2015 Survey Solutions
Washington DC, USA

Survey Solutions: List Functions

Sergiy Radyakin

sradyakin@worldbank.org

Development Data Group (DECDG),
The World Bank

September 28, 2016

THE WORLD BANK

Multiple choice questions may contain two or more items and each of the items may be selected or left unselected by an interviewer. Typical use situations:

- Which of the following appliances are present in your household?
  - microwave oven
  - refrigerator
  - stove
  - dishwasher
  - washing machine
  - dryer
- Which of the following have you consumed over the last 7 days?
  - milk
  - toast
  - honey

It is essential to understand that the response to a multiple choice question is not one value, but a list of values.

To check if the selection is left empty examine the `Length` property. Length returns an integer length of the selection list.

Examples:

1. Check that no item in the question *mcq* was checked:

   `mcq.Length==0`

2. Check that exactly one item in the question *mcq* was checked:

   `mcq.Length==1`

3. Check that more than one item in the question *mcq* was checked:

   `mcq.Length>1`

Do not compare the value of a multiple choice question with a `null`. It is never `null` regardless whether any selection was made.

If any selection was made in a multiple choice question, the value of the mcq will be a list of the codes of the selected items. The codes are assigned at the Survey Solutions designer when the question is added to the questionnaire. Each item must have a unique numeric code.

To check if a particular item was included in the selection, use the Boolean function `Contains()`. For example, to check for item with code *101* in the selection of the question *mcq*:

```
mcq.Contains(101)
```

The result is *true* if the item with the code *101* is checked in the question with variable name *mcq*, regardless whether any other items are selected. If the item with the code *101* is not checked, the result is *false*.

Note that the function doesn't issue an error if the mentioned code is not present in the selection choices of the multiple choice question.

In case we are interested in detecting the situation where only one particular item is selected, the Boolean function `ContainsOnly()` can be used.

For example, to check that only the item with code *103* is in the selection of the question *mcq*:

```
mcq.ContainsOnly(103)
```

The result is *true* if the item with the code *103* is checked in the question with the variable name *mcq* and no other item is selected. Otherwise the result is *false*.

In case we are interested in detecting the situation where several particular items are selected, the Boolean function `ContainsOnly()` can be used with a list of these values instead of a single argument.

For example, suppose a washing machine has code *105* and a dryer has a code *106* in the appliances question. We can check for selection of both and no other items with the following:

```
mcq.ContainsOnly(105,106)
```

Now, suppose we are interested in detecting the situation when all of the particular values are selected, regardless of the selection of the other items.

Recall that one can write:

```
mcq.Contains(105) && mcq.Contains(106)
```

A shorter notation for the same is using the `ContainsAll()` Boolean function:

```
mcq.ContainsAll(105,106)
```

The result is *true* if all of the specified items were selected in the multiple choice question with the variable name *mcq* regardless of the other items. The result is *false* otherwise.

Next, consider we are interested in detecting the situation where any of the specific values is selected.

Recall that one can write:

```
mcq.Contains(105) || mcq.Contains(106)
```

A shorter notation for the same is using the `ContainsAny()` Boolean function:

```
mcq.ContainsAny(105,106)
```

The result is *true* if any of the specified items were selected in the multiple choice question with the variable name *mcq* regardless of the other items. The result is *false* otherwise, meaning all of the mentioned values were left unchecked.

Suppose we are interested in detecting a situation when none of the specific items was selected.

This can be done negating the result of the `ContainsAny()` function:

```
!mcq.ContainsAny(105,106)
```

Note the exclamation mark at the beginning of the expression, which is part of the syntax and denotes logical negation of the subsequent Boolean function.

The result of this expression is *true* if the neither of the items 105 and 106 is checked, regardless of the other items' selections, and *false* otherwise.

To probe for *"Exactly one item is selected and it is neither 105 nor 106"* use the following expression:

```
mcq.Length==1 && !mcq.ContainsAny(105,106)
```

Multiple choice (multiselect categorical) questions reviewed so far were providing only two states for the options - selected or not selected (checked or not checked).

In some cases it is important to distinguish between the situation when the respondent doesn't have an item (answer is no), and the respondent doesn't indicate the presense of a particular item (answer is missing).

In such cases a multiple choice question with explicit Yes/No options should be used. This feature is available in Survey Solutions 5.3 and later.

The multiple choice question is defined in exactly the same way as before, what differs is its presentation on the interviewer's screen. The setting that controls it is *"Yes/No mode"*.

When it comes to values, multiple choice questions present themselves as a collection (array) of selected items. Correspondingly, any item that is not present in this array was not selected on the screen by an interviewer.

Multiple choice questions in the *"Yes/No mode"* present themselves for validation through 4 separate arrays: *Yes*, *No*, *Missing*, and *All*.

One can apply all of the list functions (Length, Contains(), ContainsAny(), ContainsAll(), ContainsOnly()) discussed earlier for the multiple choice questions to individual arrays of the multiple choice question in the *"Yes/No mode"*.

Specifically, for a question named *mcq*:

- `mcq.Yes` is an array of codes of items that were marked as "Yes" by the interviewer;
- `mcq.No` is an array of codes of items that were marked as "No" by the interviewer;
- `mcq.Missing` is an array of codes of items that were not marked at all by the interviewer;
- `mcq.All` is an array of all codes defined for the multiple choice question, regardless of the selection status;

Suppose our survey contains a multiple choice question named *assets* in Yes/No mode:

Which of the following assets are present? assets

101 Phone
102 TV
103 DVD player
104 VCR
105 Computer
106 Printer
107 Scanner
108 Modem
109 Microwave oven
110 Water heater

We would like to do the following:

1. Indicate the presence of a DVD player or a VCR in the absence of a TV as an unlikely situation;
2. Indicate the presence of a printer, scanner, or a modem in the absence of a computer as an unlikely situation;
3. If the respondent said a phone is present, we would like to inquire whether it is a smart phone (ask question *smartphone*);
4. If the respondent said TV is present, we would like to inquire whether it is a smart TV (ask question *smarttv*).

The first two requirements are examples of validation of question *assets*:

```
!(self.Yes.ContainsAny(103,104) && self.No.Contains(102))
    && !(self.Yes.ContainsAny(106,107,108) &&
    self.No.Contains(105))
```

Note the use of keyword *self* to address the value of the question being validated.

Read the above as following: *"Consider selection valid if only it is not a combination of positive selection of any of items 103, 104, and at the same time a negative selection of the item 102, and not a combination of positive selection of any of items 106, 107, 108 and at the same time a negative selection of the item 105"*.

The last two requirements are examples of enabling conditions (resp. for questions *smartphone* and *smarttv*):

```
assets.Yes.Contains(101)
```

Single choice questions allow only one selection of the suggested choices. The choices should be mutually exclusive.

Until the choice is made, the value of a single choice question is *null* meaning "unanswered".

Once the answer is set by the interviewer, the value of a single choice question is the code of the selected item. The codes of items are numerical codes assigned by the questionnaire developer when the question is added to the questionnaire in Survey Solutions Designer.

Values that are not provided as options can't be recorded into a single select categorical questions. Hence, include the "none of the above" option in the choice set to avoid not answered questions.

To check if the answer to a single choice question is a particular value one can use operators == and !=.

For example, given the following question structure:

What is your maximum level of education? scq

0 None
1 Primary
2 Secondary
3 Tertiary

Check if the respondent's education is exactly *secondary*:

```
scq==2
```

Check if the respondent has some education:

```
scq>=1
```

Given the following country of birth question *country*:

Where were you born?	country
1 Estonia	
2 France	
3 Germany	
4 Greece	
5 Latvia	
6 Lithuania	
7 Other country	

Check if the respondent was born in any of the Baltic countries (Estonia, Latvia, Lithiania):

```
country==1 || country==5 || country==6
```

Or in a shorter notation with the help of `InList()` Boolean function:

```
country.InList(1,5,6)
```

In many cases there are multiple alternative expressions for the same objective. For example, to check that the respondent was born not in any of the Baltic countries (Estonia, Latvia, Lithuania):

```
!(country==1 || country==5 || country==6)
```

Or alternatively:

```
country!=1 && country!=5 && country!=6
```

Or alternatively:

```
!country.InList(1,5,6)
```

Or with the help of the `IsNoneOf()` Boolean function:

```
country.IsNoneOf(1,5,6)
```

Suppose a survey asks 6 questions on how accessible was an infrastructure item (*bank*, *post office*, *market*, *hospital*, *school*, *bus or train station*) to the respondent.

Each response is measured on the scale: 1 *not accessible*, 2 *accessible*, 3 *easily accessible*.

For respondents that indicated at least 3 items of 6 were not accessible we want to ask another question, for example, which of the proposed measures the respondent will improve the infrastructure accessibility.

We can use the `CountValue()` function to write the enablement condition for this question:

```
CountValue(1,scq1,scq2,scq3,scq4,scq5,scq6)>=3
```

Note that the value of the *country* question is `null` until the question is answered. One must decide what should be the behavior of validations and enablement of subsequent questions depending on the current question being answered or skipped.

For example, one may decide that a subsequent question is opened only when a particular value is registered as an answer to a single choice question. In that case simply checking for that value is sufficient:

```
country==7
```

In other cases, one may decide subsequent question must be enabled only when answer was obtained, but it is not a special value, then probing for `null` is necessary:

```
country!=null && country!=7
```

When working with multiselect questions, a common practical situation is to provide one or both of the special choices: "all of the above" and "none of the above".
This can be done conveniently with the filtering of the options. Indeed if any of the special options is selected, nothing else applies, and if any of the normal options is selected then the special options should not apply.
Suppose regular options are 1..9 and special options are 1000 "none of the above" and 2000 "all of the above".
We apply the following:

filtering condition

```
 !IsAnswered(self)
|| (optioncode==1000 && self.Contains(1000))
|| (optioncode==2000 && self.Contains(2000))
|| (optioncode!=1000 && optioncode!=2000 && !self.ContainsAny(1000,2000))
```

Another common pattern often occurring in practice is to ask for a list of applicable items, then ask which of the selected items has the most prominent significance.

For example, consider the following two questions:

Which kinds of fuel do you use to heat your house? fuel

1 Wood
2 Wood pellets
3 Heating oil
4 Propane gas
5 Electric

Which kind of fuel is the main one you use to heat your house? mainfuel

1 Wood
2 Wood pellets
3 Heating oil
4 Propane gas
5 Electric

Obviously we want to ask the *mainfuel* question only in cases when multiple selections have been made to the *fuel* question.

So, we add the following enablement condition to the *mainfuel* question:

```
fuel.Length>=2
```

We furthermore want to make sure that the choice made in the second question is present in the selection to the first one, so we add the following validation condition to the *mainfuel* (second) question:

```
fuel.Contains((decimal)mainfuel)
```

Note the use of the `(decimal)` keyword. Or we can limit the choices right away by providing a filter condition to shortlist:

```
fuel.Contains((decimal)@optioncode)
```

Survey Solutions provides a different way of achieving the same goal: When adding a multiple choice question the designer may request that the software must track the importance of choices. In that case the question can be reformulated to:

Which kinds of fuel do you use to heat your house? fuel
Name the most important fuel types first.

1 Wood
2 Wood pellets
3 Heating oil
4 Propane gas
5 Electric

In this case the first choice will be the code of the most important fuel type.

Whether to use this approach or the approach with two questions depends on how realistic it is that the respondents will be able to rank all the fuel types before naming them to the enumerator.

Consider the following question:

What is your level of education? educat

0 None
1 Primary
2 Secondary
3 Tertiary
7 Children under 5

With the following validation expression:

```
(age>=5 && educat!=7) || (age<5) && (educat==7)
```

There is no syntax error in the above expression, but there is an error in the logic. Indeed, it leaves only one choice for the interviewer if the respondent is under the age of 5. To reduce interview time, disable the questions which only have one possible answer to them.

Revised question:

What is your level of education? educat

0 None
1 Primary
2 Secondary
3 Tertiary

With the following enablement condition:

age>=5

If necessary, the original code of 7 can be imputed for the variable *educat* after the data collection, at the data cleaning stage.

Identify an error in the expression below. Explain.

```
countriesVisited.ContainsOnly(1,2,8,9) &&
    countriesVisited.Contains(11)
```

In a hypothetical labor force survey a question on managerial career track should be asked only to *skilled employees*. Skilled employees are defined on the basis of the information about their training:

- certificate type: A, B, C, D, E, F, G (multiple choice);
- years of full time education (numeric);
- years of vocational training (numeric);

Skilled employees are ones that fall into any of the following groups:

- have a certificate of type E, or F, or G
- have both certificates of type C and D
- have a certificate of type C and 3 years of vocational training
- have 8 or more years of full time education and 5 or more years of vocational training.

Write the enablement condition for the managerial career track question. Explain.

A hypothetical household survey collects information about the person's education, in years, and occupation, according to the following list of coded categories: 1 *teacher*(10), 2 *doctor*(12), 3 *policeman*(10), 4 *farmer*(0), 5 *musician*(0), 6 *mathematician*(12), 7 *salesman*(3), 8 *singer*(0), 9 *construction worker*(3), 10 *programmer*(8), 11 *bank clerk*(8), 12 *truck driver*(3), 13 *bus driver*(3), 97 *all other occupations*(0). The numbers in parentheses indicate an expert estimate of the minimal number of years of education to be employed in such an occupation.

Write a validation expression that validates the occupation (variable name *occup*) on the basis of years of education (variable name *educat*). Explain.

In a hypothetical corruption survey respondents are asked how much they trust each of the 10 organizations A, B, C, D, E, F, G, H, I, J.

The responses are recorded using the following scale:

1. Don't trust at all (0)
2. Don't trust (1)
3. Neutral (2)
4. Somewhat trust (3)
5. Trust fully (4)

The answers are then aggregated with weights shown in parentheses above. Hence the minimal number of trust points is $10 \times 0 = 0$ and maximum is $10 \times 4 = 40$ points.

Suppose the next section of the questionnaire must be administered to respondents that indicated 15 or less trust points. Write an enablement condition for this next section. Explain how it works.