

PAPI to Blaise 5 – Challenges and Solutions in Creating Complex Tables

Emily Caron, Lilia Filippenko and R. Suresh - RTI International

1. Abstract

Recent Blaise 5 development at RTI International included creating large tables to display preload data, with the option of entering or editing the data values in similar fashion to what was previously done on paper. This brought about challenges in coding and configuring the Resource Database. The process we followed and the solutions we developed to overcome the challenges will be discussed in the paper using a household roster example.

2. Background

RTI International has a long history with developing Blaise instruments for use in survey research, as part of its overall mission to improve the human condition. We continue to primarily use Blaise version 4.8 due to long-standing survey projects using it and its overall stability. Within the past few years we have also delved into Blaise 5 with software evaluation, integration into existing case management systems, and deploying to production. We currently have one web survey in production and previously conducted a CAPI pilot survey. Three other Blaise 5 projects are in development stage with plans to deploy to production this summer.

Images and functionality mentioned in this paper come from a demo survey in Blaise version 5.6.9.2082. This survey mimics the features and behavior we wanted to highlight after programming an instrument for a client and includes different question content than what was used for the client. For simplicity's sake, the client will be referred to as "ClientX" in this paper, and discussion below will be worded as if this was the exact instrument we created for them.

3. Our Task

Recently we were tasked with creating Blaise 5 surveys on tablets to collect information which was previously entered by ClientX onto paper forms (PAPI). The PAPI forms had been in use for quite some time and contained several large tables as well as previously collected data printed on the forms. Much of the data collected was specific to household members. If a table happened to fit within the regular paper form it would be located there; otherwise, supplemental paper forms were often used to store the previous data and/or collect new data in table format.

For the Blaise 5 surveys, ClientX requested that most previous data be visible on screen when present. It needed to be retained and readily available for interviewer access. Some of this data would be available for reference only as it was not expected to change, and other data should be verified with the ability to change the data if necessary.

Another requirement was that if preload data existed and then was changed or otherwise indicated as incorrect, comment fields would route to collect any pertinent details about the changes. Additionally, the interviewer remarks feature should be available on demand.

3.1 Task Challenges re: Large Tables with Preload Data

As previously mentioned, some of the tables we needed to display on screen were overly large, and this posed a problem considering screen width and visibility. Not only would the tables take up a lot of real estate for the data we needed to collect, but we also needed to show preload information in a manner that would be readable and readily accessible to the interviewer at the time they were collecting data.

Some different layouts were considered and tested.

Option 1: Show a read-only preload table over or under a table to update or collect data. Considering the size of some of the tables, this was not a good option because of vertical scrolling. Comparing original data to the new data row was too far to travel across the screen.

Option 2: Use odd-numbered rows to show preload data, with even-numbered rows to update or collect new data. This was a bit better on space, but a little more confusing to the eye. In households with a large number of members, this would make the table unnecessarily long and require scrolling.

Option 3: Use table tags within field text above the table to display any available preload data, while leaving the table itself to update or collect new data. Like option 1, this could cause vertical scrolling and was too far of a distance between original and new data rows.

3.1.1 Our Solution for Large Tables with Preload Data

Our eventual solution was to combine the preload data and new capture or potential changes into one table with the same cells used for both. We just needed to find a way to ensure that the interviewer could access preload data on demand if needed, no matter if they had made adjustments to what was initially loaded into the table. We were able to do this with the help of a row level PreloadIn flag and a LoadHH procedure.

A PreloadIn field was created at the row level to serve as a flag to allow or prevent preload calculations when needed. Initially in the Rules section we left this field empty. We also created a LoadHH procedure to handle setting all values in the table that we knew from preload data.

```
PROCEDURE LoadHH
PARAMETERS
    piRow : INTEGER
    EXPORT pePreloadStr : STRING
RULES
    iPerson      := Preloads.Person[piRow].iPerson
    First        := Preloads.Person[piRow].First
    Last         := Preloads.Person[piRow].Last
    DOB          := Preloads.Person[piRow].DOB
    iAge         := Age(Preloads.Person[piRow].DOB)
    Sex          := Preloads.Person[piRow].Sex
    HOHrelation  := Preloads.Person[piRow].HOHrelation

    ...[some other calculations]

    PreloadIn    := 1
```

```

pePreloadStr := "
ENDPROCEDURE

```

The procedure was kicked off early in the Rules after evaluating PreloadIn value for each row. Since PreloadIn was not 1 for any rows yet, this served to initialize all known data into the table. Near the end of the procedure, PreloadIn value was set to 1 to signify that the preload info was loaded.

```

PreloadIn.KEEP
IF PreloadIn <> 1 THEN
  //fill preload data
  LoadHH(piRow,PreloadStr)
ENDIF

```

Upon arriving at the household demographics table, the interviewer could review the preloaded information and verify it as being correct or mark it as incorrect. The first column required a response or they could not move to the next screen. All columns except “Correct?” started out as read-only using .SHOW statements.

Figure 1. Household Demographics Table, original view

Correct?	P#	First name	Last name	Date of Birth	Age	Sex	Rel to O/R	Language	Education	Identity	Country	Race
Yes No	1	ANNE	DOE	7/25/1967	52	Female	Head of Hous	Only English	Professional c	Non-Hispanic	Select	White
Yes No	2	MARIE	DOE	1/22/1970	50	Female	Other Relativ	Not Applicabl	Bachelors de	Non-Hispanic	Select	White
Yes No	3	CATRINE	DOE	12/20/1993	26	Female	Son or Daugr	Not Applicabl	No schooling	Non-Hispanic	Select	White
Yes No	4	ALEX	DOE	8/1/1997	22	Male	Son or Daugr	Not Applicabl	No schooling	Non-Hispanic	Select	White

(Note: The table was much larger than this, but some rows and columns were cut for visibility’s sake.)

In the Rules, we now needed to evaluate the different values of the Correct field. If Correct = Yes, this continues to consider all other columns as read-only.

Figure 2. Household Demographics Table, all data verified as correct

Correct?	P#	First name	Last name	Date of Birth	Age	Sex	Rel to O/R	Language	Education	Identity	Country	Race
Yes No	1	ANNE	DOE	7/25/1967	52	Female	Head of Hous	Only English	Professional c	Non-Hispanic	Select	White
Yes No	2	MARIE	DOE	1/22/1970	50	Female	Other Relativ	Not Applicabl	Bachelors de	Non-Hispanic	Select	White
Yes No	3	CATRINE	DOE	12/20/1993	26	Female	Son or Daugr	Not Applicabl	No schooling	Non-Hispanic	Select	White
Yes No	4	ALEX	DOE	8/1/1997	22	Male	Son or Daugr	Not Applicabl	No schooling	Non-Hispanic	Select	White

Let’s consider the scenario where Correct = No for one or more rows where data needs to be modified. In this situation we needed to be sure to prevent the continual overlaying of valid changes to the data by the preload data. This is where PreloadIn flag came in handy once again.

When handling Correct = No, first we reset PreloadIn = 1 to ensure the LoadHH procedure didn’t get kicked off for any of these rows. Then we did .keep statements on all of the fields, followed by asking the fields where we would allow data changes:

```

IF Correct = No THEN
  PreloadIn := 1
  iPerson.keep
  First.keep
  Last.keep
  ...[other .keep statements]
  iPerson.show //not allowed for edit
  First
  Last
  DOB
  iAge := Age(DOB)
  iAge.show //calculated, not allowed for edit
  Sex
  HOHrelation
  Language
  EducLevel
  Hisporig
  CountryOrigin
  Race

```

Figure 3. Household Demographics Table, some rows indicated as not correct and fields are updated

Correct?	P#	First name	Last name	Date of Birth	Age	Sex	Rel to O/R	Language	Education	Identity	Country	Race
<input type="radio"/> Yes <input type="radio"/> No	1	ANNE	DOE	7/25/1967	52	Female	Head of Hous	Only English	Professional c	Non-Hispanic	Select	White
<input type="radio"/> Yes <input type="radio"/> No	2	MARY	DOE	1/22/1968	52	Female	Other Relativ	Spanish and	Bachelors deg	Hispanic	Select	White
<input type="radio"/> Yes <input type="radio"/> No	3	CATRINE	DOE	12/20/1993	26	Female	Son or Daugt	Not Applicabl	No schooling	Non-Hispanic	Select	White
<input type="radio"/> Yes <input type="radio"/> No	4	ALEX	DOE	8/1/1997	22	Male	Son or Daugt	Not Applicabl	No schooling	Non-Hispanic	Select	OtherAsian

So now we had some data modifications. But recall that one of the requirements from ClientX was for the interviewer to have easy access to any available preload data. What if they needed to get back to the original data values? Due to space and other reasons, we were now combined into one set of fields all in one table.

We achieved this by using the PreloadIn flag again, with the use of the LoadHH procedure. When Correct <> No, we could invoke the code below in order to return data to its original preload state if the interviewer switched from Correct = Yes to No.

```

Correct
...[some other code]
PreloadIn := 0
LoadHH(piRow,PreloadStr)
iPerson.show
First.show
Last.show
DOB.show
...etc

```

Figure 4. Household Demographics Table, one Correct=No row is set back to Yes to restore preload data

Correct?	P#	First name	Last name	Date of Birth	Age	Sex	Rel to O/R	Language	Education	Identity	Country	Race
Yes No	1	ANNE	DOE	7/25/1967	52	Female	Head of Hous	Only English	Professional c	Non-Hispanic	Select	White
Yes No	2	MARIE	DOE	1/22/1970	50	Female	Other Relativ	Not Applicabl	Bachelors de	Non-Hispanic	Select	White
Yes No	3	CATRINE	DOE	12/20/1993	26	Female	Son or Daugt	Not Applicabl	No schooling	Non-Hispanic	Select	White
Yes No	4	ALEX	DOE	8/1/1997	22	Male	Son or Daugt	Not Applicabl	No schooling	Non-Hispanic	Select	OtherAsian

The caveat is this does lose any newly changed data, but ClientX was fine with it because this situation would only occur during the interview when any necessary data modifications could be easily re-entered, and in general this was only expected to happen under rare circumstances.

3.2 Task Challenges re: Handling Comments for Changes to Preload Data and Remarks

A requirement from ClientX was to force comment capture when any preload data values were changed. In many cases, we were able to do this quite easily on an individual question basis. Other screens contained simple table layouts, such as shown in Figures 5 and 6, where comment fields were routed as a new row if a field value didn't match its counterpart in a preload block.

Figure 5. Simple Address Information Table, original view

1. Verify complete address (including county) on questionnaire:

Address	<input type="text" value="55 BOGUS RD"/>
City	<input type="text" value="SOMEWHERE"/>
State	<input type="text" value="NC"/>
Zip	<input type="text" value="12345"/>
County	<input type="text" value="TESTCOUNTY"/>
Phone	<input type="text" value="(999) 999-9999"/>

Figure 6. Simple Address Information Table, change to Address field put Address Comments on route

1. Verify complete address (including county) on questionnaire:

Address	559 BOGUS RD
Address Comments	change occurred, so leave a comment
City	SOMEWHERE
State	NC
Zip	12345
County	TESTCOUNTY
Phone	(999) 999-9999

In other situations, dealing with comments became problematic because we needed to put them on route when preload data was modified in a large table. We started out by adding a comments column to the right-hand side of tables, which would force comments to be left on a row-by-row basis when changes occurred to fields within that row. This was not desirable to ClientX because either the string length of the comment fields had to be very short to completely fit on the screen, or the comment column width was shortened in layout so not much of the comment would be visible at one time (the cell would scroll).

Regarding interviewer remarks, this feature appeared to require updates in the Resource Database to get it to work similar to behavior in Blaise 4. Only when the paperclip image was visible did it seem to work correctly, but otherwise it was not right.

3.2.1 Our Solution for Handling Comments for Changes to Preload Data and Remarks

It was decided that for comments connected to large tables, one very large comment field routed below the table would suffice. The field was large enough to accommodate any and all information about data changes. The interviewers were previously trained to leave comments on paper about any changes they happened to pick up on using PAPI mode, so this was still an improvement.

The comments field would be required if any rows showed Correct = No, and this was done using a QxxChanges auxfield (xx corresponded to question number). Putting it all together:

```

QxxChanges := EMPTY
IF Preloads.HHTotal > 0 THEN
  Qxxintro.SHOW
  Qxxtable(Preloads.HHtotal)
  IF Qxxtable.HH[Preloads.HHTotal].Correct = RESPONSE THEN
    FOR k := 1 TO Preloads.HHTotal DO
      IF Qxxtable.HH[k].Correct = no THEN QxxChanges := yes ENDIF
    ENDDO
  ENDIF
ENDIF
IF QxxChanges = yes THEN
  QxxComments

```

ENDIF

Figure 7. One or more Correct = No responses results in required comments

Yes	No	4	ALEX	DOE	8/1/1997	22	Male	Son or Daugh
Yes	No	5	CHRISTI	DOE	6/6/2000	19	Female	Long Term Vi

Comments:

Forced to enter comments when Correct = No.

< Back Next >

For interviewer remarks, we updated the shortcut to Ctrl+C as this would be easier to access from the onscreen keyboard with just one hand. The Resource Database needed modification to use the newly created “OnCtrlC” event, and this included updates to Horizontal, Vertical, InfoPane, and Appointment field pane templates. The following logic was also added to the remarkButton/clip Visibility property:

```
IF LEN(Field.FieldProperties.GetProperty('Remark').StringValue) > 0 THEN
    'Visible'
ELSE
    'Hidden'
ENDIF
```

3.3 Task Challenges re: Collecting Data in Tables with Undefined Number of Columns

Another challenging section of one of ClientX’s surveys involved collecting information for a list of household and closed family members after selecting different services used by the household. On that page of the PAPI instrument, the interviewer would write down the list of household names and then write the services selected for a given name. There could be any number of services selected, and any number of household members included from preload plus newly added members.

3.3.1 Our Solution for Collecting Data by using “OptionButtons” Table

For the corresponding service selection section in the CAPI instrument, we came up with a solution for interviewers to first collect the list of the services used by family members and then on the following screen use check boxes to specify a service for a given family member.

For our demo instrument, we use a predefined list of American department stores as an example of “services” with the ability to add more stores to the list.

Figure 8. Stores are selected from standard list presented to all respondents

RTI INTERNATIONAL Table with Undefined Number of Rows and Columns English

Considering last year's Christmas shopping, from which stores did you and your family purchase presents?

(READ THE COMPLETE LIST AND SELECT THOSE THAT APPLY):

- Belk
- Dillard's
- Macy's
- Saks Fifth Avenue
- Sears
- Nordstrom
- JCPenney
- H&M
- Gap
- American Eagle
- Express
- Don't know
- Refused

< Back Next > Exit Home End

If other stores were shopped than what was initially listed, the respondent is asked to provide names of the additional stores. This CAPI interview would use a tablet, so as a rule we used push buttons for “Yes”/“No” options and present “Don’t know” and “Refused” options as simple buttons.

Figure 9. Screen to capture other stores shopped

RTI INTERNATIONAL Table with Undefined Number of Rows and Columns English

Were any other stores or websites used to buy presents that we have not already discussed?

Yes

No

Don't know

Refused

Please enter up to 8 additional store names most frequently used for purchasing Christmas presents last year.

Store Name	Any More?	
Finish Line	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Lululemon	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Pandora	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Sephora	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Best Buy	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Amason.com	<input type="radio"/> Yes	<input checked="" type="radio"/> No

< Back Next > Exit Home End

After stores are marked on the initial screen and any additional are collected after, this information is combined to generate columns for a new table where the interviewer can indicate which stores were shopped by each family member. The known family members (with age) make up the rows of the table.

Figure 10. Table to indicate stores shopped per family member

Table with Undefined Number of Rows and Columns

What stores were used to buy presents for each family members?

	Belk	Saks Fifth Avenue	Nordstrom	H&M	Gap	American Eagle	Express	Finish Line	Lululemon	Pandora	Sephora	Best Buy	Amason.com
SUSAN (40)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HERBERT (42)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
JENNIFER (15)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
TREVOR (13)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
JASON (5)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
MARGRET (62)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
JEAN (32)	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
MIKE (34)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ANITA (3)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

< Back Next > Exit Home End

In the program, all variables for populating rows and columns in the table above are defined as arrays:

FillNameAge : ARRAY[1..20] OF STRING[40] {Family member Name and Age}
 FillStore : ARRAY[1..20] OF STRING[30] {Name of selected or added Store}

Fields in the block to hold all selections are defined as below. This table will be populated only if at least one store was added to the combined list of stores, so attributes “Don’t know” and “Refusal” are not allowed, but “Empty” is allowed. A check is in place to ensure at least one store is selected.

FIELDS //Family Members and Stores

D1 "^{FillNameAge[1]}/^{FillNameAge[1]}": SET OF tStores, NODK, NORF, EMPTY
 D2 "^{FillNameAge[2]}/^{FillNameAge[2]}": SET OF tStores, NODK, NORF, EMPTY
 ...etc.

Figure 11. At least one store must be selected

Table with Undefined Number of Rows and Columns

What stores were used to buy presents for each family members?

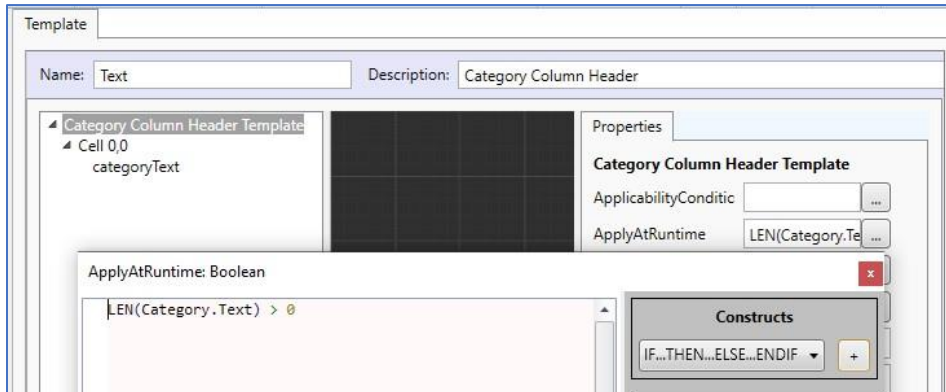
	Belk	Saks Fifth Avenue	Nordstrom	H&M	Gap	American Eagle	Express	Finish Line	Lululemon	Pandora	Sephora	Best Buy	Amason.com
SUSAN (40)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HERBERT (42)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
JENNIFER (15)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
TREVOR (13)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
JASON (5)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MARGRET (62)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
JEAN (32)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MIKE (34)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ANITA (3)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

PLEASE SELECT AT LEAST ONE STORE.

< Back Next > Exit Home End

Only one change was made to the default Resource Database to “Text” of “Category Column Header Template” to show names for only needed stores to display in the column titles.

Figure 12. Category Column Header Template change was needed

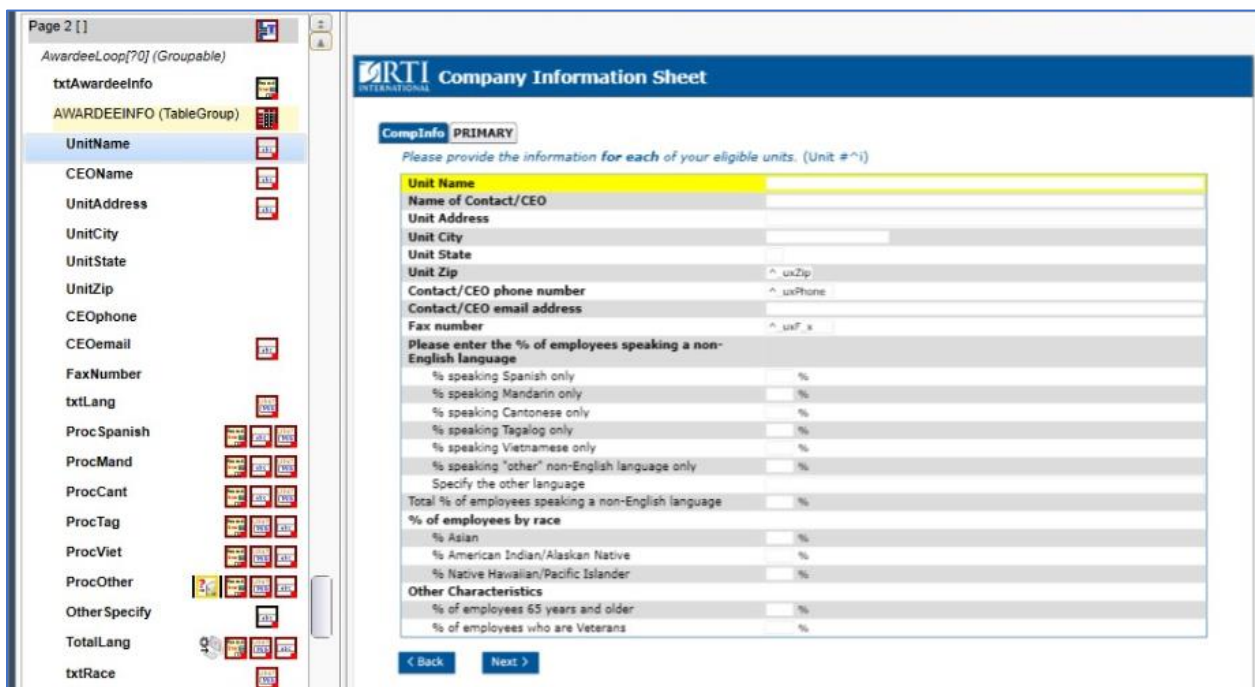


4. Use of Generic Pages with Tables

We thought it was worth mentioning that when Generic pages were introduced in Blaise 5, we attempted to use them with our demo instrument described in Section 3. In addition to collecting data in complicated tables, a few blocks were needed to collect information per household member on one page.

Unfortunately, each row in the previously defined tables was presented on a new page after setting “Generic Pages” for our demo instrument. But we did successfully use them on another project where the requirement was to collect data presented in an Excel spreadsheet for a CAWI instrument. Considering that up to 120 pages could be entered, that was great to use.

Figure 13. Use of Generic Pages with Table



5. Conclusions

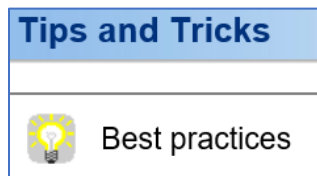
During our recent development of Blaise 5 instruments, we found that large tables can be used and generally work well, but they need extensive testing and tweaking to get them to function and appear just right. Most large tables using tablet or smaller screen size will be a challenge.

Once a layout is applied to fields within in a table, any change made to the number of fields in the table should be done carefully and without numerous changes made all at once. This tends to throw off the layout. A better way to do it is to make one change at a time, test, and then save after confirming the table layout looks good. Layout continues to be where we spend most of our time in fine-tuning instruments.

In our experience thus far, the default company Resource Database can be used quite nicely but just needs additional templates developed for specific instruments. Typically, this doesn't take much time or effort. Having a solid company Resource Database template to reuse across instruments is key.

In general, if something is not working as expected, checking online help under "Tips and Tricks" has been quite helpful. Often the solution is right there.

Figure 14. Find "Tips and Tricks" in the Blaise 5 online Help index



6. References

Blaise 5 Help Reference Manual, Statistics Netherlands

<http://help.blaise.com/>

Lilia Filippenko, Preethi Jayaram, Joe Nofziger, Brandon Peele, R. Suresh, "Blaise 5 with RTI's Integrated Field Management System on Field Interviewer Laptops", 2018, Proceedings of the 18th International Blaise Users Conference

R. Suresh, Keith Bajura, Matt Boyce, Emily Caron, Lilia Filippenko, Preethi Jayaram, Patty LeBaron, Joe Nofziger, Jean Robinson, Gil Rodriguez, Vorapranee Wickelgren, "Rolling Out Blaise 5 – An Interesting Journey", 2018, Proceedings of the 18th International Blaise Users Conference