How Blaise 5 Improves Table Presentation

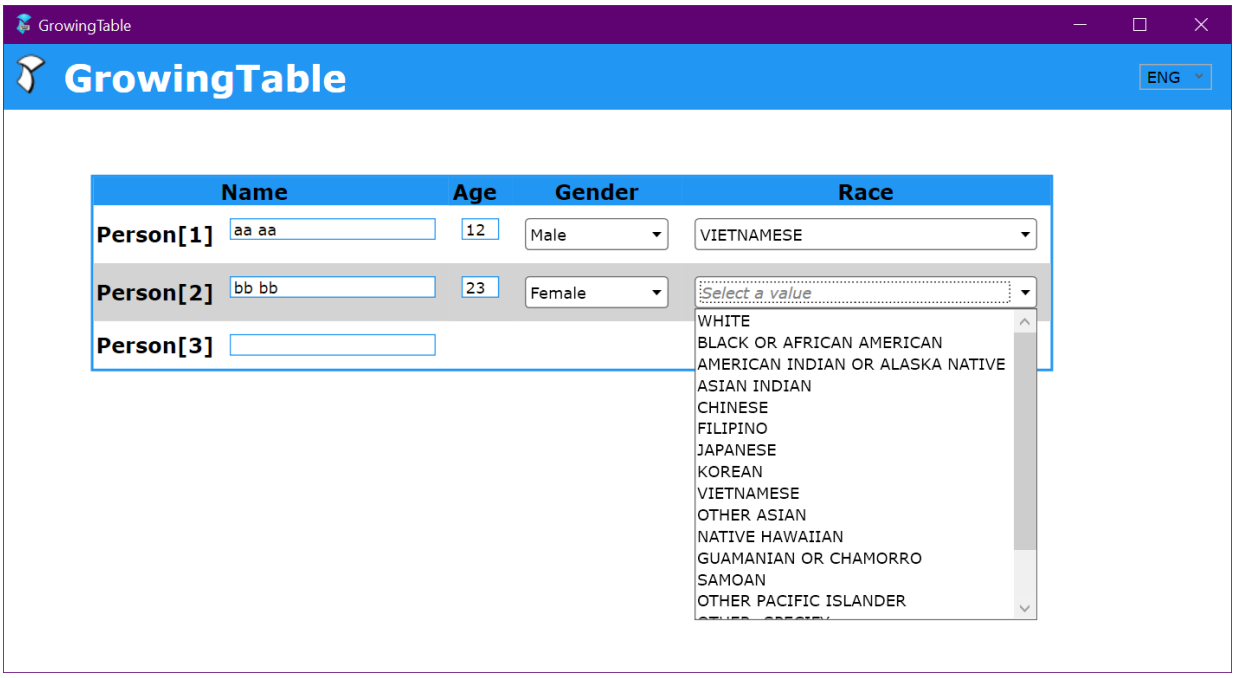
G J Boris Allan and Stéphane Ridoré, Westat

Presenter: Siu Chong Wan

As we look across versions of Blaise, the presentation of tables has continued to evolve. When we first encountered Blaise 3, one of our first questions concerned rosters and their implementation. The answer was Blaise tables, and we used tables successfully in both Blaise 3 and its successor, Blaise 4.
When Blaise 5 was introduced, we could use additional forms of tables, and one of the most important for our work is known as Option Tables. This paper discusses ways to work with tables in Blaise 5. We found that not only could we massage standard tables but also that we could extend our applications to various forms of option table by changing the nature of tables using the resource database (.blrd file).

1. The Basic Blaise Cross-Tabulation Table

The Blaise 5 samples have a variety of instruments that use tables, for example the (growing) roster-style table:



This is a growing table, but that is incidental. We have embellished this table by adding on an extra column, the one labeled “Race.” The race question is based on categories used in an earlier CAPI instrument in Blaise 4.8 and is used only as an example. Race questions can become complex (Blaise 4 was actually a SET OF), and so in the original interview, the CAPI interviewer had a good deal of extra information that was not read aloud to the respondent. Here is a version of the Blaise 4.8 CAPI translated to Blaise 5.13:

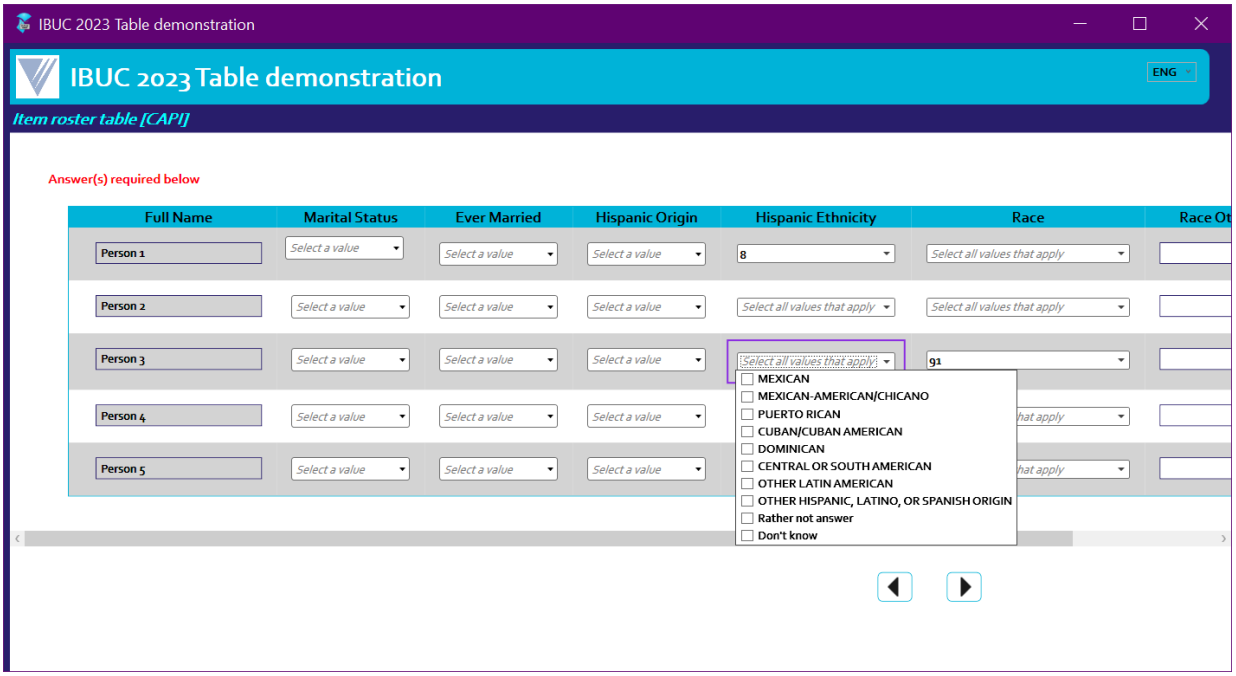
```
Race
"For this survey, Hispanic origins are not races. <newline> <newline>What is your
race? Please select one of the categories on this card. <newline><newline>ENTER ALL
```

```
THAT APPLY."
/ "Race"
: (TRWhite    (1)  "WHITE" "BLANCA",
   TRAfrAmer  (2)  "BLACK OR AFRICAN AMERICAN" "NEGRA O AFROAMERICANA",
   TRAmInAlk  (3)  "AMERICAN INDIAN OR ALASKA NATIVE" "INDIA AMERICANA
```

One drawback in the Blaise 5 CAPI version above, therefore, is the lack of interviewer instructions we could have with Blaise 4.

1.1 A Simple Roster Version

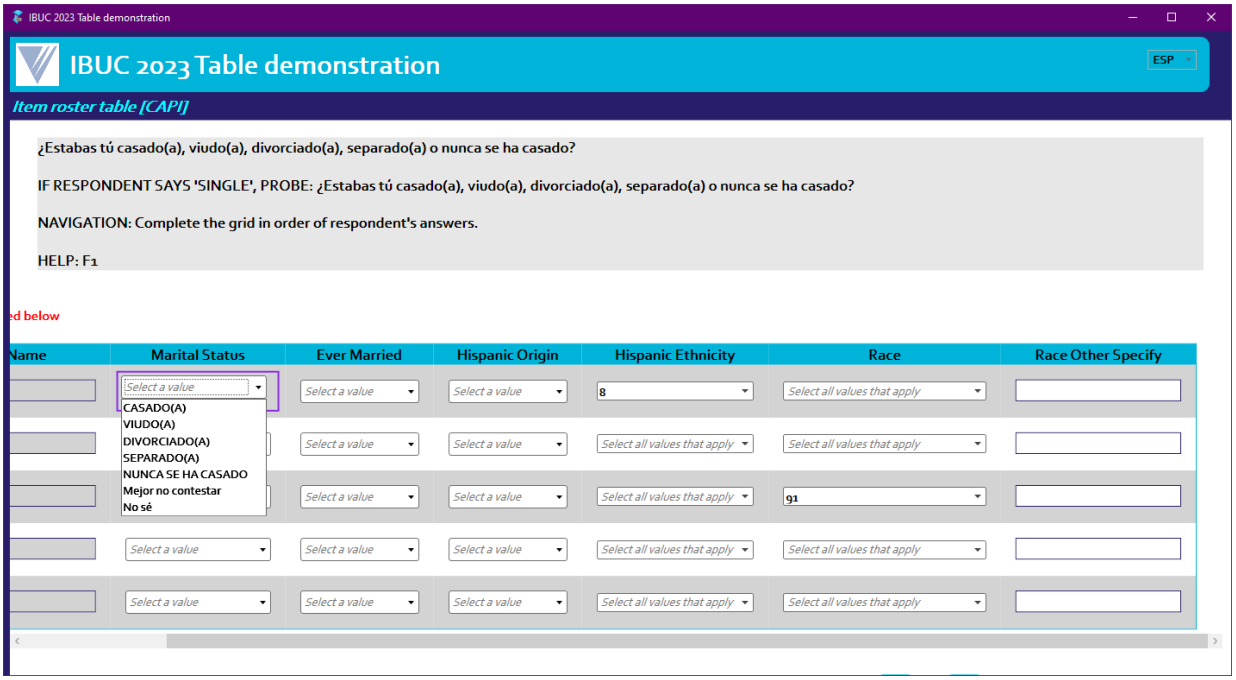
A more extensive example of a roster in Blaise 5.13, based on the style of rosters in Blaise 4.8, is:



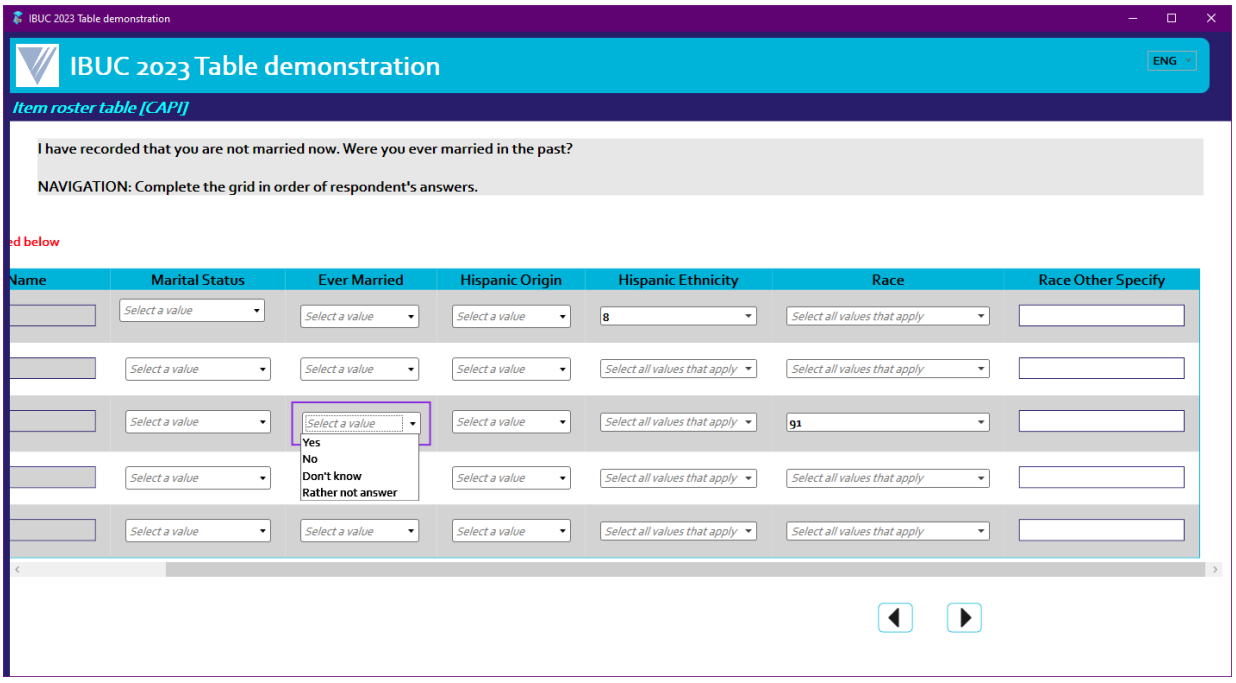
Note the horizontal scroll bar—very useful for wide tables—but there are no instructions for the list of types of Hispanic ethnicities.

1.2 The Simple Roster Version with an Extra Display

We can extend the roster with information for the interviewer and, in this case, a mixture of English and Spanish. The English parts of the question text (mainly in CAPS) are directions to the interviewer, and the Spanish text is to be read to the respondent. This is a fairly simple extension of the appropriate master page template:



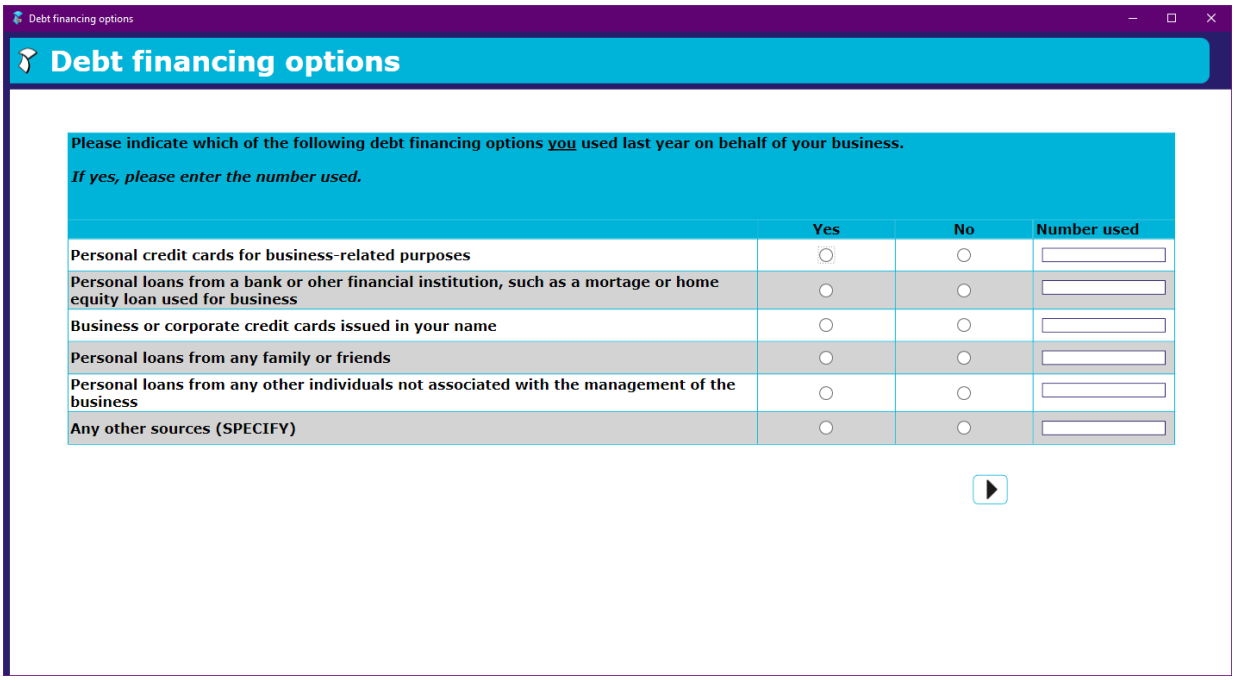
A version totally in English is:



Note the purple highlight to show which cell is being answered.

2. The Blaise Options Table

We have—for many studies—a particular need for a type of table that is not a roster-style table, that is, we need an options table. The Blaise 5 samples have an example that is a combination of the two table types (roster and options):



Each subject topic (say, credit cards for business) has two associated questions: the first is a Yes/No question (two options) and the second is a simple numerical question. Whereas with, say, the question about Hispanic Origin in the roster (Yes/No with perhaps DK/RF) we used a drop-down list, in this options table example, we showed the two options in two columns. Options tables are used most frequently when we have many stimuli/topics and we want to have the options in a more convenient presentation.

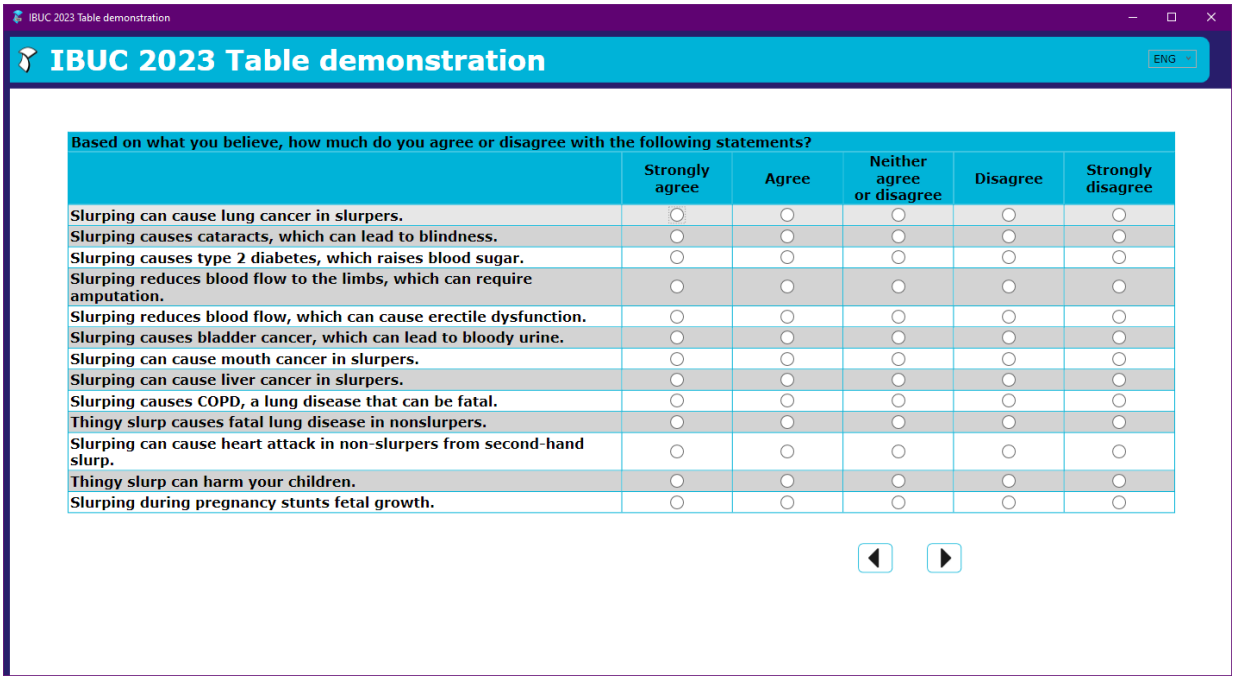
2.1 Initial Views

The rest of this paper investigates two types of options table (many versions are possible) and how we have extended functionality to improve understanding of what is required. Based on some of our methodologists' findings and some of our clients' requirements, we have extended functionality: all these changes are elaborations of basic templates extended to incorporate new features in the .blrd. The elaborations are dependent on these requirements.

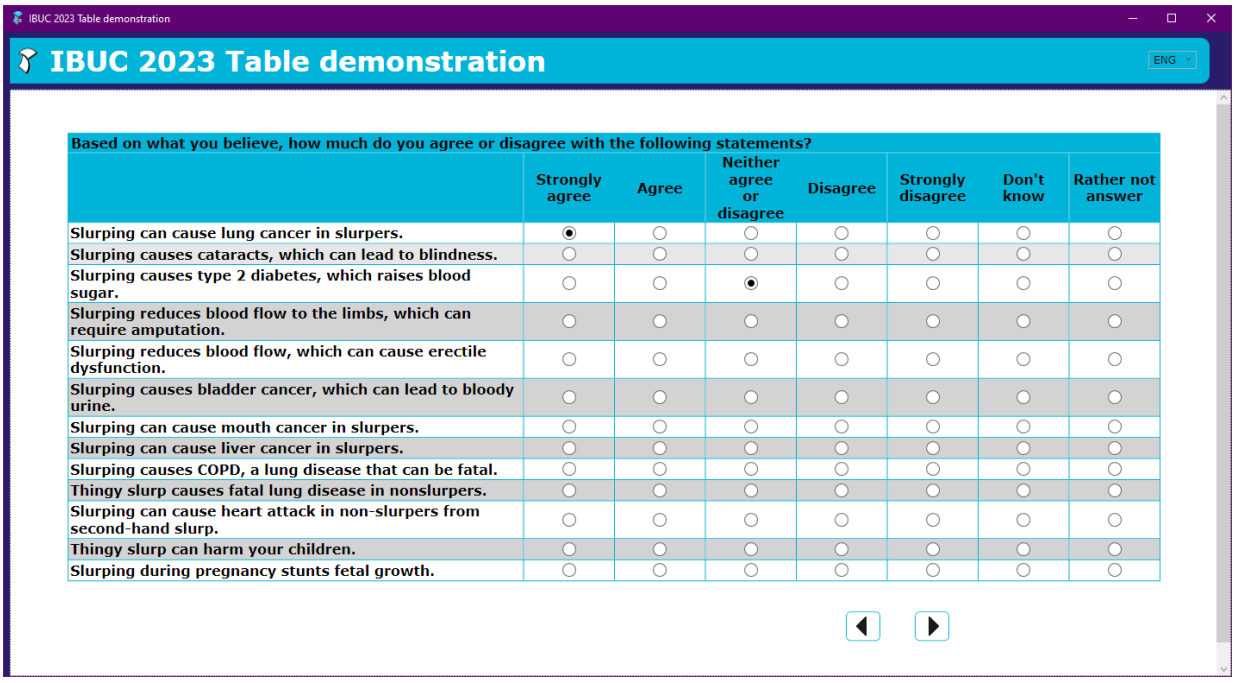
Whereas rosters are principally a CAPI approach, options tables have applicability for CASI, such as a questionnaire on a browser.

2.1.1 The Standard Options Table

This is a standard display.

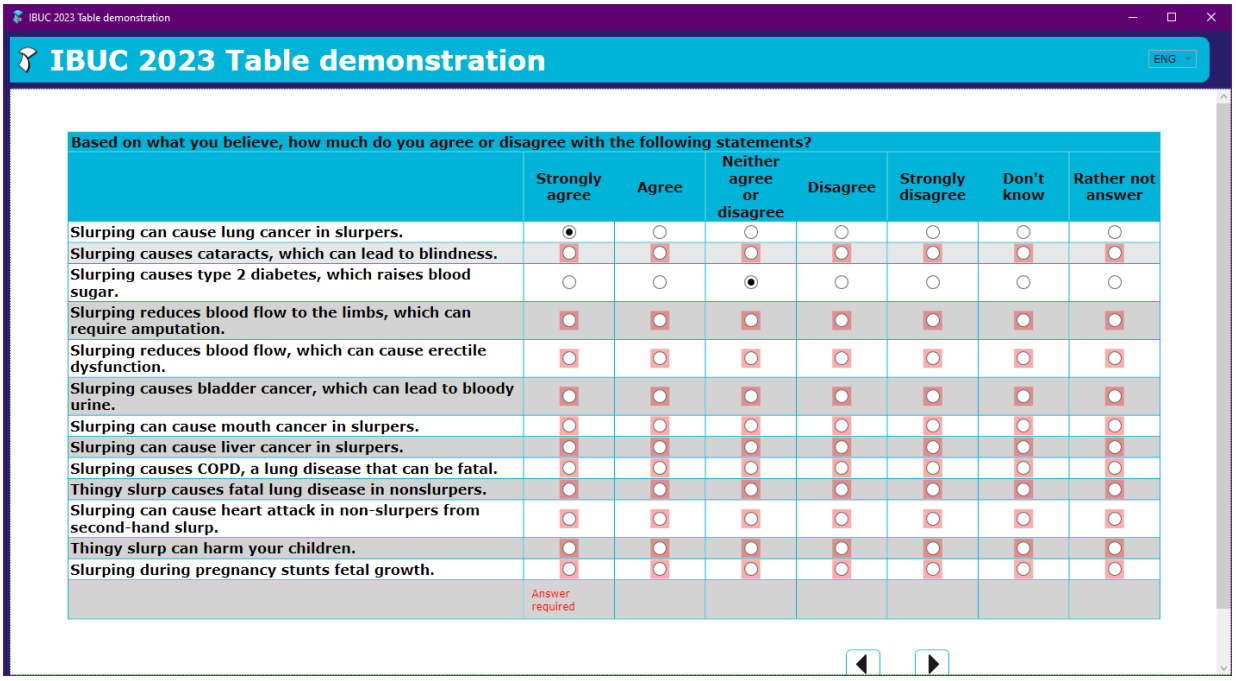


If you click/touch/select some buttons on the above table, you can add data:

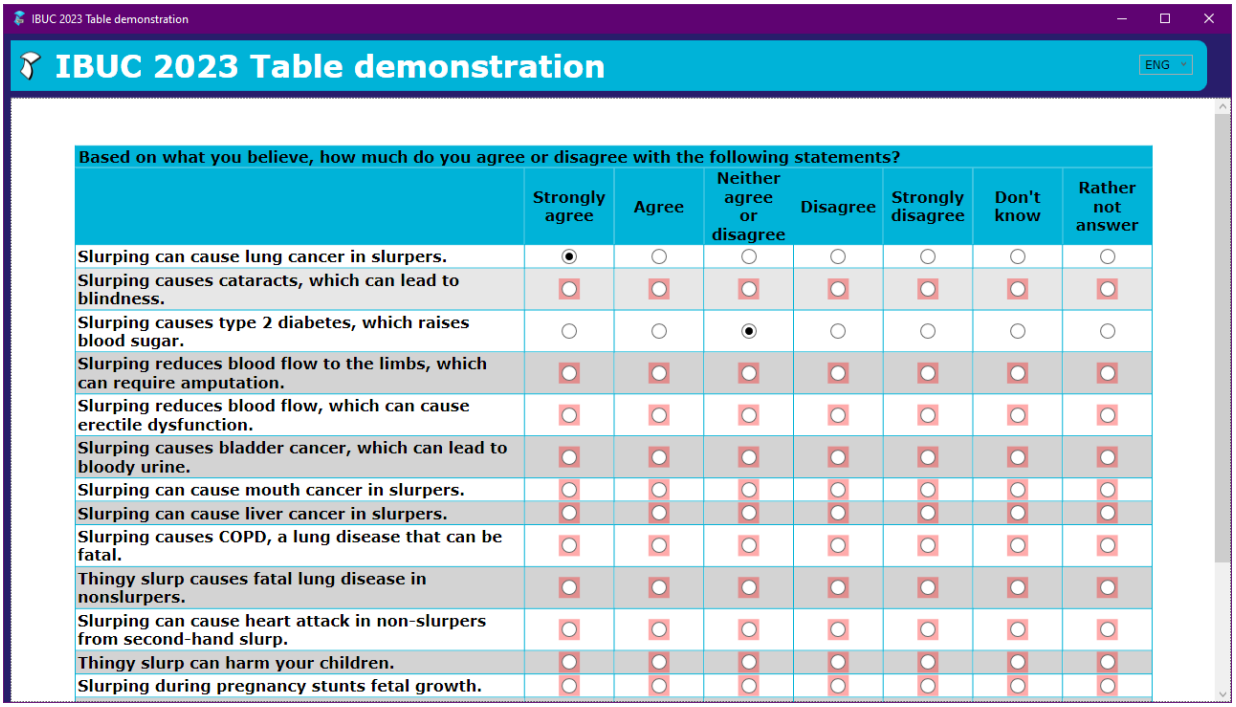


You selected “Strongly agree” for lung cancer, skipped a row, and then selected “Disagree” for blood sugar. As soon as you leave an empty row and select the next row, the Blaise 5 normal nonresponse behavior (defined in Settings) kicks in. “Normal behavior” is to not show a nonresponse button until the next field on route receives focus.

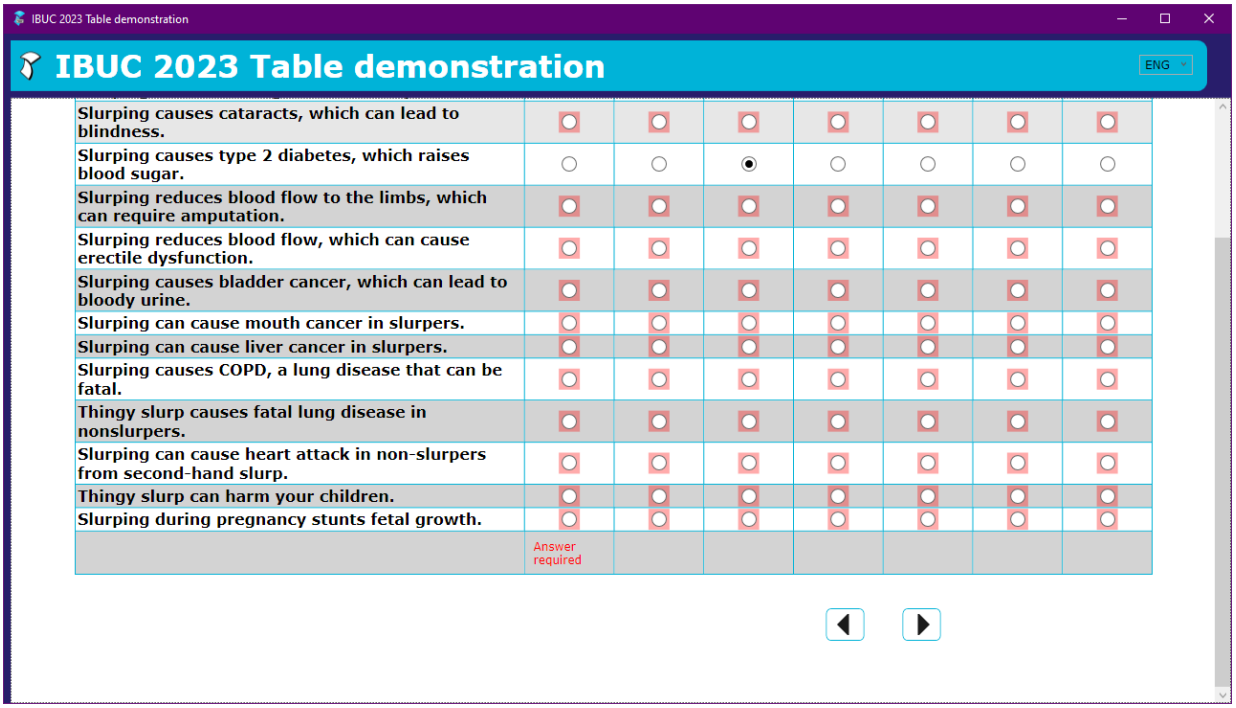
If you select the next-page button, then you find:



If we had any more rows, we might find that the back and next buttons would drop off the bottom of the view and disappear. There are studies for which the number of rows far exceeds the number we have above and, to show disappearing rows and buttons, we make the above window smaller:

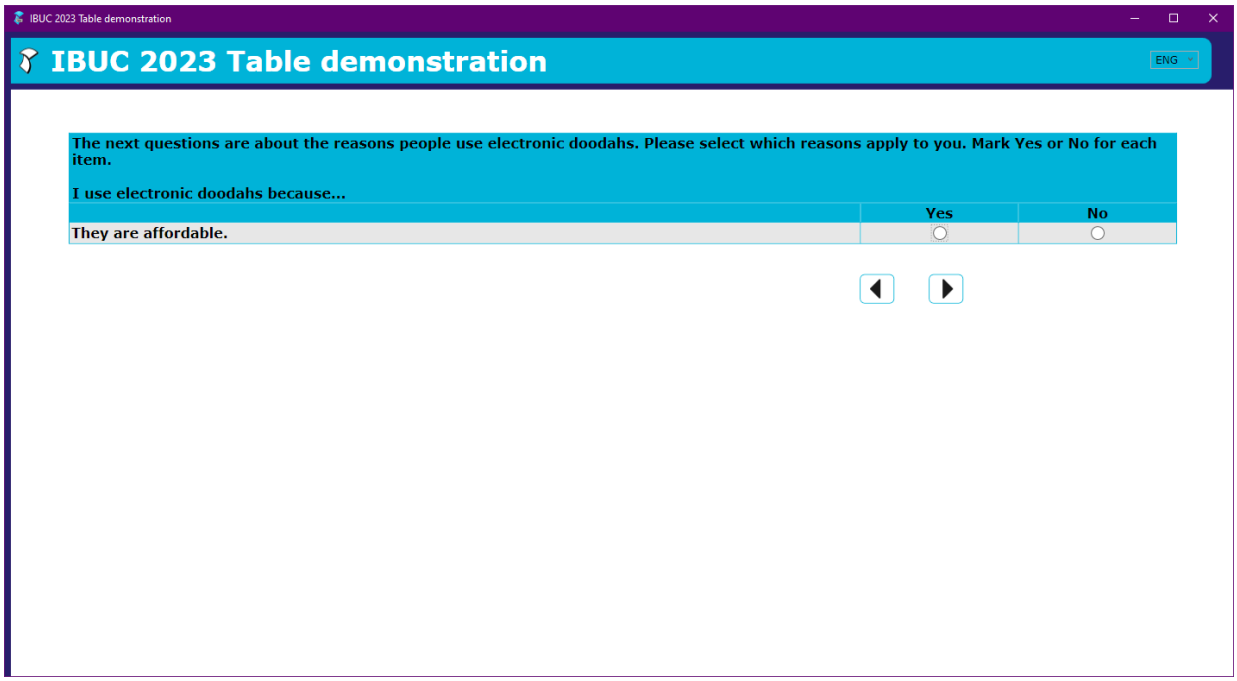


There is a scroll bar so that we can see lower rows, but if the rows and navigation buttons are shown, then we cannot see the question or the column headings:

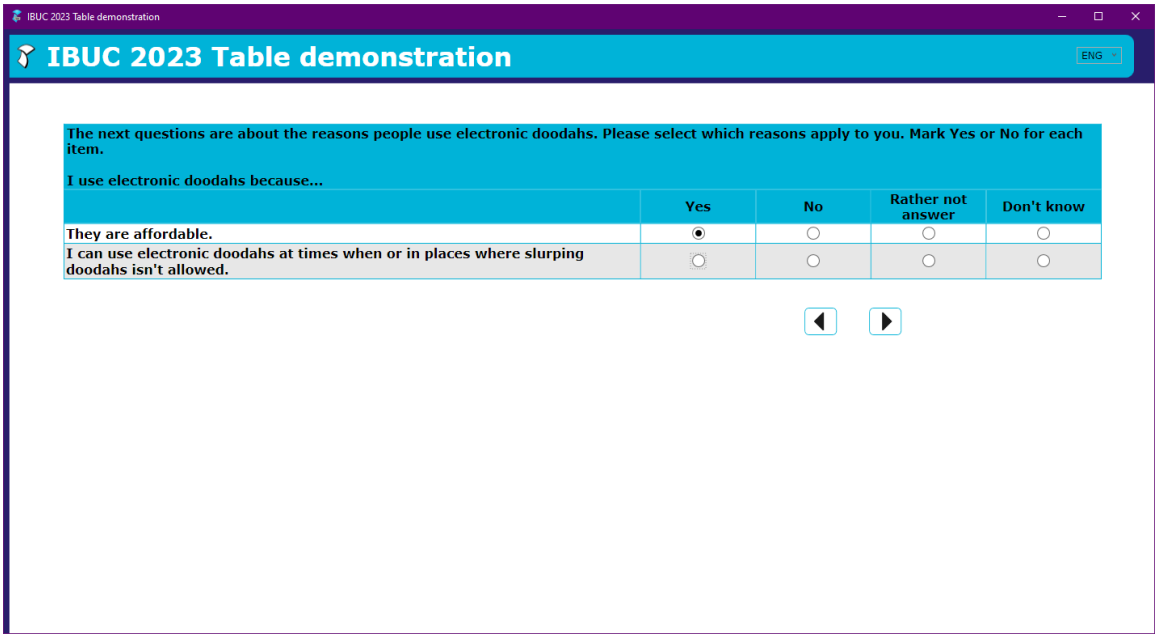


2.1.2 The Growing Options Table

We have already come across an expanding (or growing) table, and here is an example based on a study's requirements:



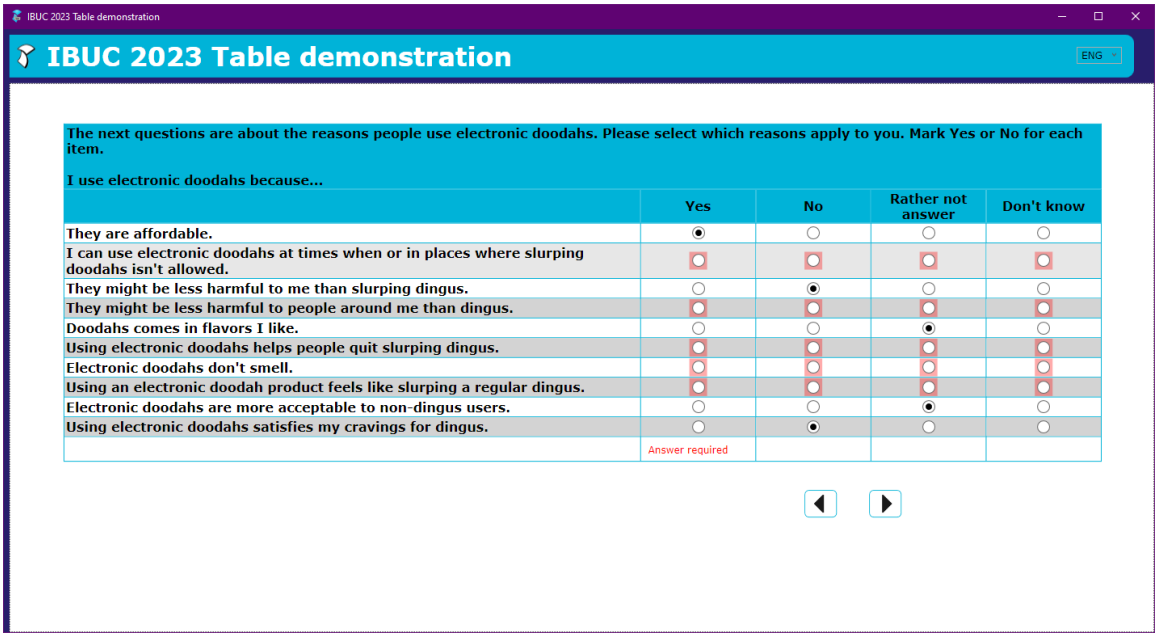
Select buttons for the first rows, that is, select a button then select next, and so on.



Because of the way that the appearance of rows is controlled in the Blaise code:

```
ROW1
IF ROW1 <> EMPTY OR ROW1.IsVisited = 1 THEN
  ROW2
ENDIF
```

Our client wanted us to allow new rows to appear, even though an answer was required. We were not to show any indication that answers were required until the last row was reached and the next page was selected.



This goes beyond the normal behavior of DK/RF.

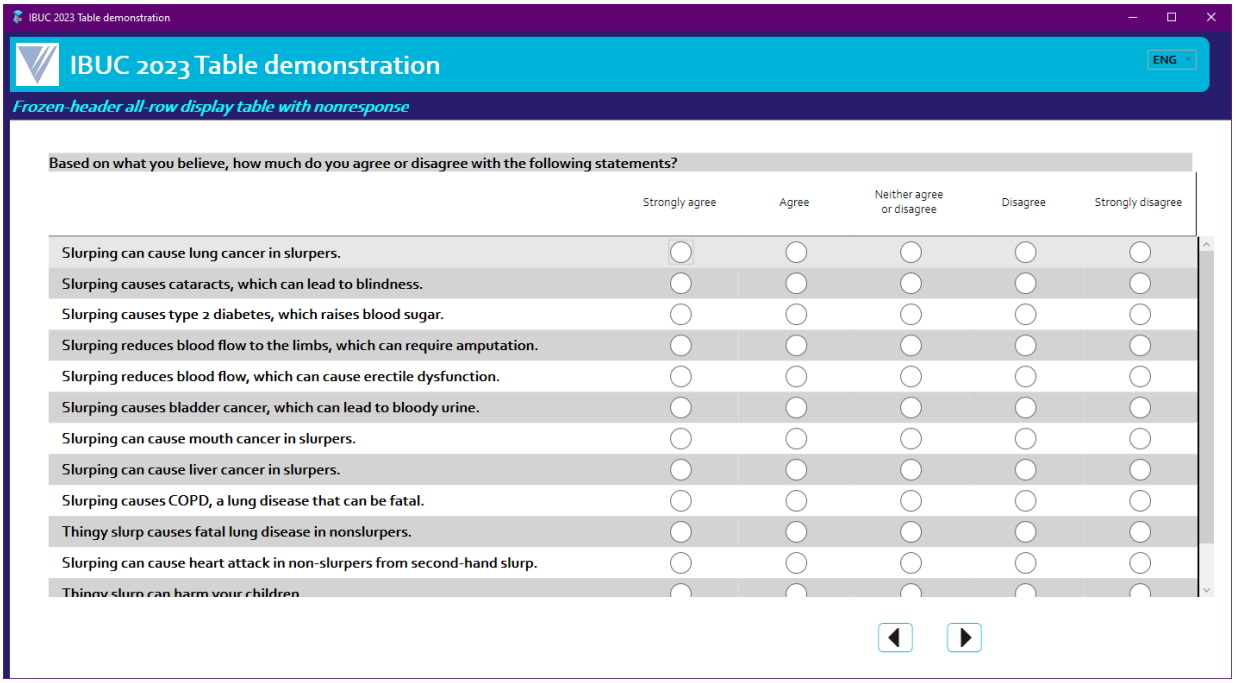
2.2 Elaborated Views

Our client had two major concerns:

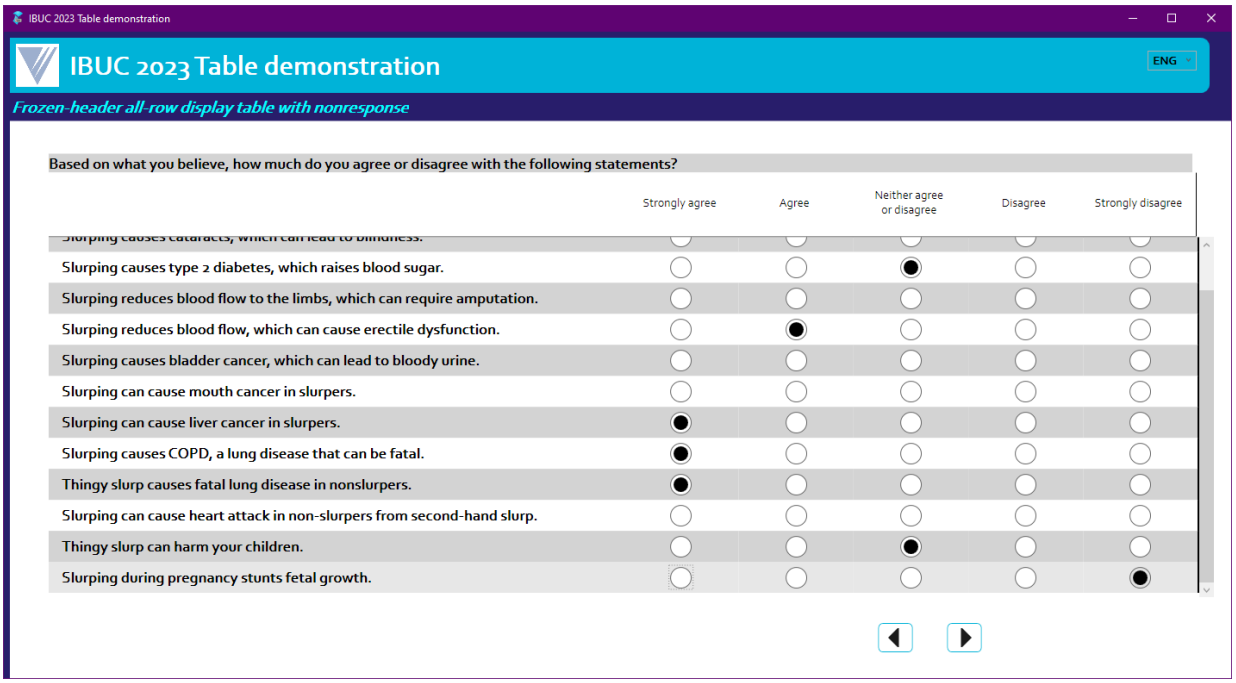
- The client did not want either the navigation buttons or the table headings (question and option headings) to disappear when scrolling. Their preference was to have a frozen header such as existed in Excel.
- The client did not want the answer-required columns to appear until the respondent had gone through the complete table.

2.2.1 The Standard Options Table Version

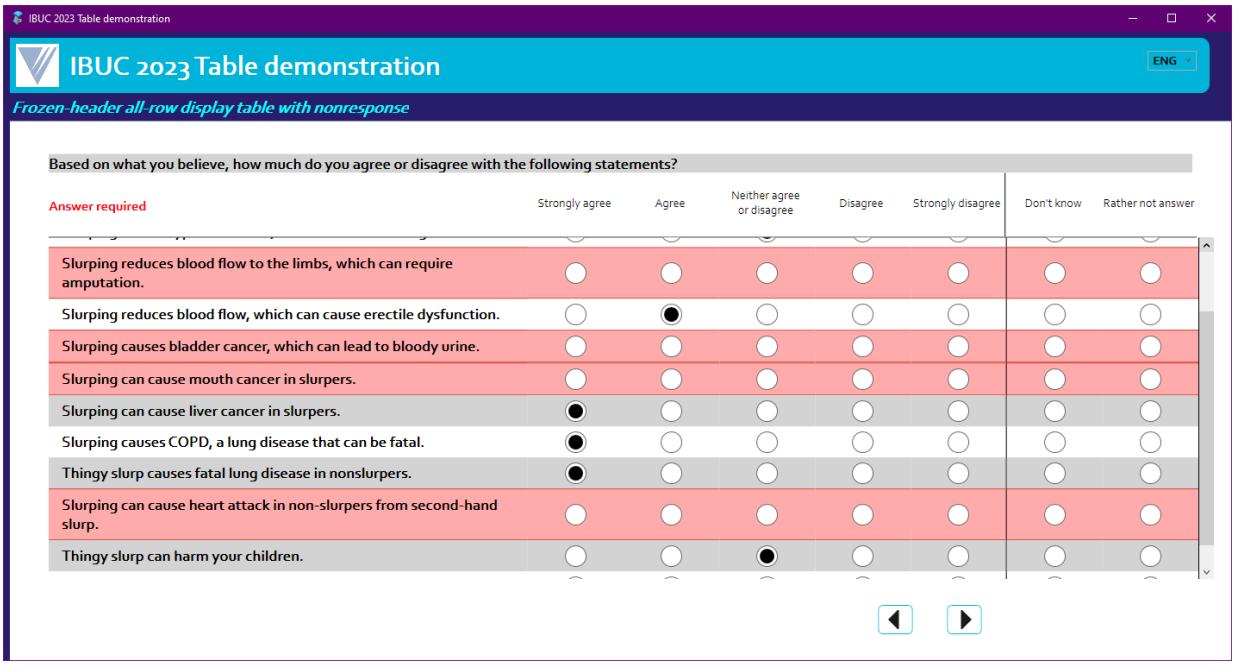
After some design work that involved paying a good deal of attention to the resource database, we have an approach that satisfies our original client and is being evaluated by others.



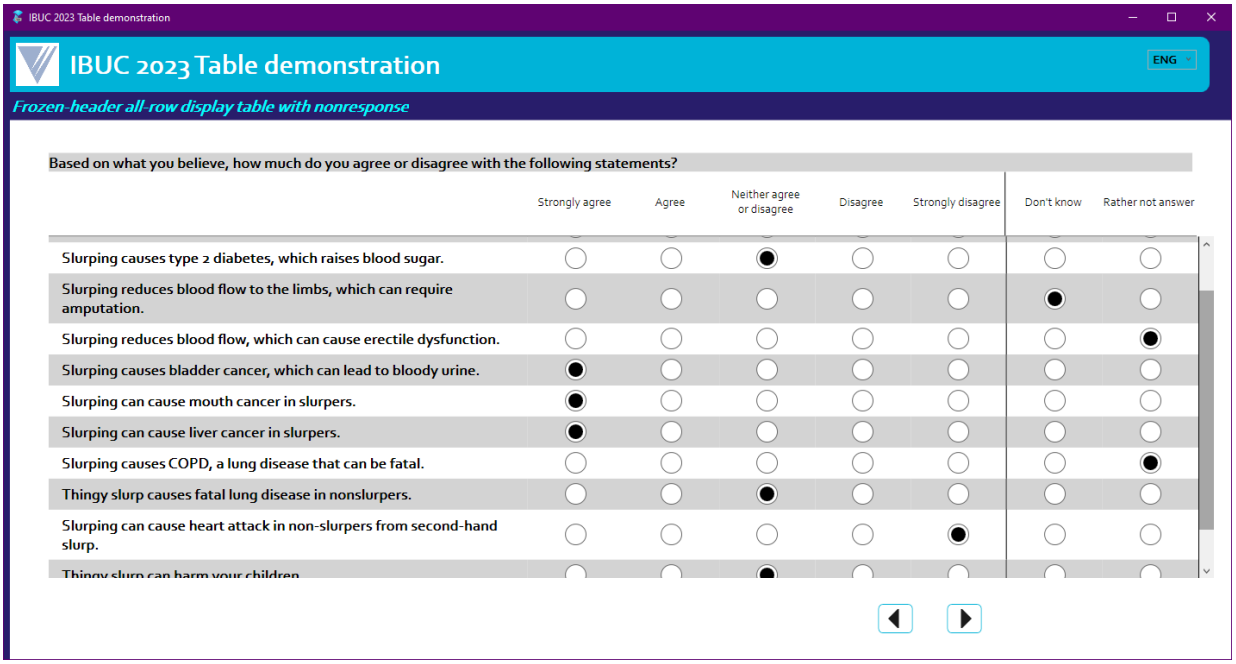
We answered some, but not all, questions because the specification said that the respondent should not be troubled by missing an answer:



However, respondents needed to answer all the questions, and so when they select the next-page button:

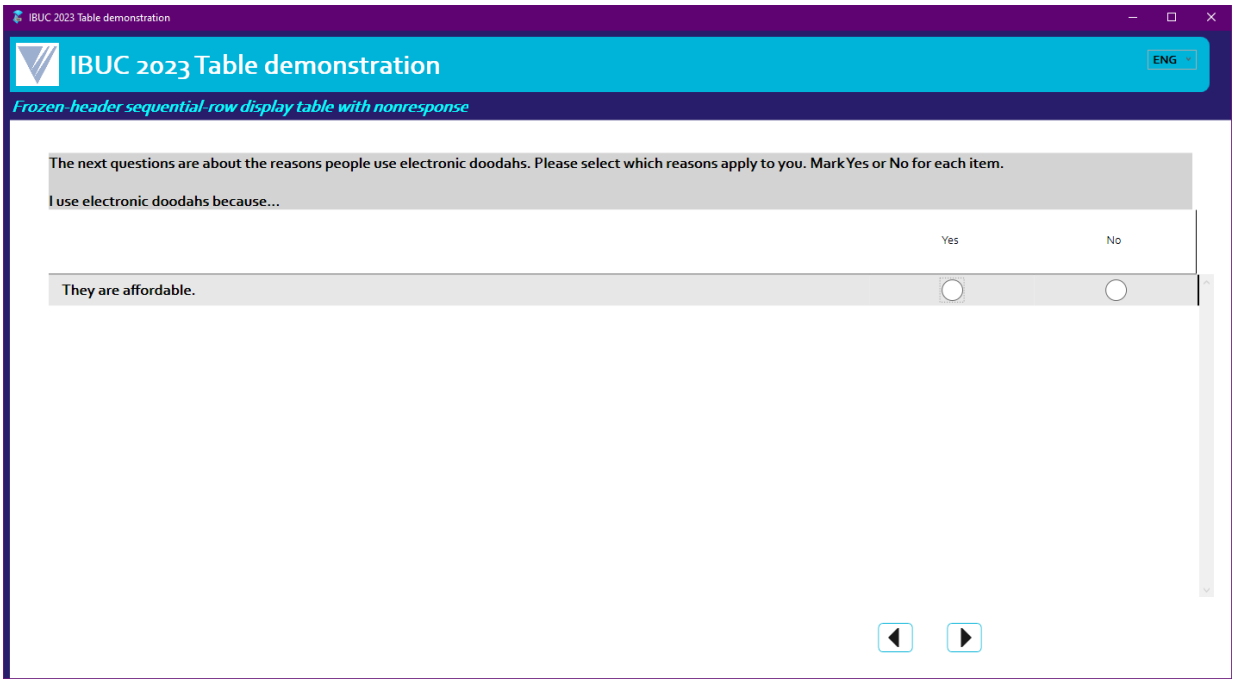


The rows without answers are highlighted, and the respondent has to answer all the missing rows:

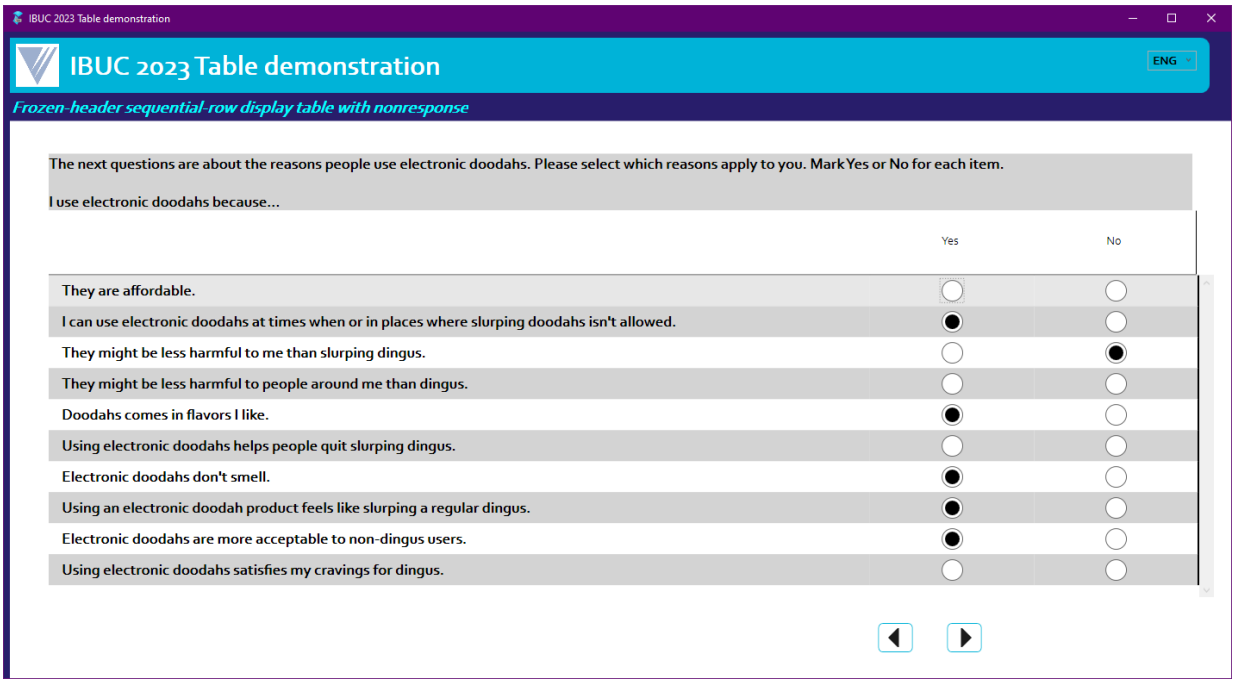


2.2.2 The Growing Options Table Version

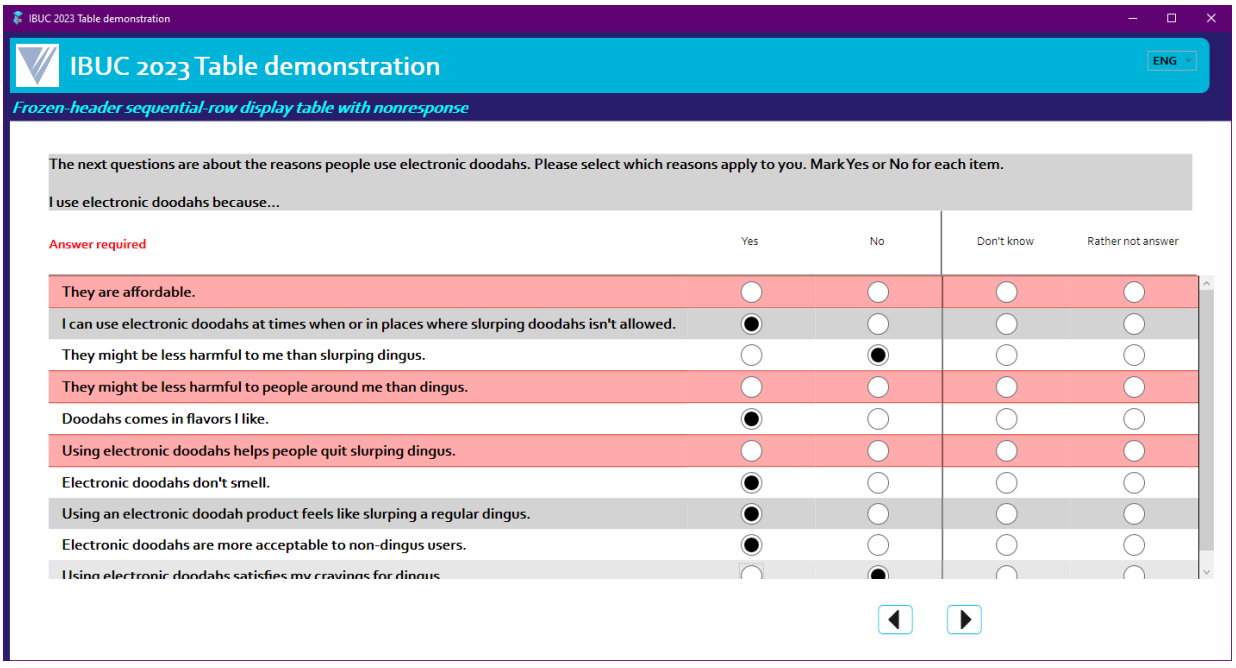
We start here:



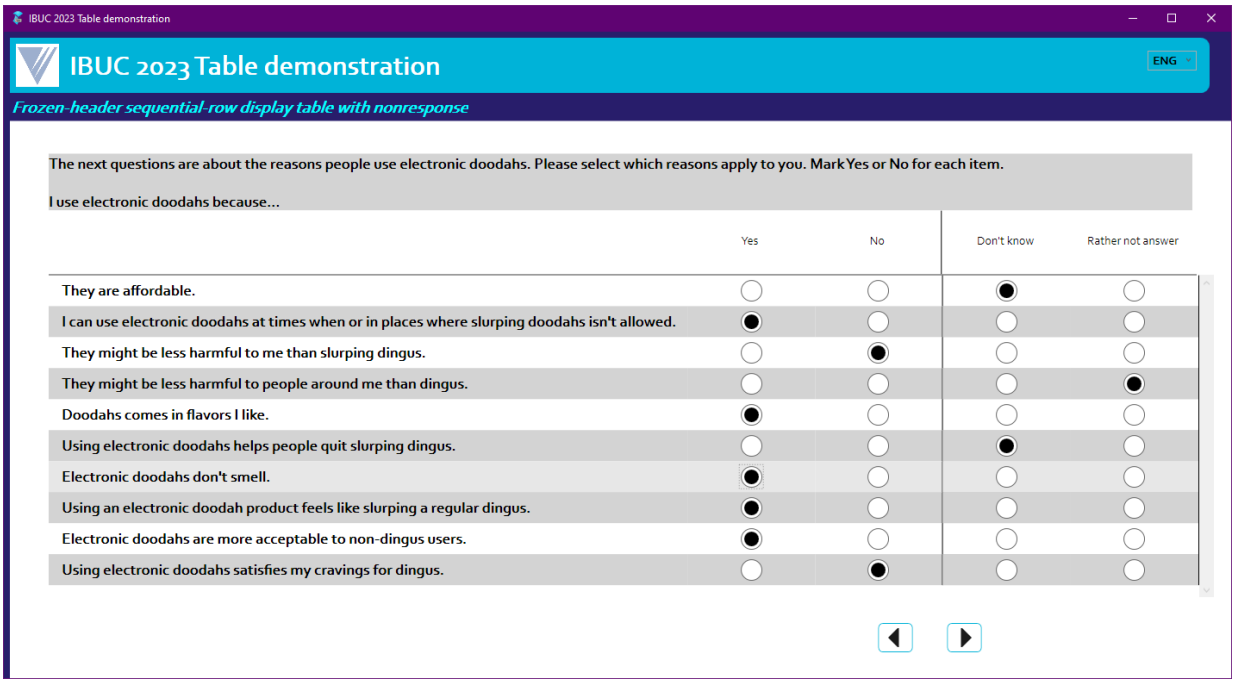
The respondent is not forced to answer a question:



However, at the end of the questions and when the next-page button is selected:



The respondent has to answer all the questions before continuing to the next page.



3. Conclusions

There are many powerful tools in Blaise 5, and many of them reside in the resource database. There are so many tools that we are only now discovering some of them—and many are not well documented. It is only by pushing the envelope that we discover some of these tools.

For example, we use a Boolean server variable named AllVisited that is true when all rows of a table have been visited. We started with:

```
ServerVariables.SetBoolean('AllVisited', Page.Fields.All(item.isvisited))
```

However, sometimes we had cases where we could see that all fields on a page were visited, but AllVisited was never true. On further investigation, we found that in the "items" were Field References that were never visited. After consulting with StatNeth, we added another condition to the All() directive:

```
ServerVariables.SetBoolean('AllVisited', Page.Fields.All(item.IsVisited OR
item.Origin = Reference))
```

That is, there was a way to distinguish between types of fields by their origin, and this feature had existed for some time (but was unpublicized). At the time of writing, the origin feature is still undocumented.

The main takeaway from our adventures with tables is that many things are possible and, in Blaise 5, those many things are often tucked away in the resource database. We always try to find a generic solution to a specific problem because we want to apply the solution to a range of instances. In the case of tables, by using tools in the resource database, we found generic solutions.