



# REDCap Training 301

## Advanced REDCap Features

*Sui Tsang*

*REDCap@Yale Team*

*5/16/2024*



YALE UNIVERSITY  
SCHOOL OF  
MEDICINE

# REDCap 301 Learning Objectives

- Understand *Form Display Logic*
- Identify *Special Functions* that are commonly used in *calculations* and *branching logic*
- Know how *Action Tags* can be used to make your data collection more dynamic
- Learn how to incorporate *Smart Variables* to customize questions, forms, and communications.
- Be able to build and display a *Project Dashboard*
- Learn how to manage *Randomization* in REDCap

# Form Display Logic

Data Collection Instrument	Baseline Baseline Visit	FU 45d Follow Up Visit - 45 days	FU 6m Follow Up Visit - 6 Months
Patient Screening Form	<input type="radio"/>		
Caregiver Screening Form	<input type="radio"/>		
Patient Contact	<input type="radio"/>		
Caregiver Contact	<input type="radio"/>		
Patient Call Log			
Patient Scheduled Interview Date			
Callahan Screener			
Patient Interview Start	<input type="radio"/>		
Patient Baseline Characteristics	<input type="radio"/>		

Data Collection Instrument	Baseline Baseline Visit	FU 45d Follow Up Visit - 45 days	FU 6m Follow Up Visit - 6 Months
Patient Screening Form	<input checked="" type="radio"/>		
Caregiver Screening Form	<input type="radio"/>		
Patient Contact	<input type="radio"/>		
Caregiver Contact	<input type="radio"/>		
Patient Call Log		<input type="radio"/>	<input type="radio"/>
Patient Scheduled Interview Date		<input type="radio"/>	<input type="radio"/>
Callahan Screener		<input type="radio"/>	<input type="radio"/>
Patient Interview Start	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Patient Baseline Characteristics	<input type="radio"/>		

Data Collection Instrument	Baseline Baseline Visit	FU 45d Follow Up Visit - 45 days	FU 6m Follow Up Visit - 6 Months
Patient Screening Form	<input checked="" type="radio"/>		
Caregiver Screening Form	<input checked="" type="radio"/>		
Patient Contact	<input type="radio"/>		
Caregiver Contact	<input type="radio"/>		
Patient Call Log		<input type="radio"/>	<input type="radio"/>
Patient Scheduled Interview Date		<input type="radio"/>	<input type="radio"/>
Callahan Screener		<input type="radio"/>	<input type="radio"/>
Patient Interview Start	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Patient Baseline Characteristics	<input type="radio"/>		

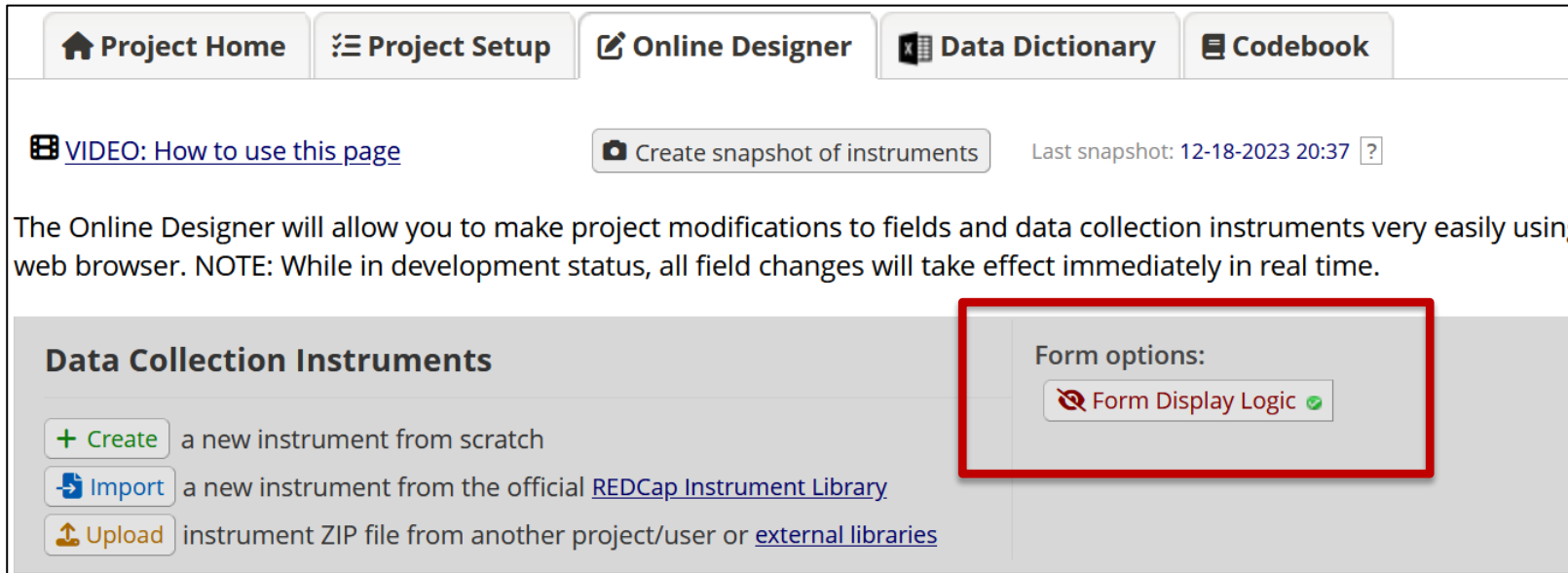
New Record – only Patient Screening Form is available for data entry

Caregiver consent=Yes – all patient forms available for data entry

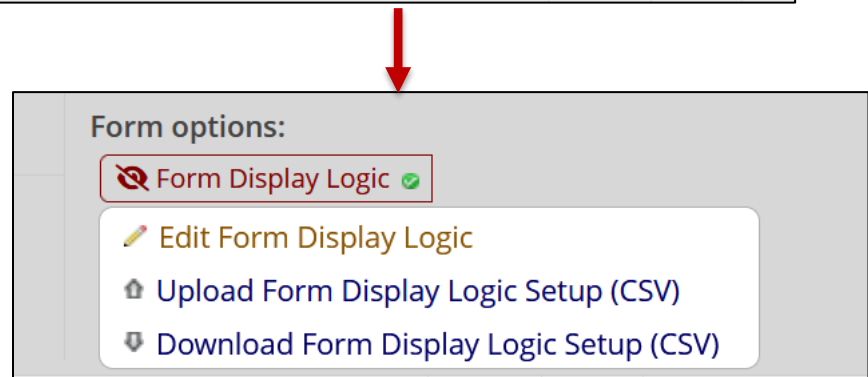
Caregiver consent=Yes – all caregiver forms available for data entry

# Form Display Logics

## Online Designer -> Form Display Logic



The screenshot shows the REDCap Online Designer interface. At the top, there are navigation tabs: Project Home, Project Setup, Online Designer (selected), Data Dictionary, and Codebook. Below the tabs, there is a video link 'VIDEO: How to use this page', a 'Create snapshot of instruments' button, and a 'Last snapshot: 12-18-2023 20:37' indicator. A paragraph explains that the Online Designer allows for easy modifications to fields and data collection instruments in a web browser, with a note that changes take effect immediately in real time. The 'Data Collection Instruments' section is visible, with options to 'Create', 'Import', and 'Upload' instruments. A red box highlights the 'Form options' section, which contains a 'Form Display Logic' button with a green checkmark.



A red arrow points from the 'Form options' section in the previous screenshot to this close-up view. The 'Form options' section is titled 'Form options:' and contains a 'Form Display Logic' button with a green checkmark. Below this button, there is a white box containing three actions: 'Edit Form Display Logic' (with a pencil icon), 'Upload Form Display Logic Setup (CSV)' (with an upload icon), and 'Download Form Display Logic Setup (CSV)' (with a download icon).



Form Display Logic is an advanced feature that provides a way to use conditional logic to disable specific data entry forms that are displayed on the Record Status Dashboard, Record Home Page, or the form list on the left-hand menu. You might think of it as 'form-level branching logic'. Form Display Logic can be very useful if you wish to prevent users from entering data on a specific form or event until certain conditions have been met. The forms will still be displayed on the page, but they will be disabled in order to prevent users from accessing them. Below you may define as many conditions as you want. **A form may be selected in multiple conditions, but if so, please note that the form will be enabled if at least one of the conditions is met.** The Form Display Logic does not impact data imports but only operates in the data entry user interface to enable/disable forms. Additionally, Form Display Logic is not utilized by the Survey Queue at all but can affect the behavior of the Survey Auto-Continue feature if the checkbox for it is enabled below. LIMITATION: Please note that the conditional logic used below will be evaluated at the record level and not within the context of an event or a repeating instance, which means that it is not possible to use relative instance or relative event Smart Variables - i.e., those with the name 'current', 'next', or 'previous', such as [next-instance] or [previous-event-name].

Optional Settings:

- Keep forms enabled if they contain data**  
Only disable empty forms (those with a gray form status icon).
- Hide forms that are disabled**  
All forms that are disabled will also be hidden (not visible) on the Data Collection menu and on the Record Home Page.

▼ Condition 1:



Keep the following forms enabled...

**[All Events]**

- Patient Screening Form [All Events]
- Caregiver Screening Form [All Events]
- Patient Contact [All Events]
- Caregiver Contact [All Events]
- Patient Interview Start [All Events]
- Patient Baseline Characteristics [All Events]

(You may select multiple forms by clicking them)

[View list of selected forms](#)

...when the logic below is **TRUE**.

[baseline\_arm\_1][pt\_cg\_approach]=1

e.g., [enrollment\_arm\_1][age] > 30

[How to use this](#)

Save

Cancel

# *Form Display Logics*

QUESTIONS?

# Special Functions for Calculation, Branching Logic

## √\* Special Functions

### √\* Special Functions

[View text on separate page](#)

#### List of functions that can be used in Branching Logic, Calculations, Report filtering, Data Quality Module, Automated Survey Invitations, etc.

REDCap logic can be used in a variety of places, such as Branching Logic, Calculations, Report filtering, Data Quality Module, Automated Survey Invitations, and more. Special functions can be used in the logic, if desired. A complete list of ALL available functions is listed below. Listed below are some examples of common use cases where these functions might be used.

**NOTICE:** Please be advised that it is not possible to pipe the choice label of a multiple choice field into any of the special functions listed below. In other words, you cannot use the ':label' option, such as [my\_field:label], to output the text label into a function. **The special functions only utilize the value of a field, never its label.**

#### Practical examples for common use cases

1. Calculate the number of days separating today's date and a date/datetime field's value in the past or future.

```
datediff([date1], 'today', 'd')
```

2. Calculate a person's age based on date of birth.

```
rounddown(datediff([date_of_birth], 'today', 'y'))
```

3. Calculate a person's BMI (in metric units of 'cm' and 'kg') and rounding to the first decimal place.

```
round(([weight]*10000)/((([height])^(2))), 1)
```

4. Calculate a person's BMI (in English units of 'lb' and 'in') and rounding to the first decimal place.

```
round(([weight]/((([height])^(2))*703), 1)
```

5. Remove a prefix and dash from the beginning of a record name (e.g., convert '4890-2318' to '2318').

```
mid([record-name], find('-', [record-name])+1, length([record-name])-find('-', [record-name])+1)
```

6. Convert a person's first and last name into a username-looking format (e.g., convert 'John' and 'Doe' to 'john\_doe'). We may want to trim the values just in case there were spaces accidentally entered.

```
lower( concat( trim([first_name]), '_', trim([last_name]) ) )
```

7. Add leading zeros to an integer in which the number near the end of the equation represents the maximum length of the result after



[REDCap FAQ](#)  
Calculated Field






# Special Functions

## Example 1 – Date Functions

year([date\_field])

month([date\_field])

day([date\_field])

<b>Interview Date</b> <i>* must provide value</i>	[pt_int_start_date]	 <input type="text" value="05-10-2024"/>  Today M-D-Y
<b>Interview Year</b>	year([pt_int_start_date])	 <input type="text" value="2024"/> <a href="#">View equation</a>
<b>Interview Month</b>	month([pt_int_start_date])	 <input type="text" value="05"/> <a href="#">View equation</a>
<b>Interview Day</b>	day([pt_int_start_date])	 <input type="text" value="10"/> <a href="#">View equation</a>



# Special Functions

## Example 2 – Text Functions

1. Joins the text from multiple text strings with a separator – `concat_ws`

<b>When is the best time of day to call?</b>	<input checked="" type="checkbox"/> Morning <input checked="" type="checkbox"/> Afternoon <input type="checkbox"/> Evening
<b>Patient Best Time to Call</b>	Morning   Afternoon <a href="#">View equation</a>

### Action Tags / Field Annotation (optional)

```
@CALCTEXT(concat_ws(" | ", (if([pt_besttime(1)]=1, "Morning ", "")),  
(if([pt_besttime(2)]=1, " Afternoon ", "")), (if([pt_besttime(3)]=1, " Evening", ""))))
```

Syntax:

`concat_ws` function: `concat_ws (separator, text, text, ...)`

# Special Functions

## Example 2 – Text Functions

### 2. Extract a text string from a text field – mid, find

Participant ID	YAL-034
Extract last three numbers of record ID	<input type="text" value="034"/> <a href="#">View equation</a>
Convert the last three numbers of record ID to number	<input type="text" value="34"/> <a href="#">View equation</a>

#### Action Tags / Field Annotation (optional)

```
@CALCTEXT(mid([partid], (find('-', [partid])+1), 3))
```

#### Calculation Equation [How do I format the equation?](#)

```
[extract_number]*1
```

#### Syntax:

mid function: mid (text, start position, number of characters)

find function: find (needle, haystack)

# Special Functions: Calculations & Branching Logic

QUESTIONS?

# Action Tags



REDCap FAQ  
Action Tags

Action tags allow you to modify fields in very specific ways. Once applied, a corresponding action is performed.

In REDCap, action tags begin with the '@' sign and are placed inside a field's "Action Tags/Field Annotation" box.

**Field Label**

Test default value

**Choices (one choice per line)** [Copy existing choices](#)

1, Yes  
2, No

[How do I manually code the choices?](#)

**Action Tags / Field Annotation** (optional)

@DEFAULT='2'

Learn about [@ Action Tags](#) or [using Field Annotation](#)

+ Adding new Participant ID code 1111

Event Name: **Flu Clinic (Arm 1: U19)**

Participant ID code	1111
test secondary unique field	<input type="text"/> (Secondary unique NOTE: Modifying th corresponding inst Repeating Instance
Test default value	<input type="radio"/> Yes <input checked="" type="radio"/> No

# Action Tags

## Example 1 - @IF

- Allows action tags to be set based on conditional logic provided inside an @IF() function.
- Syntax:

@IF(CONDITION, ACTION TAGS if condition is TRUE, ACTION TAGS if condition is FALSE)

Example: @IF([yes\_no] = '1', @HIDDEN, @HIDECHOICE='3' @READ-ONLY)

Implement when  
[yes\_no]='1'

Implement when  
[yes\_no]<>'1'

- If you wish not to output action tags for a certain condition, set it with a pair of apostrophes/quotes as a placeholder

Example: @IF([my\_radio]='1', @READONLY, "")

***Note: The conditional logic is evaluated only when the survey page or data entry form initially loads; thus, action tag will not be evaluated in real-time as data is entered.***

# Action Tags

## Example 2 - @CALCTEXT and @CALCDATE

### @CALCTEXT

Evaluates logic that is provided inside a @CALCTEXT() function for Text Box fields \*only\* and outputs the result as text, typically performed with an if(x,y,z) function - e.g.,

```
@CALCTEXT(if([gender]='1', 'male', 'female'))
```



### @CALCDATE

Performs a date calculation by adding or subtracting a specified amount of time from a specified date or datetime field and then provides the result as a date or datetime value - e.g.,

```
@CALCDATE([visit_date], 7, 'd')
```



# Action Tags

## Example 2 - @CALCTEXT and @CALCDATE

- On the Interview End Form, a prompt is shown on the form for the interviewer to tell the patient when they will be contacted again for the next interview.

### Prompt at 45-day visit:

Event: **FU 45d**

Participant ID

YAL-034

**PROMPT:** Those are all the questions I have for you today. Thank you very much for taking the time to complete this interview! We will be contacting you again in four and a half months, around 10-30-2023 - 11-15-2023.

### Prompt at 6-month visit:

Event: **FU 6m**

Participant ID

YAL-034

**PROMPT:** Those are all the questions I have for you today. Thank you very much for taking the time to complete this interview! We will be contacting you again in six months, around 05-02-2024 - 05-18-2024.

# Action Tags

## Example 2 - @CALCDATE and @CALCTEXT

[ calc_fu_interval ]	Calctext fu interval for piping	text Field Annotation: @HIDDEN @CALCTEXT(if([event-name]='fu_45d_arm_1', 'four and a half months', If([event-name]='fu_6m_arm_1', 'six months', '')))
[ calc_6m_start ]	6 month interview window start date	text (date_mdy) Field Annotation: @HIDDEN @CALCDATE([baseline_arm_1] [random_date], 179, 'd')
[ calc_6m_end ]	6 month interview window end date	text (date_mdy) Field Annotation: @HIDDEN @CALCDATE([baseline_arm_1] [random_date], 195, 'd')
[ calc_12m_start ]	12 month interview window start date	text (date_mdy) Field Annotation: @HIDDEN @CALCDATE([baseline_arm_1] [random_date], 364, 'd')
[ calc_12m_end ]	12 month interview window end date	text (date_mdy) Field Annotation: @HIDDEN @CALCDATE([baseline_arm_1] [random_date], 380, 'd')
[ calc_int_window_start ]	Window start date for piping	text (date_mdy) Field Annotation: @HIDDEN @CALCTEXT(if([event-name]='fu_45d_arm_1', [calc_6m_start], if([event-name]='fu_6m_arm_1', [calc_12m_start], '')))
[ calc_int_window_end ]	Window end date for piping	text (date_mdy) Field Annotation: @HIDDEN @CALCTEXT(if([event-name]='fu_45d_arm_1', [calc_6m_end], if([event-name]='fu_6m_arm_1', [calc_12m_end], '')))
[ pt_int_end_fu ] Show the field ONLY if: [event-name]='fu_45d_arm_1' or [event-name]='fu_6m_arm_1'	PROMPT: Those are all the questions I have for you today. Thank you very much for taking the time to complete this interview! We will be contacting you again in [calc_fu_interval], around [calc_int_window_start] - [calc_int_window_end].	descriptive





# Action Tags

QUESTIONS?

# Smart Variables

Allow reference information other than data fields (e.g., event, repeat instance, DAG or users)

[⚡] Smart Variables

Smart Variables		Example of Usage	
Name of Smart Variable	Description	Example input	Example output
<b>User</b>			
[user-name]	The current user's REDCap username.	[user-name]	jane_doe
[user-fullname]	The current user's first and last name (as listed on their My Profile page).	[user-fullname]	Jane Doe
[user-email]	The current user's primary email address (as listed on their My Profile page).	[user-email]	jane.doe@example.edu
[user-dag-name]	The Data Access Group (the unique group name) to which the current user belongs (blank if not in a DAG).	[user-dag-name]	vanderbilt_group
[user-dag-id]	The group ID number of the Data Access Group to which the current user belongs (blank if not in a DAG).	[user-dag-id]	324
[user-dag-label]	The name/label of the Data Access Group to which the current user belongs (blank if not in a DAG).	[user-dag-label]	Vanderbilt Group
<b>Record</b>			
[record-name]	The record name of the current record.	[record-name]	108
[record-dag-name]	The Data Access Group (the unique group name) to which the current record belongs (blank if not in a DAG).	[record-dag-name]	harvard_site
[record-dag-id]	The group ID number of the Data Access Group to which the current record belongs (blank if not in a DAG).	[record-dag-id]	96
[record-dag-label]	The name/label of the Data Access Group to which the current record belongs (blank if not in a DAG).	[record-dag-label]	Harvard Site
<b>Form</b>			
[is-form]	Detects if the current instrument is being viewed as a data entry form (returns 1 for True, 0 for False), as opposed to a survey.	[is-form]	1



# Smart Variables

## Example 1:

Set the option selected at the previous event as the default choice in the current event.

### Choices (one choice per line)

[Copy existing choices](#)

- 1, Amlodipine (Norvasc)
- 2, Diltiazem (Cardizem, Tiazac, etc.)
- 3, Nicardipine (Cardene SR)
- 4, Nifedipine (Procardia, Adalat)
- 5, Verapamil (Calan SR, Verelan)
- 8, Other {mr\_ccb\_oth}

### Action Tags / Field Annotation (optional)

```
@DEFAULT='[previous-event-name][mr_ccb:value]'
```

- *Append ':value' to the variable name inside brackets to pipe the value (not the label) of a multiple-choice field*
- *To pipe the value of a multiple-choice field as the default value inside the @DEFAULT Action Tag, ':value' must be used.*

# Smart Variables

Example 2:

Branching logic based on data access groups -

[record-dag-name]='bwh' or [record-dag-name]='inova'

Example 3:

Branching logic based on user roles -

[user-role-label]='medical monitor' or [user-role-label]='admin' or [user-role-label]='unblinded dcc staff'

# Smart Variables

## Example 4:

## Customize message for alert and notification

**An unanticipated problem has been reported by [record-dag-label].** ← **DAG Name**

Participant ID: [partid]

Event: [event-label] ← **Event Label**

SAE form instance number: [current-instance] ← **Current repeat instance number**

[form-link:safety\_report\_sae\_upirso:Click here to open the form] ← **Form link**

This event is serious, unexpected and related to study participation. This event must be reviewed by the Medical Monitor immediately.

Per the protocol: If the Medical Safety Monitor believes that a finding or trend suggests a threat to the safety of study participants, the Principal Investigator and chair of the DSMB will be notified and the DSMB convened as necessary. The IRB will be notified of any changes in risk to study participants.

Please note: The report status is [ae\_site\_rpt\_status][current-instance]. If the report is "in process", the project manager will work with the local site to obtain outstanding information while you review the report.

# Smart Variables

## Example 4:

## Customize message for alert and notification

---

**An unanticipated problem has been reported by BWH.**

Participant ID: 15

Event: FU 45d

SAE form instance number: 7

[Click here to open the form](#)

This event is serious, unexpected and related to study participation. This event must be reviewed by the Medical Monitor immediately.

Per the protocol: If the Medical Safety Monitor believes that a finding or trend suggests a threat to the safety of study participants, the Principal Investigator and chair of the DSMB will be notified and the DSMB convened as necessary. The IRB will be notified of any changes in risk to study participants.

Please note: The report status is Ready for review. If the report is "in process", the project manager will work with the local site to obtain outstanding information while you review the report.

# Smart Variables

QUESTIONS?

# Project Dashboard

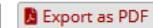
- Project Dashboards are pages with dynamic content that can be added to a project
- They can utilize special Smart Variables called Smart Functions, Smart Tables, and Smart Charts that can perform aggregate mathematical functions, display tables of descriptive statistics, and render various types of charts, respectively



# Project Dashboard

## Project Dashboard Example 1

Displaying recently generated information



**SMART FUNCTIONS** are aggregate mathematical functions that are applied across ALL records in a project.

Smart Functions include **min**, **max**, **mean**, **median**, **sum**, **stdev**, **count**, and **unique**.

This project contains **50 records**. The average age of all participants is **47.38** (stdev=30.93). The median weight is **109** (min: 47, max: 197).

**SMART TABLES** display descriptive statistics for fields with each field as a row in the table.

Smart Tables can be displayed with ALL columns by default:

	Count	Missing	Unique	Min	Max	Mean	Median	StDev	Sum
Height (cm)	50	0	35	133	214	177.22	180	24.64	8,861
Weight (kilograms)	50	0	44	47	197	114.72	109	42.79	5,736
Race	50	0	7						
Gender	50	0	2						

[Export table \(CSV\)](#)

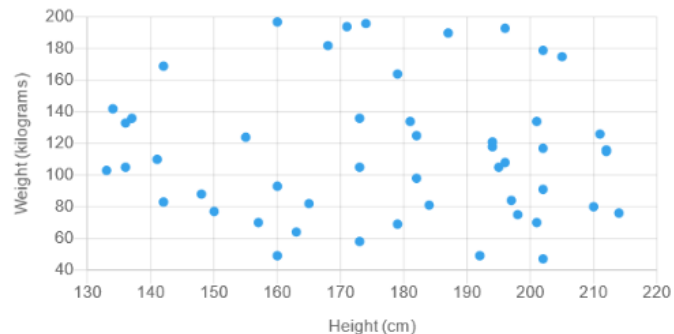
Or with only specified columns:

	Mean	StDev
Age (years)	47.38	30.93
Weight (kilograms)	114.72	42.79
Height (cm)	177.22	24.64

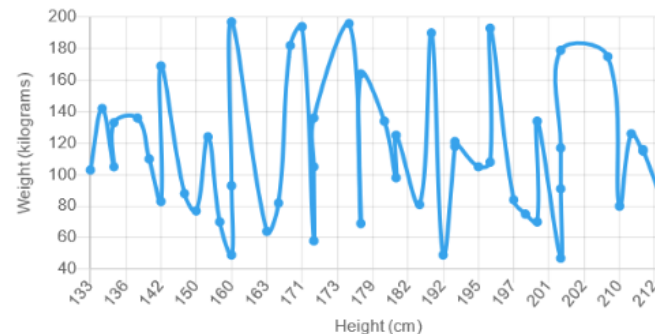
[Export table \(CSV\)](#)

**SMART CHARTS** can be used to display many types of charts for one or more fields in the project.

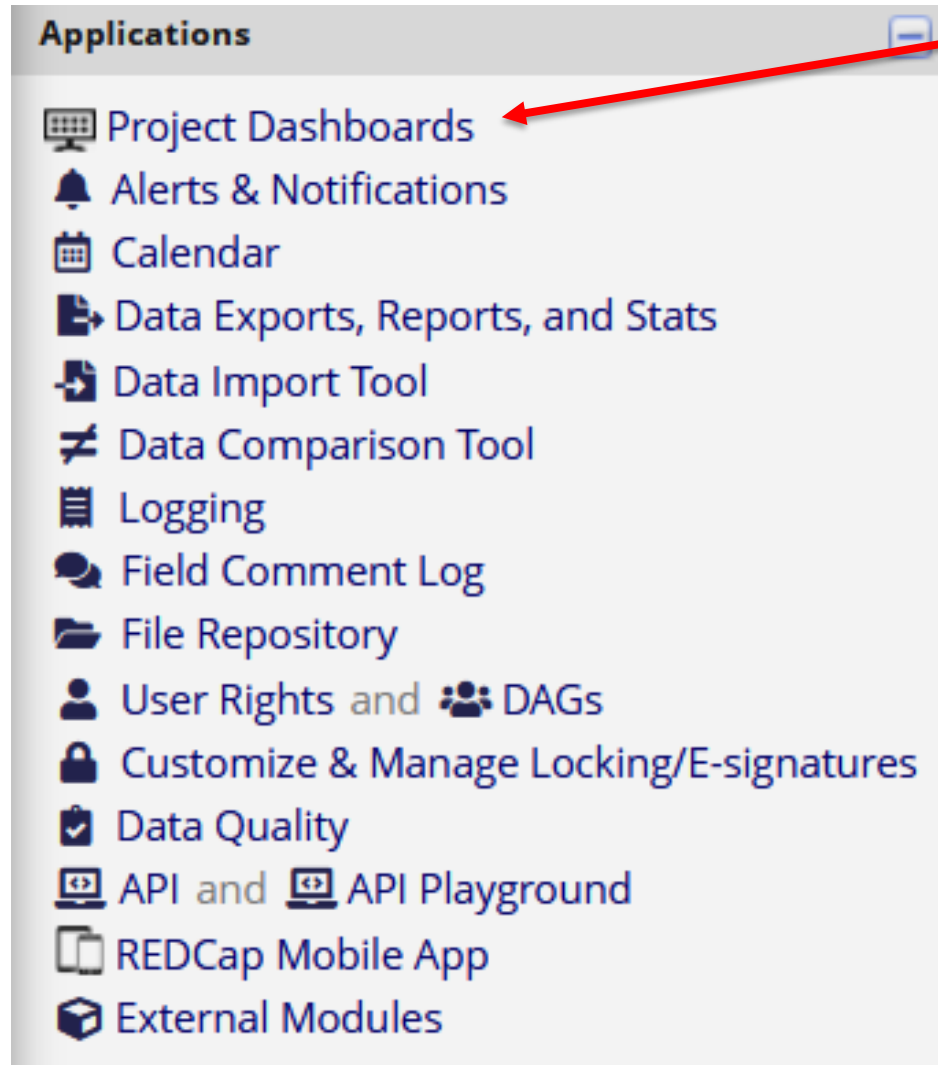
Display a **scatter plot** of two fields (x vs y):



Use **line charts** with two fields (x vs y):



# Project Dashboard



The screenshot shows a sidebar menu titled "Applications" with a list of various tools and features. A red arrow points to the "Project Dashboards" option at the top of the list.

- Project Dashboards
- Alerts & Notifications
- Calendar
- Data Exports, Reports, and Stats
- Data Import Tool
- Data Comparison Tool
- Logging
- Field Comment Log
- File Repository
- User Rights and DAGs
- Customize & Manage Locking/E-signatures
- Data Quality
- API and API Playground
- REDCap Mobile App
- External Modules

**Dashboard title:** Project Dashboard Example 1

**User access:** Choose who sees this dashboard on their left-hand project menu  
 (Note: Users with Setup/Design privileges can still access all dashboards via the Project Dashboards page.)  
 **All users** – OR –  **Custom user access** (Choose specific users, roles, or data access groups who will have access)

**Set as "public":** Enabling this feature below will auto-generate a public link for viewing the dashboard without needing to log in to REDCap.  
 **Dashboard is publicly viewable by anyone with the public link**

**Dashboard content:**  
 Add any static or dynamic text to be displayed, including Smart Functions, Smart Tables, and Smart Charts.

**Dashboard help:**

**SMART FUNCTIONS** are aggregate mathematical functions that are applied across ALL records in a project. Smart Functions include **min**, **max**, **mean**, **median**, **sum**, **stdev**, **count**, and **unique**.  
 This project contains **[aggregate-count:study\_id] records**. The average age of all participants is **[aggregate-mean:age]** (stdev=**[aggregate-stdev:age]**). The median weight is **[aggregate-median:weight]** (min: **[aggregate-min:weight]**, max: **[aggregate-max:weight]**).

**SMART TABLES** display descriptive statistics for fields with each field as a row in the table.

Smart Tables can be displayed with ALL columns by default: <b>[stats-table:height,weight,race,gender]</b>	Or with only specified columns: <b>[stats-table:age,weight,height:mean,stdev]</b>
--	--

**SMART CHARTS** can be used to display many types of charts for one or more fields in the project.

Display a <b>scatter plot</b> of two fields (x vs y): <b>[scatter-plot:height,weight]</b>	Use <b>line charts</b> with two fields (x vs y): <b>[line-chart:height,weight]</b>
Add a third field for grouping (by color): <b>[scatter-plot:height,weight,gender]</b>	Add a third field for grouping (by color): <b>[line-chart:height,weight,gender]</b>

## Wizard for creating Smart Functions, Smart Tables, and Smart Charts

This wizard can help you create a new Smart Function, Smart Table, or Smart Chart that you can then use in a Project Dashboard. Simply make selections using the drop-down lists below regarding which Smart Variable you wish to use, which fields to be utilized in it, and other optional features available. Once your selections have been made, you may click the 'Copy to clipboard' button at the bottom, after which you may paste that generated text into the body of your Project Dashboard.

### Step 1) Choose a Smart Variable to create:

aggregate-count ▾

### Step 2) Choose a field to utilize:

study\_id "Study ID" ▾

+ Add another field

### Step 3) Optional data filtering and other settings:

By default, Smart Functions, Smart Tables, and Smart Charts will utilize all the data from *\*all records\** in the project. However, you may utilize a subset of the data in the project by limiting them to a specific report's data (using a unique report name), to records belonging to one or more DAGs, and/or to data in specific events (if the project is longitudinal).

Filter the data using a report: -- no filtering by report -- ▾

Filter the data using DAGs:

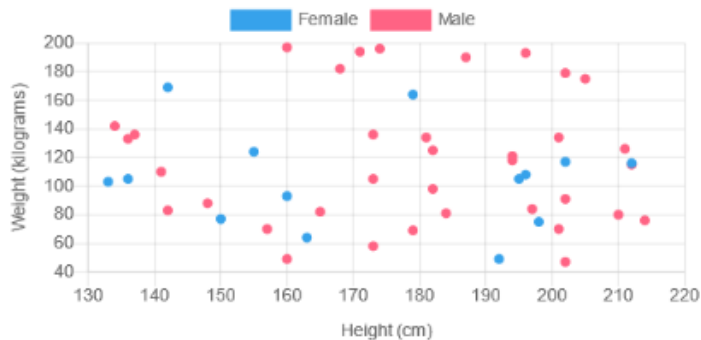
-- no filtering by data access group -- ▲  
Current user's DAG  
Duke ▼

### Step 4) Copy the generated Smart Variable syntax and paste it in your project dashboard:

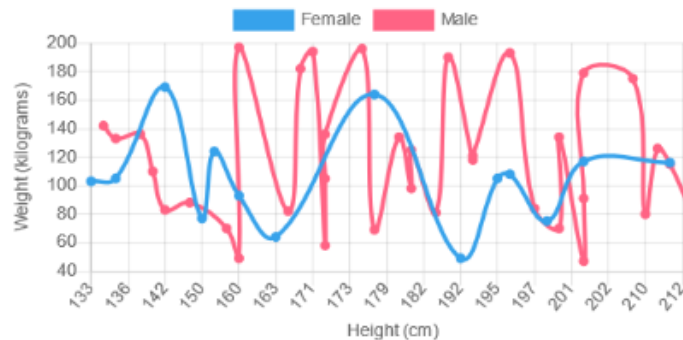
[aggregate-count:study\_id]

Copy to clipboard

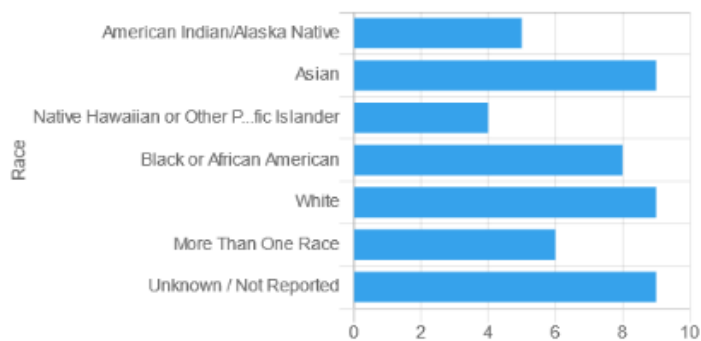
Add a third field for grouping (by color):



Add a third field for grouping (by color):



Display a **bar chart** with a single multiple choice field:



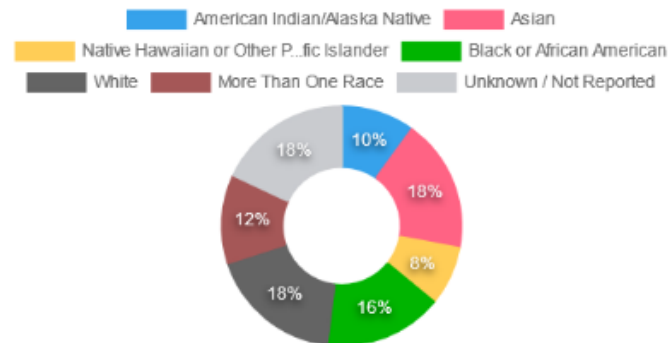
Display bar charts vertically, and add a second field for grouping:



Display a **pie chart**



Or a **donut chart**



Smart Variables

Name of Smart Variable	Description	Example input	Example output
	<i>(repeating instruments/events only) The last (highest</i>	<code>[glucose][last-instance]</code>	119
<p><b>Aggregate Functions, Charts, and Tables</b> (also known as Smart Functions, Smart Charts, and Smart Tables)</p>		<p><a href="#">VIDEO: How to use Smart Charts, Functions, and Tables (14 min)</a></p>	
<p><code>[aggregate-min:fields:parameters]</code></p>	<p>The minimum value of a field across all records in the project (including all events and/or repeating instances in all records). Multiple fields may be used and must be comma-separated.</p>	<p><code>[aggregate-min:age]</code></p>	13
<p><code>[aggregate-max:fields:parameters]</code></p>	<p>The maximum value of a field across all records in the project (including all events and/or repeating instances in all records). Multiple fields may be used and must be comma-separated.</p>	<p><code>[aggregate-min:age,participant_age,other_age]</code></p>	7
<p><code>[aggregate-mean:fields:parameters]</code></p>	<p>The mean/average value of a field across all records in the project (including all events and/or repeating instances in all records). Multiple fields may be used and must be comma-separated.</p>	<p><code>[aggregate-max:age]</code></p>	95
<p><code>[aggregate-mean:fields:parameters]</code></p>	<p>The mean/average value of a field across all records in the project (including all events and/or repeating instances in all records). Multiple fields may be used and must be comma-separated.</p>	<p><code>[aggregate-mean:age]</code></p>	100.1
<p><code>[aggregate-median:fields:parameters]</code></p>	<p>The median value of a field across all records in the project (including all events and/or repeating instances in all records). Multiple fields may be used and must be comma-separated.</p>	<p><code>[aggregate-median:age]</code></p>	57
<p><code>[aggregate-sum:fields:parameters]</code></p>	<p>The sum of all values for a field across all records in the project (including all events and/or repeating instances in all records). Multiple fields may be used and must be comma-separated.</p>	<p><code>[aggregate-sum:age]</code></p>	9451
<p><code>[aggregate-count:fields:parameters]</code></p>	<p>The count of all values for a field across all records in the project (including all events and/or repeating instances in all records). Multiple fields may be used and must be</p>	<p><code>[aggregate-count:age]</code></p>	68

Close

# Project Dashboard

QUESTIONS?

# Randomization

## Enable randomization:

Project Setup -> Enable optional modules and customizations -> Enable Randomization Module -> Randomization link becomes available on left menu

**Enable optional modules and customizations**

Modify	✔ Repeating instruments and events ?
Disable	✔ Auto-numbering for records ?
Enable	✖ Scheduling module (longitudinal only) ?
Disable	✔ Randomization module ?

**Applications**

- Project Dashboards
- Alerts & Notifications
- Multi-Language Management
- Calendar
- Data Exports, Reports, and
- Data Import Tool
- Data Comparison Tool
- Logging and Email Log
- Field Comment Log
- File Repository
- User Rights and DAGs
- Customize & Manage Locks
- Randomization



# Randomization

1. Randomization in REDCap works by allowing you to create your custom allocation list, which will serve as a lookup table for deciding how to randomize subjects.
2. Set up your randomization model and all its parameters in step 1
3. Download randomization template in step 2 and send to your statistician
4. Ask your statistician to create randomization table using the variable codes in the template
5. You will need one randomization table for testing and one for production. **The two tables cannot be the same.** Ask your statistician to create a randomization table with fewer number of allocations for testing.

# Randomization

## Step 1

### Step 1: Set up your randomization model and all its parameters

#### STEP 1: Define your randomization model

This step will allow you to define the randomization model you will be implementing and all its parameters, which includes defining strata (if applicable) and optionally randomizing subjects per group/site (if a multi-site study).

**NOTE:** This section is currently locked and uneditable because the randomization setup process has already been completed. If you wish to modify the randomization setup below, you will need to click the Erase Randomization Model button below.

#### A) Use stratified randomization?

It is often necessary to ensure equal treatment among a number of factors. Stratified randomization is the solution to achieve balance within one or more subgroups, such as gender, race, diabetics/non-diabetics, etc. By choosing strata (multiple choice criteria fields), you may then be able to ensure balance within those subgroups. [Tell me more](#)

**Choose strata** (criteria fields used for stratification; may specify up to 14 multiple choice fields)

scrn\_nitrate (1. Is the patient prescribed a long-acting nitrate?)  for

#### B) Randomize by group/site?

If this is a multi-center/multi-site project (or something similar), you may want to stratify the randomization by each group/site. You can select an existing multiple choice field that represents the groups/sites, OR you can use Data Access Groups to stratify by group/site.

- Use Data Access Groups to designate each group/site (5 groups currently exist)
- Use an existing field to designate each group/site

- select a field -  for

#### C) Choose your randomization field

This is the field where the allocated randomization (treatment) group will be saved and stored, and is where the Randomize button will appear on your data collection form.

random\_arm (Treatment Arm)  for

Save randomization model

Erase randomization model

# Randomization

## Step 2

### Step 2: Download the randomization template and send it to your statistician

#### STEP 2: Download template allocation tables (as Excel/CSV files)

Below are some example files that you may download to get a general idea for how you may structure your own randomization table. You do not have to use any of these. In fact, **we recommend that you NOT use these exact templates** but instead recommend that you merely use them as an example or baseline to start from in order to create your own custom allocation file. After uploading your allocation table in Step 3 below, it will then be used as a lookup table to perform assignments when subjects are being randomized. **NOTE:** Record names (e.g., study ID) should NOT be included as a column in your allocation table, but only the fields listed in the example files below. [More details](#)

Example #1 (basic)

Example #2 (all possible combos)

Example #3 (5x all possible combos)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	random_arm	scrn_nitrate	redcap_data_access_group														
2	1	1	449														
3	2	1	449			NOTES:											
4	1	0	449			- Do NOT modify the first row, although you may modify, add, or delete any other row in this file.											
5	2	0	449			- Remember that this file is ONLY a template and should NOT be used as-is as your allocation table.											
6	1	1	450			- You do not have to delete this 'notes' column when uploading your allocation table (it will be ignored).											
7	2	1	450			- Below is a list of all raw coded values and their corresponding option labels for each strata field and/or Data Access Groups.											
8	1	0	450														
9	2	0	450			Values/labels for "random_arm" (Treatment Arm):											
10	1	1	451			1 Beta Blocker											
11	2	1	451			2 Calcium Channel Blocker											
12	1	0	451														
13	2	0	451			Values/labels for "scrn_nitrate" (<div class="rich-text-field...</p></div>):											
14	1	1	452			1 Yes											
15	2	1	452			0 No											
16	1	0	452														
17	2	0	452			Values/labels for "redcap_data_access_group" (Data Access Group):											
18	1	1	448			449 BWH											
19	2	1	448			450 Duke											
20	1	0	448			451 Inova											
21	2	0	448			452 Kaiser											
22						448 Yale											
23																	

# Randomization

## Step 3

### STEP 3: Upload your allocation table (CSV file)

Once you have created your custom allocation table as a CSV file and made sure that you kept the format prescribed in the template files from Step 2 above, you may now upload the file below. It will be checked for any possible errors first before it is accepted and stored in REDCap. Please note that you will need to create two different allocation tables: one to be used for testing while your project is in development status and the other for use when in production status. Below are some important reminders before you begin uploading your allocation tables.

#### Reminders:

- Once your project is in production status, the allocation tables will become locked and unmodifiable.
- Be sure to include more assignments in your allocation table than you think you will need (to accommodate possible drop-out and drop-in of subjects).
- Record names (e.g., study ID) should NOT be included as a column in your allocation table, but only the fields listed in the example files from Step 2 above.



Not  
uploaded  
yet

#### Upload allocation table (CSV file) for use in DEVELOPMENT status

No file selected.



Not  
uploaded  
yet

#### Upload allocation table (CSV file) for use in PRODUCTION status

No file selected.

You will need two randomization tables – one for development and one for production.  
The two tables must NOT be the same.  
Ask your statistician to generate a smaller randomization table for testing (development).

# Randomization Testing

- Upload the development randomization table for testing
- Test randomization in the development project
- **Very Important** – Do NOT randomize any real participants in your development project

Upload allocation table (CSV file) for use in DEVELOPMENT status  
[Delete allocation table?](#)

Already uploaded

---

Upload allocation table (CSV file) for use in PRODUCTION status  
 No file selected.

Not uploaded yet

- After the randomization table is uploaded, the randomization button will become available in the randomization field. Click the button to randomize test records.

Treatment Arm  
\* must provide value

# Randomization

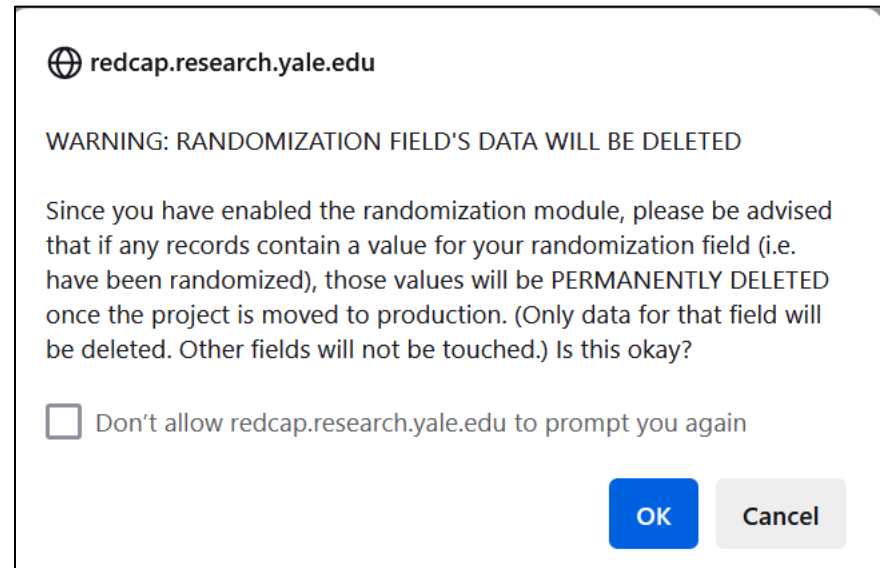
## Move to Production

- After testing completed and no more changes are needed, upload production randomization table



- Move project to production. All test randomization data will be **deleted**.

This is the reason why you should never randomize any real participants in development project.



# Randomization Reminders

## Reminders:

- Once your project is in production status, the allocation tables will become locked and unmodifiable.
- Be sure to include more assignments in your allocation table than you think you will need (to accommodate possible drop-out and drop-in of subjects).
- Record names (e.g., study ID) should NOT be included as a column in your allocation table, but only the fields listed in the example files from Step 2.
- If you need to upload more allocations to the randomization table, contact us at [redcap@yale.edu](mailto:redcap@yale.edu). We can help you append more rows to the randomization table.

# Randomization Dashboard

To monitor randomization, go to the dashboard tab of the randomization module.

Setup Dashboard

The table below displays the allocation dashboard for use in DEVELOPMENT status. All assignments are grouped to show in aggregate the count of records that have been randomized for each row (i.e. combinations). Assignments that have been used will get counted in the 'Used' column while those that are still unallocated will get counted in the 'Not Used' column. Once all assignments have been used for a given row/combination, it will display a checkmark icon in its row. The headers in the table may be clicked to sort the table by that column either in ascending or descending order.

	Used	Not Used	Allocated records	Data Access Group	Treatment Arm (random_arm)	1. Is the patient prescribed a long-ac... (scrn_nitrate)
<input type="radio"/>	0	49		Yale (379)	Beta Blocker (1)	No (0)
<input type="radio"/>	7	43	<a href="#">YAL-034, 8, 18, 379-2, 21, 30, 7</a>	Yale (379)	Beta Blocker (1)	Yes (1)
<input type="radio"/>	2	49	<a href="#">9, 28</a>	Yale (379)	Calcium Channel Blocker (2)	No (0)
<input type="radio"/>	6	44	<a href="#">6, 17, 19, 20, 379-5, 379-6</a>	Yale (379)	Calcium Channel Blocker (2)	Yes (1)
<input type="radio"/>	2	48	<a href="#">363-018, BWH-018</a>	BWH (380)	Beta Blocker (1)	No (0)
<input type="radio"/>	3	47	<a href="#">15, 380-3, 363-034</a>	BWH (380)	Beta Blocker (1)	Yes (1)
<input type="radio"/>	0	50		BWH (380)	Calcium Channel Blocker (2)	No (0)
<input type="radio"/>	3	47	<a href="#">380-4, 380-5, 380-6</a>	BWH (380)	Calcium Channel Blocker (2)	Yes (1)
<input type="radio"/>	0	51		Duke (381)	Beta Blocker (1)	No (0)

Note: If your study has blinded study staff, make sure that you do not give them access to the randomization module or the form containing the randomization variable.



# *Randomization*

QUESTIONS?

# Wrap Up

- *Form Display Logic*
- *Special Functions*
  - *Calculations*
  - *Branching Logic*
- *Action Tags*
- *Smart Variables*
- *Project Dashboard*
- *Randomization*

Visit:



Further Questions:  
[REDCap@yale.edu](mailto:REDCap@yale.edu)

Thank You!