

REDCAP HOW TO: BRANCHING LOGIC



This tutorial was created by:

Bas de Veer

Institute of Translational Health Sciences

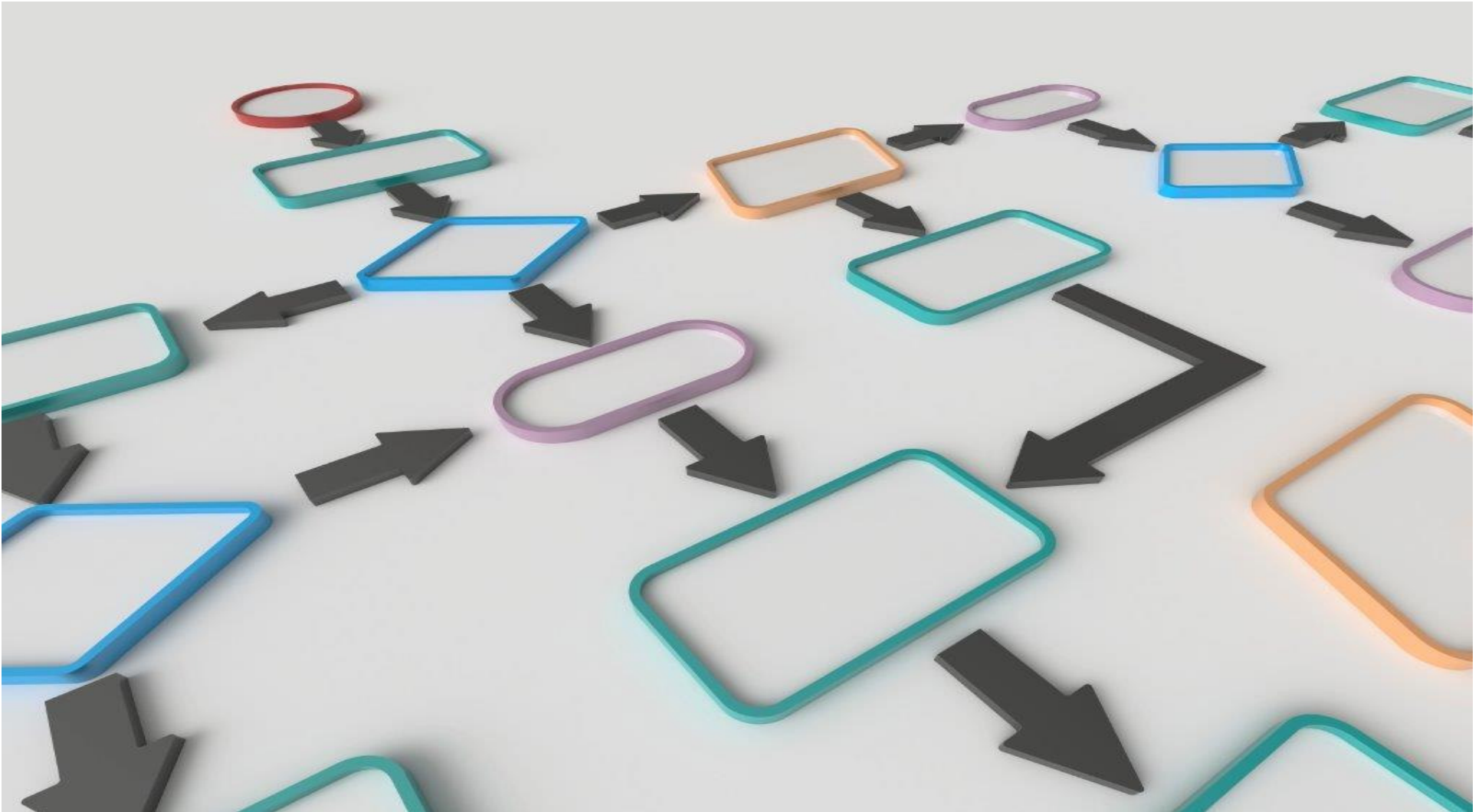
University of Washington

LEARNING OBJECTIVES

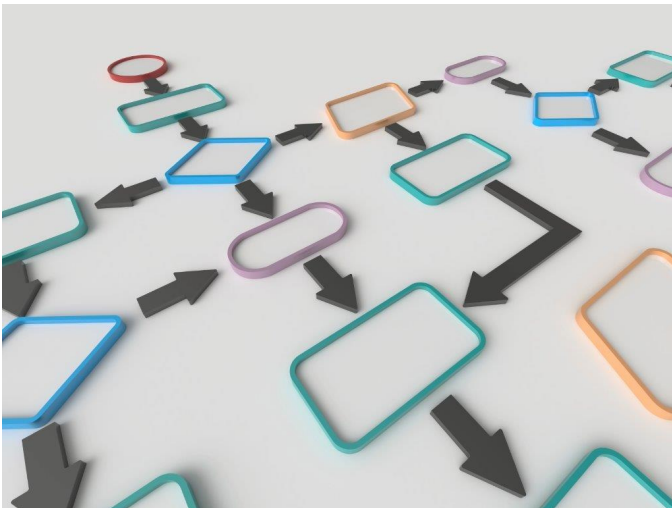
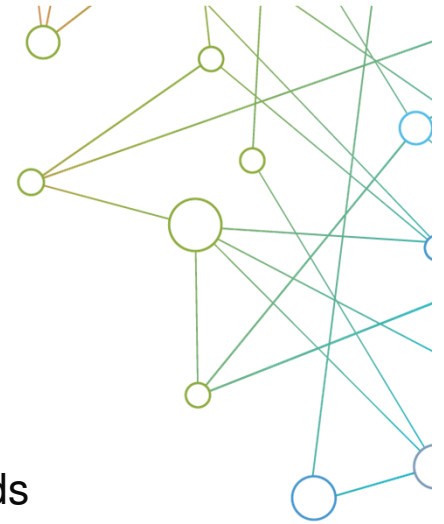
- Learn how to leverage the Data Dictionary
- Branching logic:
 - Basics
 - Advanced techniques
 - Best practices



BRANCHING LOGIC



What's Branching Logic?



**Adds flexibility to
your instruments**

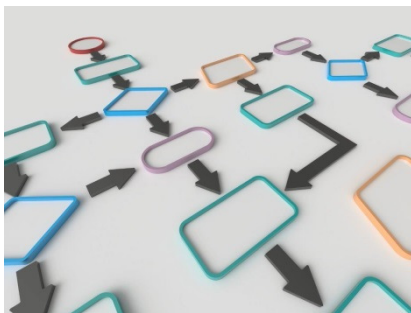
- ▶ The art of hiding or showing fields
- ▶ Based on previously entered values
- ▶ Limited to a single project
- ▶ Drag and Drop Method:
 - ❑ The “easy” way
 - ❑ Reduced flexibility
- ▶ Advanced Syntax Method:
 - ❑ The “hard” way
 - ❑ Programming experience helps
 - ❑ Allows you to get creative
 - ❑ Can be used both in the online interface and in the data dictionary

Reversing your thought process



- ▶ Logic located in the “source” question
- ▶ Directs you to “skip” to a question down the line
- ▶ Very linear
- ▶ Hard to account for complex logic
- ▶ Mostly found on paper forms

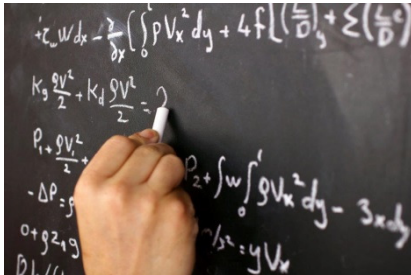
Classic skip logic



- ▶ Logic located in the “destination” question
- ▶ Hides or shows the question
- ▶ Allows for multiple logic pathways (e.g. Inclusion criteria)
- ▶ Can get very complex
- ▶ Extensive used in REDCap

Branching logic in REDCap

The language of logic in REDCap



- And Statement
 - ❑ **and**
- Or statement
 - ❑ **or**
- Equals statement
 - ❑ **=**
- Not statement
 - ❑ **<>**

Logic



- Standard math
 - ❑ **+, -, /, ***
- Comparing
 - ❑ **>, <, >=, <=**
- Order of operation
 - ❑ **Parentheses ()**

Math

Building a basic logic statement

```
if (!identityAssertion) {  
    String password = null;  
    try {  
        password = database.getUserPassword(username);  
    } catch (NotFoundException e) {  
        password = null; // password not found  
    }  
    if (password != null) {  
        if (password.equals(passwordValue)) {  
            throw new LoginException("Authentication Failed: User = " + username + ", Password = " + passwordValue + ", Message = " + passwordValue + ".");  
        }  
    }  
}  
// anonymous login - let it through  
System.out.println("empty username");  
loginSucceeded = true;  
principalsForSubject.add(new MSilverImp(username));  
addGroupsForSubject(username);  
return loginSucceeded;  
}
```

- ▶ Define the **variable/field**
 - ❑ Brackets (e.g. [field1])
 - ❑ Use the variable name instead of the field label
 - ❑ Brackets are also used for event definition (e.g. [event_arm_1][field1])
- ▶ Put in an **operator**
 - ❑ e.g. = , <>, >=, <=, >, <
- ▶ Declare your **comparison value**
 - ❑ Can be a "hard" value like a number or a date
 - ❑ Can be another variable
 - ❑ Use of single quotes is optional
 - Double quotes allowed, not recommended

[age_of_child] >= '18'

Branching logic example 1

Simple statement (**single/radio**)

Basic statements

Simple (Single/Checkbox)

And

Or

Not

Empty

Complex statements



Logic context

- ▶ You want to ask the question:
Is the participant on Medicaid?
But this is only relevant for people who are 65 or over.

Needed elements

- ▶ Variable: `[age]`
- ▶ Operator: `>=`
- ▶ Comparison value: `'65'`

Branching logic statement

- ▶ `[age] >= '65'`

Branching logic example 2

Simple statement (checkbox)

Basic statements

Simple (Single/Checkbox)

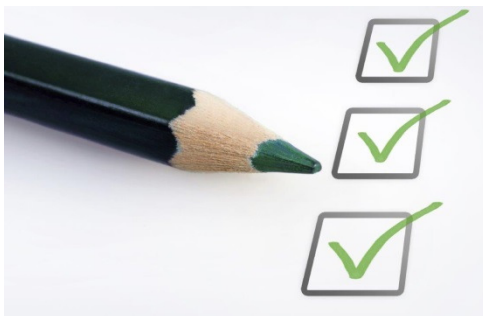
And

Or

Not

Empty

Complex statements



Logic context

- ▶ You want to ask the question:
Did the participant get vaccinated for malaria?
But this is only relevant for people who recently went to a country where malaria is prevalent .

Needed elements

- ▶ Variable: `[country(3)]`
- ▶ Operator: `=`
- ▶ Comparison value: `'1'`
 - ▶ for checkboxes, 1 = “checked”; 0 = “unchecked”

Branching logic statement

- ▶ `[country(3)] = '1'`

Branching logic example 3

And statement

Basic statements

Simple (Single/Checkbox)

And

Or

Not

Empty

Complex statements



Logic context

- ▶ You want to ask the question:
How did the medication affect your allergy symptoms?
But this is only relevant for people who have allergy symptoms and take the medication.

Needed elements

- ▶ Variables: [allergy] and [symptoms]
- ▶ Operator: =
- ▶ Comparison value: '1'
 - ▶ In this example, allergy and symptoms are Yes/No questions (1 = "Yes"; 0 = "No").

Branching logic statement

- ▶ [allergy] = '1' and [symptoms] = '1'

Branching logic example 4

Or statement

Basic statements

Simple (Single/Checkbox)

And

Or

Not

Empty

Complex statements



Logic context

- ▶ You display a warning to warn for ineligibility when a participant is either a smoker or a drug user.

Needed elements

- ▶ Variables: [smoker] and [drugs]
- ▶ Operator: =
- ▶ Comparison value: '1'

Branching logic statement

- ▶ [smoker] = '1' or [drugs] = '1'

Branching logic example 5

Not statement

Basic statements

Simple (Single/Checkbox)

And

Or

Not

Empty

Complex statements



Logic context

- ▶ You want to ask the question:
Have you ever had heart attack-like symptoms?
But you only want to ask this when people have NOT had a heart attack before.

Needed elements

- ▶ Variable: `[heart_attack]`
- ▶ Operator: `<>`
- ▶ Comparison value: `'1'`

Branching logic statement

- ▶ `[heart_attack] <> '1'`

Branching logic example 6

Empty statement

Basic statements

Simple (Single/Checkbox)

And

Or

Not

Empty

Complex statements



Logic context

- ▶ You want to display a warning when the date of birth field has not been filled out. But the warning needs to disappear if the date of birth field has a value in it.

Needed elements

- ▶ Variable: `[dob]`
- ▶ Operator: `=`
- ▶ Comparison value: `'` (two single quotes)

Branching logic statement

- ▶ `[dob] = ''`

Branching logic example 7

Complex statement

Basic statements

Complex statements

Date differential

Sum

Contains

If

Nested If

```
if (!isIdentityAssertion) {
    String passwordHint = null;
    try {
        passwordHint = database.getUserPassword(username);
    } catch (NotFoundException shouldNotHappen) {}
    catch (NotHavePasswordException shouldNotHappen) {
        String passwordHint = null;
        if (passwordHint != null) {
            passwordHint.equals(passwordHint);
        }
        if (passwordHint != null) {
            throw new AuthenticationFailedException(
                "Authentication failed: User " + username + " had password: " +
                "Have " + passwordHint + ", Hint " + passwordHint);
        }
    }
} else {
    // anonymous login - let it through?
    System.out.println("empty username");
    loginSucceeded = true;
    principalsForSubject.add(new MSUserImp(username));
    addGroupsForSubject(username);
    return loginSucceeded;
}
```

Logic context

- ▶ You want to display a question about mid life crisis, but only when the participant is outside of the standard mid life crisis age range and the date of birth field has been filled out.

Needed elements

- ▶ Variable: [age], [dob]
- ▶ Operator: >=, <=, <>
- ▶ Comparison value: '0','39','63','120',' (two single quotes)

Branching logic statement

- ▶ ([age] >= '0' and [age] <= '39' and [dob] <> '') or ([age] >= '63' and [age] <= '120' and [dob] <> '')
- ▶ ((([age] >= '0' and [age] <= '39') or ([age] >= '63' and [age] <= '120')) and [dob] <> '')
- ▶ Note: Both of these are correct. There are always multiple ways of coding.

Branching logic example 8

Date differential

Basic statements

Complex statements

Date differential

Sum

Contains

If

Nested If



Logic context

- ▶ You want to display a question about how somebody's heart attack affected their work life, but only if they had the heart attack when they were younger than 65.

Needed elements

- ▶ Variable: `[dob]`, `[date_of_attack]`
- ▶ Function: `datediff([date1],[date2],"units","format")`
- ▶ Operator: `<=`
- ▶ Comparison value: `'65'`

Branching logic statement

- ▶ `(datediff([dob],[date_of_attack],'y','mdy'))
<='65'`

Branching logic example 9

Sum statement

Basic statements

Complex statements

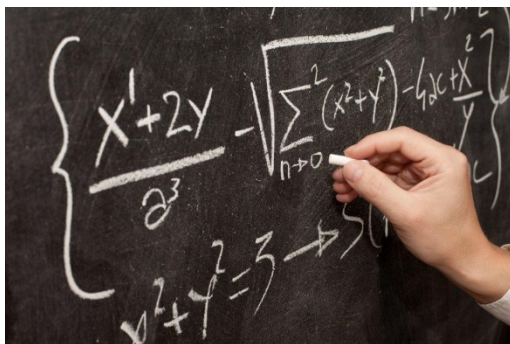
Date differential

Sum

Contains

If

Nested If



Logic context

- ▶ You want to ask a question about depression when the total score of a depression scoring tool reaches above a certain value.

Needed elements

- ▶ Variable: [depr1], [depr2], [depr3]
- ▶ Function: **sum()**
- ▶ Operator: **>=**
- ▶ Comparison value: '4'

Branching logic statement

- ▶ **(sum([depr1],[depr2],[depr3]))>='4'**

Branching logic example 10

Contains statement

Basic statements

Complex statements

Date differential

Sum

Contains

If

Nested If



Logic context

- ▶ You want to ask for a survey respondents private email if they provide an university email address in their initial response.

Needed elements

- ▶ Variable: `[email]`
- ▶ Function: `contains()`
- ▶ Comparison value: `' .edu'`

Branching logic statement

- ▶ `contains([email], '.edu')`

Branching logic example 11

If statement

Basic statements

Complex statements

Date differential

Sum

Contains

If

Nested If



Logic context

- ▶ You want to ask a question about depression when the total score of a depression scoring tool reaches above a certain value. However, you've built in a "prefer not to answer" response for the first question that you coded as '99'. You need to filter out this response from your logic.

Needed elements

- ▶ Variable: [depr1], [depr2], [depr3]
- ▶ Function: if(), sum()
- ▶ Operator: >=, =
- ▶ Comparison value: '4', '99', '0'

Branching logic statement

- ▶ `(sum(
(if([depr1]='99','0',[depr1]))
,[depr2],[depr3]))>='4'`

Branching logic example 12

Nested if statement

Basic statements

Complex statements

Date differential

Sum

Contains

If

Nested If



Logic context

- ▶ You want to ask a question about depression when the total score of a depression scoring tool reaches above a certain value. However, you've built in a "prefer not to answer" response for the first question that you coded as '99'. You've also added an option for "unknown" (98). You need to filter out these responses from your logic.

Needed elements

- ▶ Variable: [depr1], [depr2], [depr3]
- ▶ Function: if(), sum()
- ▶ Operator: >=, =
- ▶ Comparison value: '4', '99', '98', '0'

Branching logic statement

- ▶ (sum(
(if([depr1]='99','0',
(if([depr1]='98','0',[depr1]))))
,[depr2],[depr3]))>='4'

Diagram your logic

**For complex logic, draw out
your logic in a flow chart.**

**This will help you visualize
your logic flow and allow you
to troubleshoot any issues.**



Institute of Translational Health Sciences
Accelerating Research. Improving Health.