbersome and even error-prone. If on the right side of the protected variable names, it could interfere with the addition of local variables. If in a hidden column on the left, it would confuse central office staff when they unprotect the sheet to append it to a master file.

One solution is to add a temporary worksheet that does the transfer job and is deleted just before the macro ends. The data entry person never sees it. Thus, in the opening sections of the macro, we find these lines:

```
Set TempSheet = ActiveWorkbook.Worksheets.Add
    TempSheet.Name = "TempForTransit"

'Name a transit cell in it, to transfer lookup results:
ActiveWorkbook.Names.Add Name:="TransferCell",_
      RefersToR1C1:="=TempForTransit!R1C1"

'Transfer the new target cell content:
Sheets("TempForTransit").Range("A1").FormulaR1C1 = _
      Target.FormulaR1C1
```

Later, when the code finds the variable in point, it calls on the TransferCell:

```
If t = ["variable name"] Then

    Target.FormulaR1C1 = _
      "=vlookup(TransferCell, [lookup table], 2, false)"

    Target.Copy 'These two lines replace the formula with its
value
    Target.PasteSpecial Paste:=xlPasteValues, _
      Operation:=xlNone, SkipBlanks:=False, Transpose:=False

    Application.CutCopyMode = False
      'Removes the blinking border from the copied cell
[Etc.]
```

Just before the macro ends, the name of the transfer cell and its sheet are deleted:

```
   Application.DisplayAlerts = False
      'To avoid a prompt when deleting the temporary sheet

    ActiveWorkbook.Names("TransferCell").Delete
    Sheets("TempForTransit").Delete

[some housekeeping instructions]
End Sub
```

We are convinced that there must be more elegant solutions to this problem, and are grateful for suggestions.

Skipping variables

The command

```
Cells(Target.Row, Target.Column + 1).Activate
```

activates the next variable in the current record, one column to the right. This is the default step in row-wise data entry.

Depending on the response to a question, one or several of the next variables in the data sheet may remain blank. In the interview, the corresponding questions were skipped; in

the data sheet, a cell several columns to the right is activated. For example, records of households without any current loans will cause eight variables to be skipped:

```
ElseIf Target.FormulaR1C1 = "No" And t = "LoanCurrentAny" Then
            Cells(Target.Row, Target.Column + 9).Activate
```

However, data entry templates may undergo changes, some of which may affect the composition and order of the variables. In this case, a set number for the variables to skip (as in "Target.Column + 9") may lead to errors. It is safer to define the column number of the next cell to activate in a different way.

By way of example, the monitors entering data of this baseline survey are allowed to copy geographical information from the first records since it stays the same in a one-village file. Therefore, once the household number has been accepted, the cursor jumps directly to the name of the household head as the next variable. Since, for whatever reasons, the number of fields between household number and name of the household head could change, a different approach is chosen:

1. We name the cell that holds the variable name. In the example we name cell R1C14 as "HHheadColHead" (admittedly not very elegant as a name!).
2. In the macro code, before any variables are treated, we refer to the column number of this variable by
   ```
   HhhColHead = Range("HHheadColHead").Column
   ```
3. In the code block for the household number, the "activate next variable" instruction is
   ```
   Cells(Target.Row, HhhColHead).Activate
   ```

This arrangement makes the skipping immune to variable order changes.

Skipping and scrolling when no information is to be entered

Worksheet_Change-type event data execute only if and when the cell content is changed (other than as the result of a formula calculation). Hitting the Enter key does not constitute a change. The cell below, rather than any to the right, is activated.

However, it is desirable to be able to mimic the skipping and branching behaviors of the questionnaire. A workaround is needed. For categorical variables, a one-letter prompt can be defined in the category set in point and can be entered in the cell from which the operator wishes to skip to the cell in some other column to the right. The macro can then be refined to scroll to the location when this prompt is entered. However, the column number needs to be passed to it through an appropriate range definition.

We demonstrate this with the variable set "Income sources". If no further sources are to be entered, the cell in first column of the next relevant block is to be activated and scrolled to. In this template, this is the variable "LoanHistory".

We add the prompt "z" to the list of IncomeSource codes in the sheet "Hlists", leaving the corresponding value cell empty.

To make the column number immune to changes in the template, we define its column header cell as the named range "LoanHistoryColHead" and put this convention among the local range names near the beginning of the macro code:

```
LHistColHead = Range("LoanHistoryColHead").Column
```

Entering "z" in the income source data cell does not cause an error. But it must not just move the active cell by one column to the right. This ElseIf clause takes care of the "skipping" (in the interviewer behavior sense) and scrolling:

```
ElseIf Target.FormulaR1C1 = "0" Then
        Target.FormulaR1C1 = ""
        Cells(Target.Row, LHistColHead).Activate

        'Scroll to active cell,
            'but make sure previous record remain visible:
         Application.Goto _
         Reference:=ActiveCell.Offset(RowOffset:=-1), scroll:=True
         Cells(ActiveCell.Row + 1, ActiveCell.Column).Activate
```

The full code block for this demonstration can be inspected through this link. This skipping by entering a one-letter code has not yet been practiced in the field[12]. Skipping structures for other variable sets have not yet been added to the macros, with the exception of the last field in the household member table, "Comment".

Data from one or from several villages

At present, field-based monitors enter the data from one village only per Excel workbook. They are free to copy the tables of several workbooks and collate the data in a separate workbook for their local analysis needs. This will happen anyway at the time of building the master tables in the central office.

The restriction to one village is motivated by the control of household numbers, which must be unique within a village.

However, not every one may want to follow this arrangement. To allow data from several villages into one workbook, the uniqueness check for household numbers (not for serial numbers!) can be switched off:

In the declaration section of the macro attached to the household-level data sheet, the declaration section includes:

```
Dim OneVillageOnly As Boolean
'If true, HHlevelData table only for one village
'If false, uniqueness of household numbers not enforced.

OneVillageOnly = True
'True (=restricted to one village) is the default.
```

[12] And will be seen by some as an inelegant solution. Suggestions for code performing these actions upon simply hitting the Enter key are very welcome.

```
'Not checking uniqueness of household numbers
'opens the door to indexing errors.
```

The code concerning the household number at first checks whether a number has been entered. If so, then this subordinate if-clause is executed:

```
If OneVillageOnly = True Then
[code to check that entered number is unique]
[if not, error message]
[if yes:]
        Cells(Target.Row, HhhColHead).Activate

Else
        'If not restricted to one village:
        'No uniqueness check. Go directly to:
        Cells(Target.Row, HhhColHead).Activate
End If
```

We reiterate that multi-village data entry workbooks pose added risks. Without control of the uniqueness of household numbers within a village, duplicate household codes may ensue, and eventually errors in table linking. Also, if monitors copy geographical information downwards without proper attention to the point where entry of a different village begins, all floodgates for serious errors are open.

These precautions must not instill fear. They address real problems in the work of data entry and analysis personnel. The code examples discussed above, however, are of concern mainly to those who wish to adapt macros for new data entry templates. Once these work, the macros are like invisible hands, quietly guiding and reducing anxiety.

Experience working with the template

During summer and autumn 2010, eleven monitoring associates, most of them posted to field offices, entered data from approx. 860 household survey questionnaires into village-wise workbooks that used this data entry template. Those at the head office supervised a number of interns and hired hands who processed another 2,040 questionnaires.

In addition, FIVDB, as part of a plan to expand computer skills and facilities among the Jonoshilon Community Learning Centers (CLCs), contracted some of the data entry out to a computer center that seven CLCs had been jointly managing since 2007. The center has so far processed about 6,000 household questionnaires. In fact, the FIVDB monitoring coordinator considers this template the precondition for the successful outsourcing of most of the data entry. We will report on this experiment in the sidebar starting on page 42. Here we concentrate on the performance of the new entry template as it has been used inside FIVDB.

Speed

The monitoring associates agree broadly that it takes about 30 percent less time to dispose of a household survey than it used to under the previous, drop-down list-based template.

But there are aspects of the new template that the monitoring associates resent, and which cost them time. The built-in sheet protection forbids cell comments; these had been popular for a variety of purposes. Cell comments served as "to do" items for values that required call-back with interviewers, and as justifications of outliers that otherwise would provoke queries. Comments now have to be written into an unprotected field to the extreme right of the tables, necessitating long moves from and back to the entry cell last opened. Similarly, the undo button no longer works. When an associate makes an entry error, the cell content has to be deleted, and the red cell background flagging the error manually reset to "no fill". This requires alternating between key and mouse movements.

Some users would have preferred to first enter the names of all members in a given household rather than finishing the record for one member and then proceeding to the next member record. They felt that the entry, vertically in the columns, of names, age, and occupations would save considerable time. The template does not offer the option to offset the next active cell vertically for certain variables.

There appears to have been limited use made of the option to add categories or to create additional prompts for existing ones, in multinomial variables. Additional categories with one-letter prompts were created for NGOs that were listed locally, and had not yet been included in the pre-installed category list.

Reliability

Data that the monitoring associates in the field offices themselves enter reaches the head office as digital files, some sent by e-mail. The questionnaires remain in the field offices. The PPR Unit coordinator and research associates perform a number of consistency checks.

Inconsistencies - e.g., between "student" as occupational status and age outside the typical range for students - are rarely found. Even assuming that part of the entry errors are not caught, they appear much rarer than errors committed during the interviews. This belief is based on the number of errors that the associates detect during data entry. Nevertheless, some users requested built-in validation rules, such as a minimum age for those with a known occupation.

When we contract out data entry work, we use more extensive checks. For an illustration, see the sidebar below, especially on page 43.

Assembly of master tables

The fixed order of fields in the template - columns cannot be inserted, deleted or moved - makes the append operation - the stacking of datasets from several villages one upon the other - straightforward and error-safe. We have tested this and have not found any problems in this respect. In fact, the certainty of correctly appending datasets is one of the strengths of the new template.

Appended data sets do not by themselves make a master table. Record IDs have to be created in the leftmost column. The categories and values used in the variables have to be inspected for missing values, and whether they are consistent and within reasonable range.

These requirements have not changed with the new template. However, since the field offices enjoy the freedom to add categories locally, the final category sets have to be reviewed all the more carefully. Some of the variables may not be usable in their initial raw form, and new variables may have to be recoded from them. For example, age is easier accessible through distinct age groups. Indeed, one of the key competencies in survey data management, closely following data entry, is the ability to recode variables. We devote the next sidebar to this operation.

Shortcomings

How well field and head office workers, collaborating through this template, succeed in producing master tables of the survey data cannot yet be finally assessed.

A common complaint relates to the requirement to enable macros. Some monitoring associates started entering data into copies of the template that they had saved in a non-macro-enabled format (.xlsx, instead of .xlsm, in Excel 2007). Some who shared computers with other workers in common field offices found that security levels had been set such that macros would not work. Particularly new associates did not always know how to resolve these problems.

One of the ambitions using this template is to encourage the early use of data in local field offices, long before the head office distributes copies of master tables that combine data from several offices. Field-based associates know how to use the workhorse of descriptive statistics in Excel - Pivot tables -, but they find it inconvenient that they have to make separate copies of the data tables in order to create Pivot tables. These cannot be made off the protected sheets.

[Sidebar:] Re-coding variables - an important skill

On page 23, we saw an example of a field-based monitor creating a new occupational category, "bee keeping". Her colleague in another field office may not know of this initiative, may know of it but not follow it, or may find it equally important but call it "apiculture". Also, she gets a number of interview returns that report "tubewell mechanic" as a first occupation and decides to make this into its own category, in addition to the already included simple "mechanic".

When the central unit appends the village-wise tables into the full-survey master tables, care must be taken to evaluate the combined category sets, to absorb excessive, incoherent or rarely used categories into broader ones, and to be clear about the rationale for the final number and content of distinct categories.

In many NGOs, the central unit does not give its field monitors any discretion to adapt response categories although they can be meaningful both locally and organization-wide. The reason for fixing sets in advance is the lack of data management skills, specifically the inability to gain an overview of all categories created in the primary data entry and to prune back excessive and incoherent ones.

In Excel, a full listing of all variants used in a variable is quickly established, together with their frequencies, by way of a Pivot table. The variant list can be copied into a fresh sheet, to create a table of variants and their desired replacements. Conceptually, this is equivalent to the operation of recoding a categorical variable known from statistical programs. Technically, an Excel spreadsheet variable is best recoded into a new variable using the function VLOOKUP. One of the charms of Excel is that it is easy to try out different recoding schemes before settling on a final one. We will presently demonstrate this for a numeric variable.

In low-skill monitoring units, dealing with numeric variables is often even more inept. For example, interviewers are instructed to record household income by ranges, e.g. $ 0 - 10,000, $ 10,001 - 20,000, etc. This creates a categorical variable where the continuous variable would be much more informative. Often the ranges fixed in advance have no theoretical or practical grounding and lead to invalid conclusions while obscuring the really important aspects of the distribution. This approach is nevertheless chosen because the monitoring unit does not know who to handle continuous data.

Age is another example. Rounded up or down to full years, it is a discrete variable, but with so many values that a treatment by age groups will be necessary for certain analyses. How these age range categories are to be formed depends on analytic purposes, and should not be fixed beforehand. Again, the function VLOOKUP, properly used, allows us to be flexible. One can have several recoding schemes side by side.

Figure 15: Alternative recoding schemes for a continuous variable

vlookage | fx | LowerBound

	1	2	3	4
1	LowerBound	Scheme1	Scheme2	Scheme3
2	0	00 - 09	00 - 09	00 - 04
3	5	00 - 09	00 - 09	05 - 09
4	10	10 - 19	10 - 19	10 - 14
5	15	10 - 19	10 - 19	15 - 19
6	20	20 - 29	20 - 29	20 - 39
7	30	30 - 39	30 - 39	20 - 39
8	40	40 - 49	40 - 49	40 - 59
9	50	50 - 59	50 - 59	40 - 59
10	60	60 - 69	60 and over	60 and over
11	70	70 and over	60 and over	60 and over
13		Equal ranges	Equal ranges	Narrow ranges
14		until age 69	until age 59	for youth
15				Young adults
16				Mature adults
17				Old people

The screenshot shows a range named "vlookage" offering three different recoding scheme for age. For example, a 6-year old child would be placed in the age group 0 - 9 years[13] in scheme #

[13] The notation "00 - 09" ensures the proper sort order in Pivot tables.

1 and 2, and in age group 5 - 9 years in scheme # 3. In the household member table, a new variable "age group" can thus be calculated, for the chosen scheme, as

= VLOOKUP*([age lookup value]*, vlookage, *[SchemeNo]* + 1, TRUE).

Similarly, categorical variables that require an exact match with their recodings, are easily recoded with the help of VLOOKUP(.., .., .., FALSE). The data management training of the FIVDB monitoring staff devoted a full day to recoding techniques in Excel. The formula syntax, it turned out, posed a minor challenge in comparison with the initial conceptual difficulties.

Recoding forms a necessary counterweight to the liberty afforded to fieldstaff to modify categories in a data entry template. Both local initiatives and central correctives can work together for a vivid and valid rendering of what the categories are meant to capture. The conceptual effort needed in order to produce a meaningfully recoded category set is often underestimated. In the demonstration workbook, for example, the education categories have been worked out carefully, with parallel ordered sets for functional literacy, secular school systems and koranic education. By contrast, as the user will quickly notice, the occupational categories are little more than a collection of trades and titles, demanding a great deal more work of meaningful recoding.

[Sidebar:] Contracting data entry out to grassroots groups

Between August 2010 and February 2011, the Khagamura ICT (short for "Information Communication Technology Centre"), a computer center for which the CLCs of seven adjacent villages have taken responsibility, entered data from over 6,000 survey questionnaires for FIVDB. The center workers were paid for their work at a set piece rate.

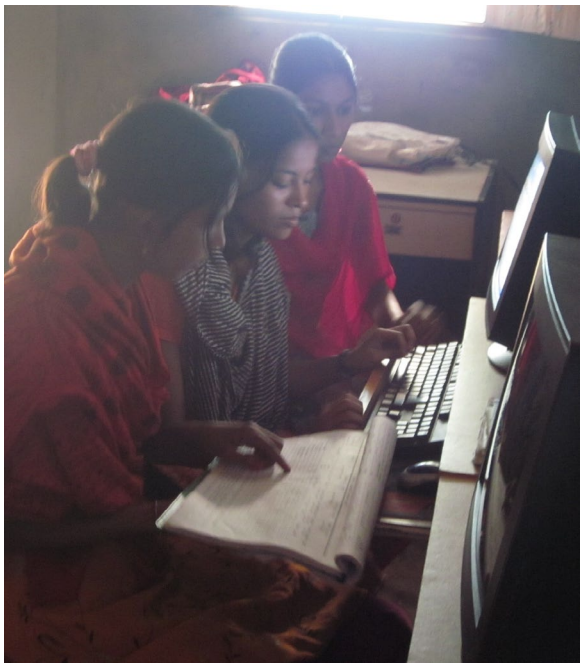
The center is located some 30 km north of Sylhet city, and about 40 km from the FIVDB head office, close enough to benefit from urban resources, and far enough for an FIVDB support person to devote half a working day for any service call.

Founded in March 2007 with help from UNESCO, the center was financially supported by FIVDB until June 2008 and was then told to become self-supporting. The participating CLCs kept it alive through subscriptions that paid for rent and electricity. Practically, the center found little commercial activity - most of it in leaflets and invitation cards -, and a good part of its equipment broke down, three of its five computers and, for some time, the internet access. It badly needed other income.

Mixed motivations

A medley of motivations prompted FIVDB to entrust part of the baseline survey data entry to such a little-proven outfit. Financial viability and the idea of involving the CLCs in the management of the data that concern their communities were two of them. Other anticipated benefits included relief for the monitoring associates, who, in theory at least, could then spend more time in the field, and the need to find a viable model for the ICTs. Under the Jonoshilon project, FIVDB has undertaken to open, by 2013, as many as 25 IT resource centers in the project villages and to expose close to 8,000 CLC members and Class V primary school students to computers.

Figure 16: Data entry in a computer center in the villages



The seven CLCs participating in the Khagamura center chose from their ranks a "facilitator", a person with some minimal computer experience, as the pointperson vis-à-vis FIVDB and coordinator of daily activities. The facilitator appointed six data entry persons from the villages. Four of them were students, including a diploma student in computer science, and one who had followed several computer-related trainings. The other two were high-school students, who received their first computer training on the job. So did the other too entry persons, who were from CLC member households. The social relationships between these persons and their communities conform to a pattern familiar in grassroots organizations in Bangladesh, which often manage to attract the expertise of middle-class persons with a minimum of English language and technical skills.

Problems and perceptions

In the first phase of the contract, the FIVDB monitoring unit inspected five percent of the questionnaires entered by the center. About nine out of every ten questionnaire captures were flawless. Problems have arisen chiefly of two kinds. Poor English has produced coding errors (e.g., remittance coded as "rental income"). Incomplete questionnaires left the center operators at a loss; they had no access to the interviewers; gaps remained unfilled. In response to those problems, FIVDB then decided to route the Khagamura center work back to the concerned field offices, for checking and cleaning by monitoring associates.

Understandably, the cooperation with Khagamura has led to differing emphases in the opinions of FIVDB staff:

- Higher up in the hierarchy, there is a conviction that monitoring associates must be relieved of data entry chores, in order to do more valuable work, and that the CLC-related center offers both a controlled environment as well as a seedbed for computer skills that can ultimately be commercialized in response to growing demand for such services in the diversifying Sylheti economy. Since the Khagamura center was started, more than a hundred girls have had the rare chance to learn about computers; this was enough to help some of them to find jobs and to attract the attention of the sub-district administration.

- Technical staff in the head office urge a reality check in terms of the effort it takes to maintain equipment and to chaperon the social organization supporting it; already FIVDB has had to reorganize the Khagamura committee when it became obvious that only two out of the seven CLCs had participated. In fact, in early 2011, the four most active volunteers moved the equipment to a room rented from a house owner in their extended family, thus essentially making this ICT a private concern.

- Finally, the monitoring associates in the field offices pointed out that shifting the data entry away from them deprived the survey of its field editing function: the only way to

perform anything close to field-editing is to check questionnaires during data entry in the very offices to which the interviewers (who, in their normal duties, are frontline workers of FIVDB line departments) report daily, and to tackle anomalies directly with the concerned ones. CLC-mediated (or, for that matter, any other outside) data entry arrangements could not routinely access the interviewers. This point was compelling; as we have seen, an additional loop was added to the process, by sending data files back to field offices for checking.

A learning process

Outside observers would probably make three points:

- First, innovation is ongoing; the pull of opportunity seizes even something as mundane as survey data entry.

- Second, as often in NGOs, objectives become hybrid. The original function of data entry has been vastly stretched to accommodate financial, organizational and training considerations that are extraneous to the logic of surveys.

- Third, work processes of the same kind become diverse, with part of the data handled under one organizational arrangement, and another part under a different arrangement. This is likely to reinforce the tendency to produce data with strongly correlated errors. These occur because different instructors of the survey personnel give different guidance (Bassi and Fabbris 1997) - say, one set of instructions for monitoring associates vs. the re-interpretations that semi-autonomous contractors make of them.

On the other hand, a diversity of a work processes has the advantage to elaborate organizational alternatives. The Khagamura center workers have learned to use the new template, after only minimal training on the job. FIVDB considers this an investment into a learning process in grassroots organizations. At the same time, FIVDB itself benefits from outsourcing data entry. Its monitors agree that the template, with its error detection facility and efficient prompts, is the precondition for this arrangement to work.

Outlook

We offer a template for survey data entry that may be adaptable to the needs of a range of development NGOs. It may attract organizations that have found that their own ad-hoc arrangements for data management do not suffice, and that they need tools that make the entry process more efficient. Within this group, the template is suitable for those which use the spreadsheet program Excel, rely on trained or trainable users to enter data, and have access to the (modest) macro code writing skills needed to adapt it to the tables and variables into which their survey data will flow.

Those are basic requirements. One of our higher purposes is to encourage decentralized data entry, particularly with a view to rapidly plowing back data to grassroots organizations that helped collect them, and to promote and facilitate analysis and usage in and around field offices. FIVDB has invested in the training of its monitoring associates to a point where most of them know, not only how to enter data, but also how to produce simple descriptive statistics. This multiplies possibilities of supporting field units and communities with useful information. With no extra effort, some of it can be

cast in a comparative mode that goes beyond the participatory analysis forms contained within one small community, yet remains accessible and stimulating to deliberating citizens.

Decentralized data entry is not without its challenges, and it should not be romanticized. Professional supervision may be more difficult to extend. There are costs. FIVDB has had to install generators in field offices to ensure adequate computer access for survey data entry as well as for other uses. We assume, but do not have proof, that data entered in field offices is somewhat more reliable than under centralized arrangements, chiefly because the monitors benefit from almost daily interaction with the interviewers.

This serves as a second-best to the kind of formal field-editing that we would see in a professional survey setup, but it does eliminate a number of errors, to a degree that centralized operations achieve with difficulty. This is in line with what others have found. In a randomized experiment with decentralized data entry in Vietnam, Glewwe and Dang (2008) concluded that decentralized entry reduced errors and inconsistencies. It saved time, particularly in data checking.

Realistically, this template may serve a mid-range group of NGOs engaged in surveys. "Mid-range" applies both to current human resources and to the dissemination of new technology. It is not helpful for groups that do very small surveys, or who cannot find the skills to adapt the macro code. It will not be attractive to those with well-endowed research departments nor to those who contract out most survey work. Moreover, the advances that are made in mobile phone-based data entry, at present most resolutely among relief agencies, will ultimately be embraced by development NGOs as well. Technology will make paper-and-pencil interviewing and subsequent data entry obsolete.

We are not yet there. The template, as developed and tested in FIVDB, may find a congenial audience in NGOs that still use paper and pencil, are concerned about data reliability, and appreciate spreadsheet technology as a platform on which workers of diverse professions, skill levels and departmental functions can meet, learn and collaborate. In fact, as we have happily observed since the first round of the Jonoshilon baseline survey, the survey process itself changes role definitions. Monitoring associates who used to be looked down upon as lowly clerical workers are actively consulted for data management problems beyond surveys. Other departments have started pinning hopes on the monitoring unit's savvy and discipline to help them move towards data collections that might answer questions of outcome and impact. Villagers meeting in Community Learning Centers are intrigued by comparisons to neighboring communities.

It would be an exaggeration to claim that data entry catalyzes any of those processes, but it is becoming less of an obstacle to them.

References

Ali, M., J. Park, et al. (2006). "Organizational aspects and implementation of data systems in large-scale epidemiological studies in less developed countries." BMC Public Health 6: Article 86.

Barchard, K. and L. Pace (2008). "Meeting the challenge of high quality data entry: a free double-entry system." International Journal of Services and Standards 4(4): 359-376.

Bassi, F. and L. Fabbris (1997). Estimators of Nonsampling Errors in Interview – Reinterview Supervised Surveys with Interpenetrated Assignments Survey Measurement and Process Quality L. Lyberg et. al. New York, John Wiley and Sons, Inc.: 733-751.

Burnett, M., C. Cook, et al. (2002). End-user software engineering with assertions [Technical Report 02-60-05]. Portland, Oregon State University: 103.

CDC (2005). Epi Info™ Community Health Assessment Tutorial [version 2.0]. Atlanta, Centers for Disease Control, Epidemiology Program Office.

Chambers, R. (2008). Revolutions in development inquiry. London and Sterling, VA, Earthscan.

Cork, D., M. Cohen, et al. (2003). Survey automation: report and workshop proceedings. Washington DC, National Academies Press.

Couper, M. (2005). "Technology trends in survey data collection." Social Science Computer Review 23(4): 486-501.

de Silva, N. and N. Gunetilleke (2008). "On trying to be Q-Squared: Merging methods for a technical minded client." International Journal of Multiple Research Approaches 2(2): 252-265.

FIVDB (2008). Jonoshilon. Popular Education. An Education Program Proposal [September 2008 – August 2013. Submitted to: Canadian International Development Agency and Embassy of the Kingdom of the Netherlands (EKN)]. Dhaka and Sylhet, Friends In Village Development Bangladesh.

Glewwe, P. and H. Dang (2008). "The impact of decentralized data entry on the quality of household survey data in developing countries: Evidence from a randomized experiment in Vietnam." The World Bank Economic Review 22(1): 165.

Groves, R. M., F. J. Fowler, et al. (2004). Survey methodology. Hoboken, NJ, J. Wiley.

Jann, B. (2005). "Tabulation of multiple responses." Stata Journal 5(1): 92-122.

Kim, W., B. Choi, et al. (2003). "A taxonomy of dirty data." Data Mining and Knowledge Discovery 7(1): 81-99.

McRitchie, D. (2008). "Event Macros, Worksheet Events and Workbook Events." Retrieved 1 March 2010, from http://www.mvps.org/dmcritchie/excel/event.htm#ws_activate.

Nicholls, W., R. Baker, et al. (1997). The effect of new data collection technologies on survey data quality. Survey measurement and process quality. L. Lyberg and e. al. New York, John Wiley and Sons: 221-248.

OzGrid. (Undated). "Dynamic Named Ranges." Retrieved 4 July 2010, from http://www.ozgrid.com/Excel/DynamicRanges.htm.

Pearson, C. (2010). "Events And Event Procedures In VBA." Retrieved 5 August 2010, from http://www.cpearson.com/excel/Events.aspx.

Rosenbloom, S., R. Miller, et al. (2006). "Interface terminologies: facilitating direct entry of clinical data into electronic health record systems." Journal of the American Medical Informatics Association 13(3): 277-288.

Sana, M. and A. A. Weinreb (2008). "Insiders, Outsiders, and the Editing of Inconsistent Survey Data." Sociological Methods & Research 36(4): 515-541.

Tomlinson, M., W. Solomon, et al. (2009). "The use of mobile phones as a data collection tool: A report from a household survey in South Africa." BMC Medical Informatics and Decision Making 9(1): 51.

U.S. Census Bureau (2009). CSPro. User's Guide, Version 4.0. Washington DC, U.S. Census Bureau, Population Division, International Programs Center.

Sample macro code

This code is attached to the worksheet "HHlevelData". The code attached to the second data sheet in the template workbook, "HHmembersData", is structurally similar and is not printed here.

```
Private Sub Worksheet_Change(ByVal Target As Range)

'Macro, first drafted in April 2010, revised 16 August 2010
'Sheet-specific macro to translate questionnaire codes into fully spelled-out response categories
'
'In addition, enters error messages into cells if:
'incorrect non-numeric or negative values
'illegal categories or entry codes

'Also moves active cell to the right if data entry correct and down to the next record when done
'-------------------------------------------------------------------------------------------

'DATA TABLE FOR ONE VILLAGE ONLY OR FOR SEVERAL:

Dim OneVillageOnly As Boolean

'If true, HHlevelData table meant to hold data only for one village. This is the default.
'If false, uniqueness of household numbers not enforced.
'Not checking uniqueness of household numbers opens the door to indexing errors.

OneVillageOnly = True
'-------------------------------------------------------------------------------------------

'MACRO EXITS IMMEDIATELY IF:

'Do nothing if more than one cell is changed or content deleted:
    If Target.Cells.Count > 1 Or IsEmpty(Target) Then Exit Sub
```

```
'Do nothing if a field name in the top row is modified:
    If Target.Row = 1 Then Exit Sub
'-------------------------------------------------------------------------------------------

'IF EXECUTION CONTINUES, THESE ENTITIES ARE USED:

'Local range names:
 DistColHead = Range("DistrictColHead").Column 'Cell in row 1 with field name District in this worksheet
 SnoColHead = Range("SerialColHead").Column 'Cell in row 1 with field name SerialNo in this worksheet
 HhhColHead = Range("HHheadColHead").Column 'Cell in row 1 with field name HHhead in this worksheet

'Field name (=formula in top row) in column where a cell value is being changed
'Its function is to refer to fields ([here:] = variables)
'not by column numbers (which may change),but by variable names:
 t = Cells(1, Target.Column).FormulaR1C1
'-------------------------------------------------------------------------------------------

'Delays updating screen until reinstated before exit or at the end of the macro
 Application.ScreenUpdating = False
'-------------------------------------------------------------------------------------------

'TEMPORARY WORKSHEET TO TRANSIT LOOKUP VALUES:

Set TempSheet = ActiveWorkbook.Worksheets.Add
    TempSheet.Name = "TempForTransit"

'Name a transit cell in it, which is later needed to pass lookup values:
ActiveWorkbook.Names.Add Name:="TransferCell", RefersToR1C1:="=TempForTransit!R1C1"

'Transfer the new target cell formula:
Sheets("TempForTransit").Range("A1").FormulaR1C1 = Target.FormulaR1C1
'Note: This seemingly obvious choice does not work:
'Range("TransferCell").FormulaR1C1 = Target.FormulaR1C1
```

```
'probably because the procedure is private

Worksheets("HHlevelData").Activate
'-------------------------------------------------------------------------------------------

'ERROR PREVENTION:
On Error Resume Next 'Stop any possible runtime errors and halting code
Application.EnableEvents = False 'Prevents endless loops due to multiple action in the target

'-------------------------------------------------------------------------------------------
'START PROCESS BY ENTERING SERIAL NUMBER'
If t = "SerialNo" Then

    'Entry error signal if non-numeric entry:

    If Not IsNumeric(Target) Then
        Target.FormulaR1C1 = "Use number!"
        Target.Interior.Color = 255
        Cells(Target.Row, Target.Column).Activate
    Else

    'If indeed numeric, then:
        'Entry error signal if same serial number as in cells above again used:

        d = 0
        With Range(Cells(2, Target.Column), Cells(Target.Row - 1, Target.Column))
            Set c = .Find(Target.Value, LookIn:=xlValues)
            If Not c Is Nothing Then
             firstAddress = c.Address
                Do
                    d = d + 1
                    Set c = .FindNext(c)
                Loop While Not c Is Nothing And c.Address <> firstAddress
            End If
```

```
        End With

                If d > 0 And Target.Row > 2 Then
                    Target.FormulaR1C1 = "Same used again"
                    Target.Interior.Color = 255
                    Cells(Target.Row, Target.Column).Activate

                Else 'If no entry error
                    Cells(Target.Row, Target.Column + 1).Activate
                    'Activates next field to the right

                End If
    End If

    'Event is over. Do necessary activities before ending the macro:
    GoTo Line1000
    '[See code at the bottom of this sheet!]

Else
End If

'END CODE FOR THIS VARIABLE
'Return to main text.
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'ENTERING HOUSEHOLD NUMBER
'Make sure field name for household number is spelt correctly, as Hhno, not HHno:
If t = "Hhno" Then

    'Entry error signal if non-numeric entry:

    If Not IsNumeric(Target) Then
        Target.FormulaR1C1 = "Use number!"
```

```
    Target.Interior.Color = 255
    Cells(Target.Row, Target.Column).Activate
Else
    'If indeed it is numeric, then, depending on restriction to one village:
    If OneVillageOnly = True Then
        'Entry error signal if same serial number as in cells above again used:

        d = 0
        With Range(Cells(2, Target.Column), Cells(Target.Row - 1, Target.Column))
            Set c = .Find(Target.Value, LookIn:=xlValues)
            If Not c Is Nothing Then
                firstAddress = c.Address
                Do
                    d = d + 1
                    Set c = .FindNext(c)
                Loop While Not c Is Nothing And c.Address <> firstAddress
            End If
        End With

            If d > 0 And Target.Row > 2 Then
                Target.FormulaR1C1 = "Same used again"
                Target.Interior.Color = 255
                Cells(Target.Row, Target.Column).Activate

            Else 'If no entry error, move to field "Household head"
                Cells(Target.Row, HhhColHead).Activate
            'This assumes that the geographical information will be entered only once
            '(in the first record), and then, since this stays within the same village,
            'be copied downward identically for all households


            'Scroll to active cell, but make sure previous record remain visible:
            Application.Goto Reference:=ActiveCell.Offset(RowOffset:=-1), scroll:=True
            Cells(ActiveCell.Row + 1, ActiveCell.Column).Activate
```

```
                End If
            Else
            'If not restricted to one village:
            Cells(Target.Row, HhhColHead).Activate
            'The danger is in copying the geographical information downwards, with potentially
            'incorrect village information. This process, however, remains manual.

            Application.Goto Reference:=ActiveCell.Offset(RowOffset:=-1), scroll:=True
            Cells(ActiveCell.Row + 1, ActiveCell.Column).Activate

            End If
        End If

        'Event is over. Do necessary activities before ending the procedure:
        GoTo Line1000
        'See end of sheet code

Else
End If

'END CODE FOR THIS VARIABLE
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'When entering DISTRICT information:

If t = "District" Then

            'Transferring the prompt to the vlookup formula, which then returns the full district name:
            Target.FormulaR1C1 = "=vlookup(TransferCell, District, 2, false)"

            Target.Copy 'These two lines replace the formula with its value
            Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
```

```
        Application.CutCopyMode = False 'Removes the blinking border from the copied cell

            'Entry error signal:
            If Target.FormulaR1C1 = "#N/A" Then
                Target.FormulaR1C1 = "Entry error"
                Target.Interior.Color = 255
                Cells(TargetRow, TargetCol).Activate

            Else 'If no entry error
                TargetRow = Target.Row
                TargetCol = Target.Column
                Cells(TargetRow, TargetCol + 1).Activate

            End If

        GoTo Line1000

Else
End If

'END CODE FOR THIS VARIABLE
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'When entering UPAZILA information:

If t = "Upazila" Then

        Target.FormulaR1C1 = "=vlookup(TransferCell, Upazila, 2, false)"
        Target.Copy 'These two lines replace the formula with its value
        Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
        Application.CutCopyMode = False 'Removes the blinking border from the copied cell

        ' Entry error signal:
```

```
        If Target.FormulaR1C1 = "#N/A" Then
            Target.FormulaR1C1 = "Entry error"
            Target.Interior.Color = 255
            Cells(TargetRow, TargetCol).Activate

        Else 'If no entry error
            TargetRow = Target.Row
            TargetCol = Target.Column
            Cells(TargetRow, TargetCol + 1).Activate

        End If

    GoTo Line1000

Else
End If

'END CODE FOR THIS VARIABLE
'-----------------------------------------------------------------------------------------------

'-----------------------------------------------------------------------------------------------
'When entering UNION information:

If t = "Union" Then

    Target.FormulaR1C1 = "=vlookup(TransferCell, Union, 2, false)"
    Target.Copy 'These two lines replace the formula with its value
    Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
    Application.CutCopyMode = False 'Removes the blinking border from the copied cell

        'Entry error signal:
        If Target.FormulaR1C1 = "#N/A" Then
            Target.FormulaR1C1 = "Entry error"
            Target.Interior.Color = 255
```

```
            Cells(TargetRow, TargetCol).Activate

        Else 'If no entry error
            TargetRow = Target.Row
            TargetCol = Target.Column
            Cells(TargetRow, TargetCol + 1).Activate

        End If

    GoTo Line1000

Else
End If

'END CODE FOR THIS VARIABLE
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'When entering VILLAGE information:

If t = "Village" Then

    Target.FormulaR1C1 = "=vlookup(TransferCell, Village, 2, false)"
    Target.Copy 'These two lines replace the formula with its value
    Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
    Application.CutCopyMode = False 'Removes the blinking border from the copied cell

    ' Entry error signal:
        If Target.FormulaR1C1 = "#N/A" Then
            Target.FormulaR1C1 = "Entry error"
            Target.Interior.Color = 255
            Cells(TargetRow, TargetCol).Activate

        Else 'If no entry error
```

```
            TargetRow = Target.Row
            TargetCol = Target.Column
            Cells(TargetRow, TargetCol + 1).Activate
        End If

    GoTo Line1000

Else
End If

'END CODE FOR THIS VARIABLE
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'For the rest of this data entry sheet, several variables are grouped together,
'as far as the macro is concerned. The variables in a block are treated the same,
'to abbreviate the code. If the macro finds that the target cell is in such a variable,
'it executes the block instructions (e.g., if it is a numeric variable: requiring numeric input),
'and then does some house-keeping and exits the macro.
'Else it moves to the next block.
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'When entering INCOME SOURCE information:

'This particular block is called "FieldsIncomeSource" [see definitions of names]

'Example of skipping if no data to enter and scolling to next block [here: Loan History]
'Type "z" if no information to be entered and you wish to skip to
'The vlookup function will then return zero. The ElseIf-clause below removes it.
'It activates the cell in the column skipped to, and scrolls to it, for convenience
'leaving one record above visible.

Set isect = Application.Intersect(Range("FieldsIncomeSource"), Cells(1, Target.Column))
```

```
'Note: The choice of this range name as isect is arbitrary.

If Not isect Is Nothing Then

    Target.FormulaR1C1 = "=vlookup(TransferCell, IncomeSource, 2, false)"
    Target.Copy 'These two lines replace the formula with its value
    Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
    Application.CutCopyMode = False 'Removes the blinking border from the copied cell

    ' Entry error signal:
    If Target.FormulaR1C1 = "#N/A" Then
        Target.FormulaR1C1 = "Entry error"
        Target.Interior.Color = 255
        Cells(Target.Row, Target.Column).Activate

    ElseIf Target.FormulaR1C1 = "0" Then
        Target.FormulaR1C1 = ""
        Cells(Target.Row, LHistColHead).Activate

        'Scroll to active cell, but make sure previous record remain visible:
         Application.Goto Reference:=ActiveCell.Offset(RowOffset:=-1), scroll:=True
         Cells(ActiveCell.Row + 1, ActiveCell.Column).Activate
        'Return to section on skipping and scrolling when no data

    Else 'If no entry error
        Cells(Target.Row, Target.Column + 1).Activate

    End If
    Set isect = Nothing
    GoTo Line1000

Else
End If
```

```
'END CODE FOR THESE VARIABLES
'Return to main text.
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'When entering data on income AMOUNTS (Income1, etc.):
'This particular block is called "IncomeAmounts" [see definitions of names]
'No values prescribed, but must be numeric positive

Set zsect = Application.Intersect(Range("IncomeAmounts"), Cells(1, Target.Column))
'Note: The choice of this range name as zsect is arbitrary.

If Not zsect Is Nothing Then

    If Not IsNumeric(Target) Then
        Target.FormulaR1C1 = "Use number!"
        Target.Interior.Color = 255
        Cells(Target.Row, Target.Column).Activate

    ElseIf Target.Value < 0 Then
    Target.FormulaR1C1 = "Must be 0 or positive"
        Target.Interior.Color = 255
        Cells(Target.Row, Target.Column).Activate

    Else
    Cells(Target.Row, Target.Column + 1).Activate

    End If

    Set zsect = Nothing
    GoTo Line1000

Else
End If
```

```
'END CODE FOR BLOCK OF VARIABLES
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'When entering past LOAN HISTORY or ANY CURRENT LOAN information:

If t = "LoanHistory" Or t = "LoanCurrentAny" Then

    Target.FormulaR1C1 = "=vlookup(TransferCell, yesno, 2, false)"

    Target.Copy 'These two lines replace the formula with its value
    Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
    Application.CutCopyMode = False 'Removes the blinking border from the copied cell

    ' Entry error signal:
        If Target.FormulaR1C1 = "#N/A" Then
            Target.FormulaR1C1 = "Entry error"
            Target.Interior.Color = 255
            Cells(Target.Row, Target.Column).Activate

        ElseIf Target.FormulaR1C1 = "No" And t = "LoanHistory" Then
            Cells(Target.Row, Target.Column + 5).Activate

        ElseIf Target.FormulaR1C1 = "No" And t = "LoanCurrentAny" Then
            Cells(Target.Row, Target.Column + 9).Activate

        Else 'If no entry error
            Cells(Target.Row, Target.Column + 1).Activate

        End If

        GoTo Line1000
```

```
Else
End If


'END CODE FOR THESE TWO VARIABLES
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'When entering ORGANIZATION and LOAN SOURCE information:
'[This block, although with two variable groups, is handled with
'If Instr(..).. Or .. , and with one lookup table only:]

If InStr(1, t, "Organisation", 1) > 0 Or InStr(1, t, "LoanSource", 1) > 0 Then

  Target.FormulaR1C1 = "=vlookup(TransferCell, LoanSources, 2, false)"
  Target.Copy 'These two lines replace the formula with its value
  Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
  Application.CutCopyMode = False 'Removes the blinking border from the copied cell

  ' Entry error signal:
     If Target.FormulaR1C1 = "#N/A" Then
         Target.FormulaR1C1 = "Entry error"
         Target.Interior.Color = 255
         Cells(TargetRow, TargetCol).Activate

     Else 'If no entry error
         Cells(Target.Row, Target.Column + 1).Activate
     End If

     GoTo Line1000

Else
End If
```

```
'END CODE FOR THESE TWO VARIABLE SETS
'Return to main text.
'-------------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------------
'When entering LOAN BALANCE information:

If InStr(1, t, "LoanBalance", 1) > 0 Then

     If Not IsNumeric(Target) Then
         Target.FormulaR1C1 = "Use number!"
         Target.Interior.Color = 255
         Cells(Target.Row, Target.Column).Activate

     ElseIf Target.Value < 0 Then
     Target.FormulaR1C1 = "Must be 0 or positive"
         Target.Interior.Color = 255
         Cells(Target.Row, Target.Column).Activate

     Else
     Cells(Target.Row, Target.Column + 1).Activate
     End If

     GoTo Line1000

Else
End If

'END CODE FOR THIS VARIABLE
'-------------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------------
'When entering the INTERVIEWER name:
```

```
If t = "Interviewer" Then

    Target.FormulaR1C1 = "=vlookup(TransferCell, Interviewers, 2, false)"
    Target.Copy 'These two lines replace the formula with its value
    Target.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False
    Application.CutCopyMode = False 'Removes the blinking border from the copied cell

    ' Entry error signal:
        If Target.FormulaR1C1 = "#N/A" Then
            Target.FormulaR1C1 = "Name must first be created in interviewer list in sheet Hlists."
            Target.Interior.Color = 255
            Cells(TargetRow, TargetCol).Activate

        Else 'If no entry error
            Cells(Target.Row, Target.Column + 1).Activate
        End If
        GoTo Line1000

Else
End If

'END CODE FOR THIS VARIABLE
'-----------------------------------------------------------------------------------------

'-----------------------------------------------------------------------------------------
'When entering the INTERVIEW DATE:

 If t = "DateInterview" Then

      If Not IsDate(Target) Then
          Target.FormulaR1C1 = "The interview date is required!"
          Target.Interior.Color = 255
          Cells(Target.Row, Target.Column).Activate
```

```
      ElseIf Target.Value < 40179 Then
      Target.FormulaR1C1 = "Must be 1st Jan 2010 or later"
          Target.Interior.Color = 255
          Cells(Target.Row, Target.Column).Activate

      Else
      'Opens new record with consecutive serial number:
      Cells(Target.Row + 1, SnoColHead).Value = Cells(Target.Row, SnoColHead).Value + 1
      Cells(Target.Row + 1, SnoColHead + 1).Activate

         'Scroll to active cell, but make sure previous record remain visible:
         Application.Goto Reference:=ActiveCell.Offset(RowOffset:=-1), scroll:=True
         Cells(ActiveCell.Row + 1, ActiveCell.Column).Activate
      End If

      GoTo Line1000

 Else
 End If

'END CODE FOR THIS VARIABLE
'-------------------------------------------------------------------------------------------

'-------------------------------------------------------------------------------------------
'END OF MACRO:

Line1000:
'House-keeping before ending the macro:

    Application.DisplayAlerts = False 'To avoid a prompt when deleting the temporary sheet
    ActiveWorkbook.Names("TransferCell").Delete
    Sheets("TempForTransit").Delete
    Application.EnableEvents = True 'Turn events back on
```

```
    On Error GoTo 0 'Allow run time errors again
    Application.ScreenUpdating = True
    Application.DisplayAlerts = True

End Sub

'-------------------------------------------------------------------------------------------
```

[This page deliberately left blank]

Revisions and known bugs

The FIVDB PPR Unit started to use this template for the computer entry of data collected during the second round of a household baseline survey in spring 2010. Increasingly, starting in August, entry has been contracted out to a village-based organization, the Khagamura ICT, which is using the same template (see page 42).

Drawing on the combined experience of the FIVDB monitoring staff as well as of the Khagamura ICT, a first version of this study was published in November 2010, together with a didactic Excel workbook template.

In February 2011, two of us (Benini, W.S. Chowdhury) visited Khagamura. Also, the personal experience that the monitoring associates had made was discussed in a workshop in the same month. The associates listed several problems, some of which are noted in the section "Experience working with the template" (page 38 sqq.). From those discussions, the current versions of the study and the workbook template have resulted. The revised template fixes a problem with a rarely incurred type of incorrectly flagged error in supervised categorical variables (described on page 25). Other suggested modifications have not been implemented, either because they hold no general interest beyond this specific survey, or because convenient work-arounds are already available.

At present (March 2011), there are no known bugs in the template. Further testing and adapting to the needs of other surveys will likely reveal hitherto unknown malfunctions. Regardless, there is ample scope for improvement. In particular, as noted on page 34, the use of a hidden temporary worksheet used in the transfer of categorical values strikes us as extremely inelegant. Readers are encouraged to propose better solutions. New ways to improve the code documentation will also become manifest once the template is investigated for other survey contexts.

Author information

Aldo Benini is an independent researcher and consultant living in Washington DC, USA. He has known FIVDB since 1980 and occasionally consults on monitoring matters.

Hasan Ahmed Chowdhury is the Quality Assurance Coordinator of FIVDB's Functional Literacy Programme. He has been with the organization since 1990.

Wasima Samad Chowdhury is the Coordinator of Policy Planning and Research (PPR) Unit. She has been with FIVDB since 1996.

Saiful Hasan is a Research Associate, Abu Saeem Arif and Md. Yasin Mazumder are Monitoring Associates, in the PPR Unit. They have been with FIVDB since 2009.

Contacts:

Friends in Village Development Bangladesh (FIVDB)
Khadimnagar, Sylhet
P.O Box 70, Sylhet 3100
Bangladesh

Tel: +88 (0821) 2870466, 2871221, 2870020
Fax: +88 (0821) 2870021.

Website: http://www.fivdb.net, www.aldo-benini.org

E-mail: wasima [at] fivdb.net, abenini [at] starpower.net.

FIVDB