Survey Solutions CAPI for surveys/censuses

Nadi, Fiji

Survey Solutions: Validation

Sergiy Radyakin
sradyakin@worldbank.org

Development Data Group (DECDG),
The World Bank

March 27-31, 2017

Data validation

- By nature, socio-economic data is commonly bound by some limits, and answers to questions may be related to each other.
- Categorical selections limit values to only the valid ones.
- Still one may need to see how different variables agree with each other.

Data validation

- Validation rules are attached to variables.
- Validation consists of up to 10 rules, each consisting of one logical expression and one error message.
- Logical expression must be written in C# language.
- Expression may refer to the current variable, or also utilize other variables in the questionnaire.
- Error message is shown when the condition evaluates to false.

Logical operators

The following logical operators can be used in validation conditions:

Logical operator	Description	Type of questions
>	Greater than	Numeric and Categorical one answer
<	Less than	Numeric and Categorical one answer
==	Equal to	Text, Numeric and Categorical one answer
!=	Not equal to	Text, Numeric and Categorical one answer
>=	Greater than or equal to	Numeric and Categorical one answer
<=	Less than or equal to	Numeric and Categorical one answer
&&	And	
||	Or	
!	Not	

- Use parentheses as necessary in complex conditions;
- To refer to the answer of a question, specify the variable name corresponding to that question, like so:

```
age
```

- To compare strings, enclose string constants in quotes, like so:

```
city=="New York"
```

Examples

- Verify the numeric variable age is non-negative:

```
age>=0
```

- Verify the numeric variable age is under 120:

```
age<=120)
```

- Verify the numeric variable age is between 0 and 120:

```
(age>=0) && (age<=120)
```

- Verify that the expenditures is no more than income (both numeric variables):

```
expendit<=income
```

- Verify that numeric variable *age* is above 22 if education category (variable *educat*) is 4 or 5:

```
!((educat==4 || educat==5)&&(age<=22))
```

- Validation expressions may be large and complicated. Use parentheses when in doubt or check the order of evaluation.

Keyword self

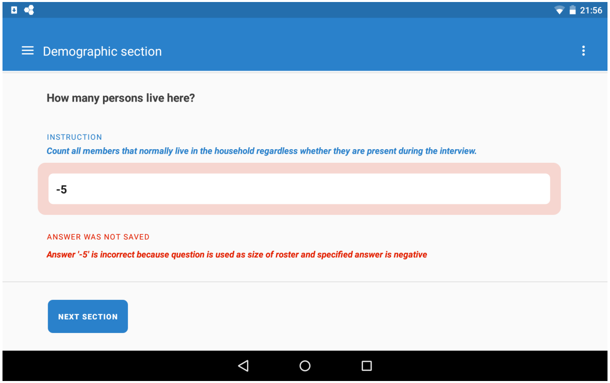
- Use the special variable `self` to refer to the value of the question itself in validation conditions
- (for example for percentages):

```
(self>=0) && (self<=100)
```

- this makes *cloning* the questions easier.

Soft vs Hard validation

- Validation is “soft”;
- “Hard” validation only for the text questions with a pattern, numeric questions triggering rosters, numeric integer questions.



How many persons live here?

INSTRUCTION

Count all members that normally live in the household regardless whether they are present during the interview.

-5

ANSWER WAS NOT SAVED

Answer '-5' is incorrect because question is used as size of roster and specified answer is negative

Data validation

- C# language provides powerful tools for writing complex validations.
- Survey Solutions' built-in functions greatly simplify some conditions, and will be covered later on in this course.

Data validation

- The error message must be clear to the interviewer. Make sure all error messages programmed in the questionnaire are explained to the interviewers during their training.
- When checking coherence of several variables (for example age and year of birth), one may not be sure which one is wrong. The message should allow for the possibility that the error is in the other variable.
- Make sure validation message is consistent with what is being checked. If you revise a validation condition, make sure to check if the corresponding message must also be revised.

Mandatory questions

- Validation validates answers;
- No answer, no validation is run;
- One cannot make a question erroneous if it is not answered;
- Hence no validation condition will make its question mandatory!

Static text

- Static text is an exception;
- Always validated;
- Can show an error, even though doesn't carry an answer.
- Useful for end-of-section and end-of-questionnaire validations.

Single-answer enforcement

- Instructions like "If child is under 5, enter code 99";
- If there is only one valid answer, often it is best to skip the question;
- If necessary, replace the missing value later with a desired constant.

Final

- prevent data entry errors whenever possible, rather than using validation: use categorical, date, formatted text, etc questions.;
- utilize validation for other checks;
- test validation: look for false positives, false negatives;
- prioritize error messages (do you always want to show all 10)?