Dynamically Created Answer Categories in Blaise Instruments

Carol Oppenheim, Battelle Memorial Institute

The use of enumerated answer types in Blaise data models provides great control over the data collected during the course of a study. These enumerated types have specific meaning for investigators and are an aid to analysis. One disadvantage of enumerated types is that they require a "one size fits all" approach. There can be circumstances within interviews that would make it desirable to have individualized responses instead of the same exact choices for all participants.

A way to take advantage of the best features of enumerated answer types and still present meaningful answer choices to respondents is to create answer categories dynamically during the course of each interview. Offering personalized answer choices that have specific meaning for each respondent is an excellent way to enhance interviews. For analysis, every person's data will have responses stored in the same format but for administering the interview, each person will be given different options. By dynamically filling the response categories, it is possible to store a response with specific meaning for the researcher and present as choices responses with specific meaning for the respondents.

There are three steps in setting up an enumerated answer type in this way. First you must establish some identifiers that will be used as text fills. The fills can be set up as fields or as locals. Using fields for the fills allows the actual text that was presented to the respondent to be stored in the data set. If there is no need to store the actual text, locals work equally well.

The second step is to create answer types that will use the fills as answer text. Each response will have a name and text. The text will consist of double quotes surrounding a caret and the fill identifier, as in "^DayOne". The answer type cannot be part of an included answer type file, but must be defined in the module(s) in which it will be used.

Finally there must be program logic to determine the content of the text fills. There are numerous ways to determine the text for the fills. Some examples are the interview date, whether the respondent is a case or control, or responses provided previously by the respondent.

At Battelle, we have successfully used this technique on a number of studies. It has been useful for questions with varying degrees of complexity. For some questions we knew that every respondent would be offered the same number of choices and that only the text content needed to be determined individually. In other cases, we needed to determine how many response choices each person needed, as well as the content of each choice. There were other instances in which the answer categories needed to be matched with particular ordinal values, a requirement that usually involved being able to have discontinuous numbering of the responses presented to any one respondent.

In a study of depression among adolescents, there is a series of questions for which the researcher wanted answers with the meanings "The current month", "One month ago", "Two months ago", "Three months ago", "Four months ago", and "Five months ago." Although the researchers were interested in the relative time periods, the respondents would be likely to answer with actual months. In order to properly select answers from these categories, the interviewer would need to use the respondent's answers to calculate back from the current month, an error-prone process that would slow down the interview. Without dynamically created answer categories, the answers for this question would be presented to interviewers and respondents in the following format:

0. The current month
1. One month ago
2. Two months ago
3. Three months ago
4. Four months ago
5. Five months ago

We were asked to program those questions so the respondent could choose actual months based on the month in which the interview takes place. So for an interview being administered today, the answer choices would be September, August, July, June, May, and April. The text for these answers is filled as soon as the interview is started, using the month of the interview date as a point of reference. Looking at the code needed to accomplish this, we first established the following fields as part of the main source code for the data model:

ThisMon : STRING[9]

LastMon : STRING[9]

Back2Mon : STRING[9]

Back3Mon : STRING[9]

Back4Mon : STRING[9]

Back5Mon : STRING[9]

In this case, we chose to use fields instead of locals for the items to be filled because there were questions in more than one block of the interview that relied on these responses. Rules for determining the text to go with these fields were also written in the source code for all months of the year:

```
  IF IntDt.MONTH = 1 THEN
       ThisMon := 'JANUARY'
       LastMon := 'DECEMBER'
       Back2Mon := 'NOVEMBER'
       Back3Mon := 'OCTOBER'
       Back4Mon := 'SEPTEMBER'
       Back5Mon := 'AUGUST'
…..
    ELSEIF IntDt.MONTH = 9 THEN
       ThisMon := 'SEPTEMBER'
       LastMon := 'AUGUST'
       Back2Mon := 'JULY'
       Back3Mon := 'JUNE'
       Back4Mon := 'MAY'
       Back5Mon := 'APRIL'
….
ENDIF
```

Within the interview, there were numerous questions that asked respondents in which of the last six months they participated in particular activities. For each of those questions, the answers that appeared on the screen were the fills created in the main source code. The respondents could choose the answers "September" and "July" and the interviewers could choose those exact responses from the options on the screen. The interviewers did not have to do any "translation" to know that July was two months ago. They selected "July" and it was stored as "two months ago" or, numerically, as code 2.

```
SET OF
(CURRENT (0) "^ThisMon",
Minus1 "^LastMon",
Minus2 "^Back2Mon",
Minus3 "^Back3Mon",
Minus4 "^Back4Mon",
Minus5 "^Back5Mon")
```

For respondents interviewed in September, this is the appearance of the enumerated type on the screen:

0. SEPTEMBER
1. AUGUST
2. JULY
3. JUNE
4. MAY
5. APRIL

Comparing this to the fixed answer categories, we can see that these responses are easier to work with, both for the interviewer and the respondent. They may even help with respondent recall since the respondents won't have to count up months in their heads to know what time period the questions refer to.

Sometimes we know that there is a maximum number of responses that we want to have available for all respondents but a smaller number for some individuals. In a study of services available to new mothers and their infants, one of the questions asked the mother when she came home from the hospital. Mothers were supposed to keep a diary of services used for up to 16 days beginning with the birth of their child. The baby's date of birth was already known before the start of the interview, so we knew that was the earliest date on which the mother and child could have come home and on which the diary could begin. We did not want to continue the diary questions past the interview date or past the 16th day of the child's life. In this study, the use of dynamically filled answer categories permitted us to display anywhere from one to 16 possible dates on which the mother could have come home. We filled the categories with precise dates, including the day of the week. The mother could reply with either an exact date or a day of the week and the interviewer would see both forms of the date on the screen. For those respondents whose answer list included fewer than 16 days, the unused days were not assigned any fill and they were not displayed on the screen. It is important to note that the unused days did still exist as answer choices, even though they had no text and were unseen. To prevent an interviewer from selecting an inappropriate response, we had to have logic that precluded their use. This was accomplished very easily with the following code that was repeated as needed for additional days.

Assuming the interview was taking place prior to the infant's tenth day of life, the code for day 10 would have looked like this:

```
IF Day10 = EMPTY THEN
        ComeHome <> Day_10
        "The date you have selected is after the current date."
```

ENDIF

The answer choices for a mother who gave birth on July 20 and was called on July 23 appeared as follows:

1. Friday, 7/20/2001
2. Saturday, 7/21/2001
3. Sunday, 7/22/2001
4. Monday, 7/23/2001

Without using the dynamically filled categories, it probably would have been necessary to have three separate fields in order to determine the date of homecoming. The first field would have been for interviewer use only (not to be read aloud) and would have been something like "Did R answer with a date or with a day of the week?" Depending on which answer the respondent provided, the program would then have to go either to a field for entering the day of the week or one for entering the actual date. Our dynamically filled answer categories eliminated the need for a lead question and displayed easily understandable dates instead of such answers as "Day of baby's birth" and "Third day after bay's birth."

In a study of unintentional injuries, we needed different response choices for each respondent. These choices were based on how the respondent completed a roster of children in the household and on who else was part of the household. The roster was completed in detail on up to 10 children under age 15, including their first names. In addition we also collected the number of children ages 15 – 18, but no other details on those children. In every instance there was an adult respondent. There could also be other adults in the household. In one section of the interview, respondents were asked if anyone in the household had been bitten by a dog. The answer categories needed to include first names of all children under age 15, one category for children 15 – 18 if there were any present in the household, one category for the respondent, and one for other adults, if any were present in the household. The enumerated answer categories had values from 1 to 13. Categories 1 through 10 were always reserved for children under age 15. These were identified in the data as CHILD1, CHILD2, CHILD3, etc.

(CHILD1 "^CH1",
CHILD2 "^CH2",
CHILD3 "^CH3",
CHILD4 "^CH4",
CHILD5 "^CH5",
CHILD6 "^CH6",
CHILD7 "^CH7",
CHILD8 "^CH8",
CHILD9 "^CH9",
CHILD10 "^CH10",
CHILDOVER14 "^CH11",
RESP "^RESPONDENT",
OTHERADULT "^OTHADULT)

Once the makeup of the household had been determined, the answer categories could be filled appropriately. In a household with 3 children under age 15, one child at least 15 years old, and a respondent, the answer categories would be:

1. Freddy
2. Sue
3. Alice

11. Child over age 14
12. Respondent

Categories 4-10 and 13 would not be needed so they wouldn't be filled or displayed. As in the example above, these categories continue to exist and we still needed to write logic to prevent anyone from selecting these invalid choices. When the question about dog bites was asked, we then cleared the text from those categories that were not selected by the respondent. The code is, again, very simple:

```
IF NOT(Child1 in DogBiteQuestion) THEN
        CH1 := EMPTY
ENDIF
```

The same code was repeated for every possible response. If two household members were bitten, we only want the interviewers to select from those two people for subsequent questions related to dog bites. The new answer list, to be used for the question "Of the people bitten, who sought medical care?" might look like this:

2. Sue
12. Respondent

All other household members and all unused slots for household members continue to exist as possible but invalid answers. The same logic used previously to keep invalid choices out of the data is still needed here.

An example of using text fills for responses in a case-control study is a study in which the cases were part of a special program. As part of the program, children of cases received intervention from program specialists who worked out of pediatricians' offices. When asking cases and controls who at the office provided services, we would want everyone to be offered certain choices, such as doctor, nurse, nurse-practitioner. Only cases should be offered "program specialist" as a choice. For controls, the answers would be:

1.Doctor
2. Nurse
3. Nurse-practitioner
5. Social worker

For cases the answers would be:

1.Doctor
2. Nurse
3. Nurse-practitioner
4. Program specialist
5. Social worker

The answer type looked like this:

```
THelper =
(Doctor,
Nurse,
Nurse_p "Nurse-practitioner",
Spec "^Code4",
SocWork "Social worker")
```

The correct set of answers was determined for this study by a digit of the case ID that identified the respondent as either a case or control. The logic used to set the contents of the answer text was:

```
IF SUBSTRING(CASEID, 3,1) = '1' THEN
        CODE4 = 'Program specialist'
ELSE
        CODE4 = ''
ENDIF
```

Another way to have different choices for cases and controls is to set this up as two different questions, one that was only used for cases, the other for controls. Each would have its own answer type. When converting the data file to a format in which it could be delivered to the client it would be necessary to assign the answers to the two questions to the same data location. By using the dynamically filled answer categories, we already had the data from the two types of interviews in the same location. This also made the program clearer than it would be with two different fields being used for the same portion of the interview.

These examples are just a few of the ways in which dynamically filled answer categories can be incorporated into Blaise datamodels. As we use this technique, we continue to identify more and more situations where it will help us produce clearer interviews and better data. Using dynamically filled answer categories does require some additional programming up front, but the advantages make the extra effort very worthwhile.