Reproducible Data Wrangling for Excel Users

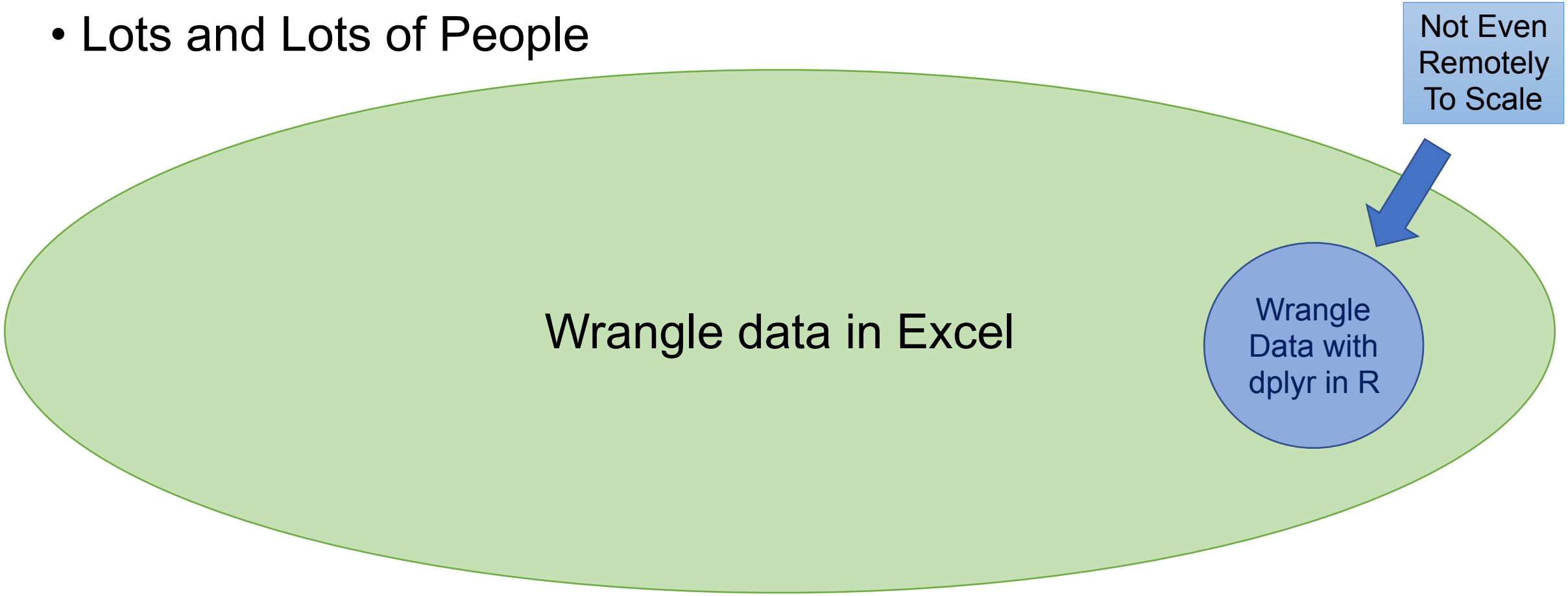
Peter Higgins

Who Am I?

- Enthusiastic R amateur.
- Day jobs:
  - Lab scientist at University of Michigan
  - Clinician (GI, Inflammatory Bowel Disease)
  - Clinical research
  - Clinical trials
  - Drug development
  - Epidemiology
  - Predictive modeling for drug efficacy in individuals
  - Director, Clinical Trials Support Unit

Who Wrangles Data?

- Lots and Lots of People



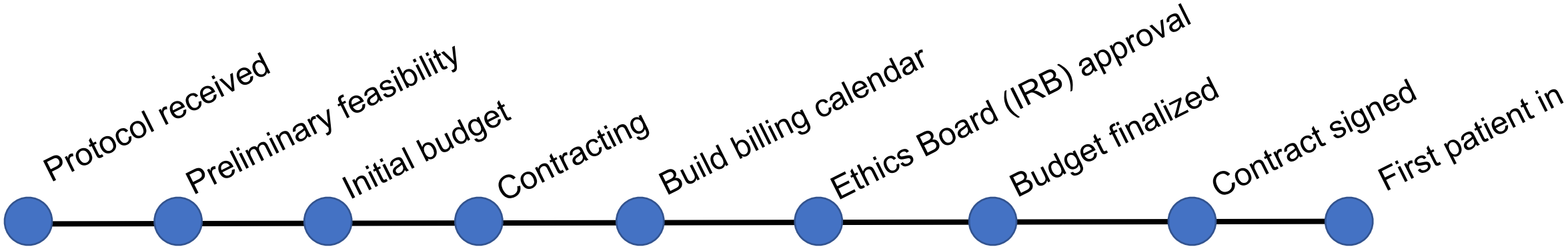
Why Excel?

- It is available
- Lots of people nearby who sort of know how to use Excel
- You can see the data while wrangling: breeds (over)confidence

- Why not Excel?
  - Point and click is ***not reproducible***
  - Easy (simple) data wrangling leads to overuse, ***fails to scale***
  - Potential for long, low-reproducibility data pipelines repeated by hand
  - Lots of scribbled data wrangling ’recipes’, errors, lots of re-doing

Typical Example

- Clinical Trial Support Units have lots of data in Forte Oncore
  - A proprietary and expensive “Enterprise Research System”
  - Lots of canned reports, exported in Excel *.xls format
  - Not easily customized
- Would like to track events in time-to-startup for clinical trials



Problem

- 332 protocols – in one report
- > 5400 tasks with completion dates – in a different report
- Each week: download the 2 reports, do 30 data-wrangling steps
  - Filter
  - Sort
  - Pivot tables (gather/spread)
  - Merges with VLOOKUP tables (joins)
  - All with point and click in 5 different tabs
  - To produce a final report
  - Costs about 2 person-hours each week, often with do-overs

Actual 30 Data-Wrangling Steps

- Run the OnCore Task Report: Reports>Task Management>Tasks
- Copy the results and past them on a blank tab and rename the tab ‘Tasks’
- Remove the first 3 rows and any blank columns.
- Select the first row and add a filter, sort to show only the tasks in which NA (column H) is marked ‘Y’
- For these studies, enter NA as the completed date (column G). (This was the only way we could indicate a blank task was N/A versus blank because it’s still pending)
- Once this is done, go back to the filter on the NA column and show all.
- Then click in the Insert Menu click on ‘Pivot Table’, select the columns containing data and click ok. Do not select any blank columns, you will get an error message.
- The pivot table will open in a new sheet. Name this tab ‘Pivot Table’.
- Use the Pivot table fields and filter by task list, task name, protocol no, and completed date:

Actual 30 Data-Wrangling Steps

- Click on the filter in cell 1B and select the task list you want to filter by in the search box. This will select all the tasks in that task list.
- Copy the entire page and paste the values in a new worksheet. Select all the columns and make sure you format them as a date, column A should be formatted as General.
- Find and Replace all of the 1/0/1900 dates as NA
- Delete the first 3 rows
- Run a protocol search and select any information you wish to show it in the results (i.e. sponsor).
- View your results in Excel and copy them into a tab and name it ‘Protocol Search’.
- Delete the first 2 rows
- Delete the Protocol Title Column and Department Column
- Go back to your results tab, the one with the values of the pivot table copied to it.
- Insert 6 rows between Row Labels and Intake Form Completed. Name them as follows:

Actual 30 Data-Wrangling Steps

- In cell B2 enter: =VLOOKUP($A2, 'Protocol Search'!$A:$C, 2, FALSE)
  - Use the plus sign to copy this formula to all the cells in this column. If you receive a #N/A for some cells but not for others, it's because that protocol no. isn't listed on your Protocol Search tab. It's likely because it's not one of the studies in your CTSU. You can delete these studies from your worksheet.
- In cell C2 enter: =VLOOKUP($A2, 'Protocol Search'!$A:$F, 6, FALSE)
  - Drag this formula to copy to all the cells in this column
- In cell D2 enter: =VLOOKUP($A2, 'Protocol Search'!$A:$F 3, FALSE)
  - Drag this formula to copy to all the cells in this column
- IN cell E2 enter: =VLOOKUP($A2, 'Protocol Search'!$A:$F, 4, FALSE)
  - Drag this formula to copy to all the cells in this column
- In cell F2 enter: =VLOOKUP($A2, 'Protocol Search'!$A:$F, 5, FALSE)
  - Drag this formula to copy to all the cells in this column
  - Make sure this column is formatted as a date
- If you want to know the CTSU finance person assigned to the study follow the instructions below. If not (i.e. you only have 1 person doing all the pre-award), delete Column G (Owner).
  - Go to the Tasks column and filter column B (Task Name) to show only the tasks of Created CTRF & Notify ORSP.
- Copy these results onto a new tab called 'Pre-Award Tasks'
- On the Pre-Award Task sheet, insert a new column before column A
- Copy Column J (Protocol No) and paste it into Column A
- In Cell A2 enter: =VLOOKUP($A2, 'Pre-Award Tasks'!$A:$G, 5, FALSE)
- Drag this formula to copy to all the cells in this column
  - This formula is pulling the owner listed for the task of Route CTRF on the pre-award task list for that study.

First solution

- Translate each of 30 data wrangling steps into R
- dplyr, tidyr, joins
- Use writexl package to write final file
- Save as script
- Run each week

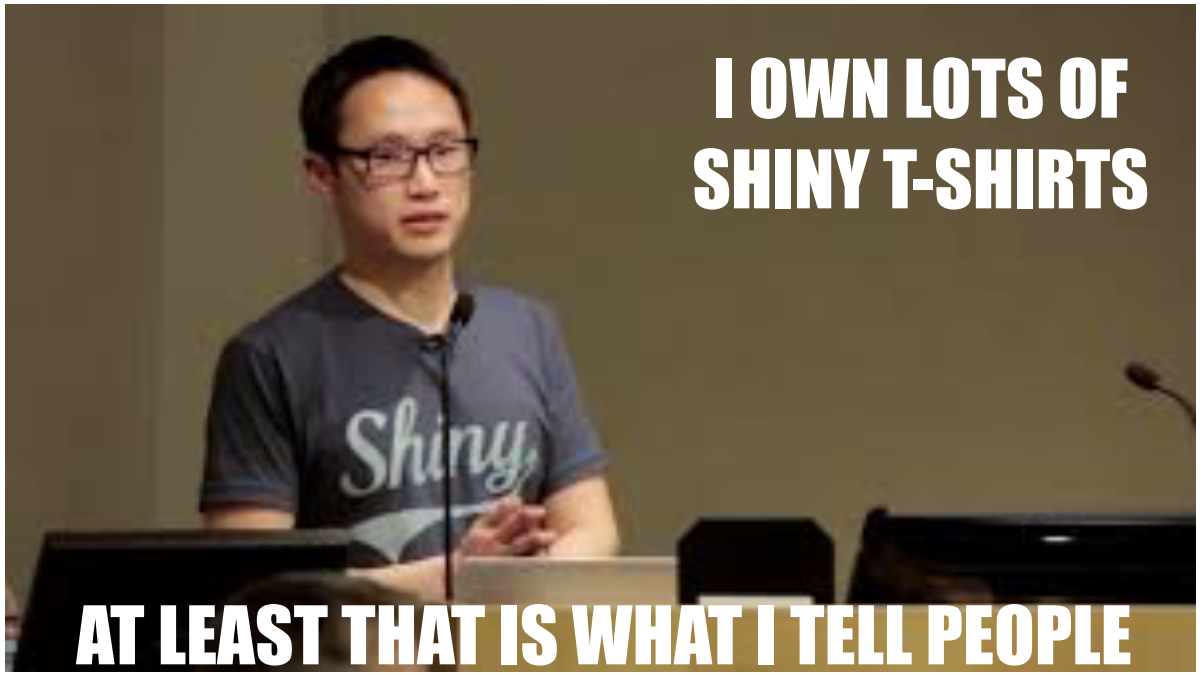
Problems Arise

- Must run queries each Wed AM
- Email 2 *.xls files to me
- Ideally I run script, email final *.xls report back by end of day

- But, some Wednesdays are endoscopy days
- Other Wednesdays for travel to meetings/presentations
- Does not always work out (I am usually the failure point)
- Finance/Admin teams will never use install R, much less code.

Second Solution

- Wrap this code in a Shiny App!
  - Upload file 1
  - Upload file 2
  - Wrangle data
  - Download final report



First attempt

- Can upload files !
- Can display data from both file 1 and file 2 !!
- But data wrangling code not running…
- Problem – need ***reactivity***
  - ***Reactive endpoints*** execute when upstream inputs change
- Thanks to Ellis – identified the issue and the fix.
  - **Code/Demo**
  - https://pdrhiggins.shinyapps.io/ctsu/

Outcomes

- Excel users can stay in their comfort zone
- Can avoid major data-wrangling in Excel
- Data wrangling needs are reproducibly met with minimal fuss
- Users do not need to install R, learn to code
- One small part of this big green ellipse
  is now doing ***reproducible*** data wrangling with Excel

Future

- There are probably a lot of not-so-reproducible data wrangling pipelines in Excel out there.
- Some of your best friends probably use Excel.
- Shiny apps could help improve both reproducibility and efficiency
- And you can seem magical to your co-workers.
- https://github.com/higgi13425/ctsu

Questions?

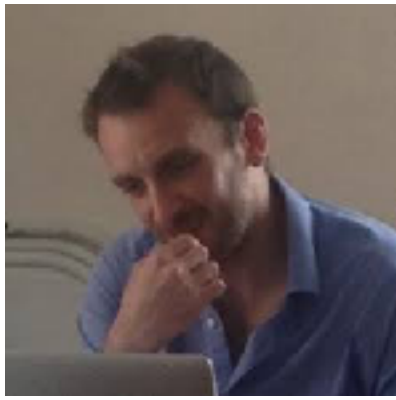
- And on to part 2!

From a REDCap Clinical Trial Database to an NIH Enrollment Report

Wrangling (mostly) tidy data from a database into a standard (very untidy) government-mandated reporting table

Useful R stuff you could learn

- Use of ***keyring*** package to securely use passwords
- Use of ***RedCapR*** package to extract data via API
- Use of dplyr::mutate(***case_when***) for wrangling
- Use of ***janitor::tabyl*** for creating a 3D counts table
- Use of ***tidyr::complete*** to fill in all combinations of factors (even ones that did not occur in the data)
- Lots of ***tidyr*** (unite, separate, and spread)
- Make the table pretty with ***flextable***
- Save directly to Word and Powerpoint with ***officer***



David Gohel

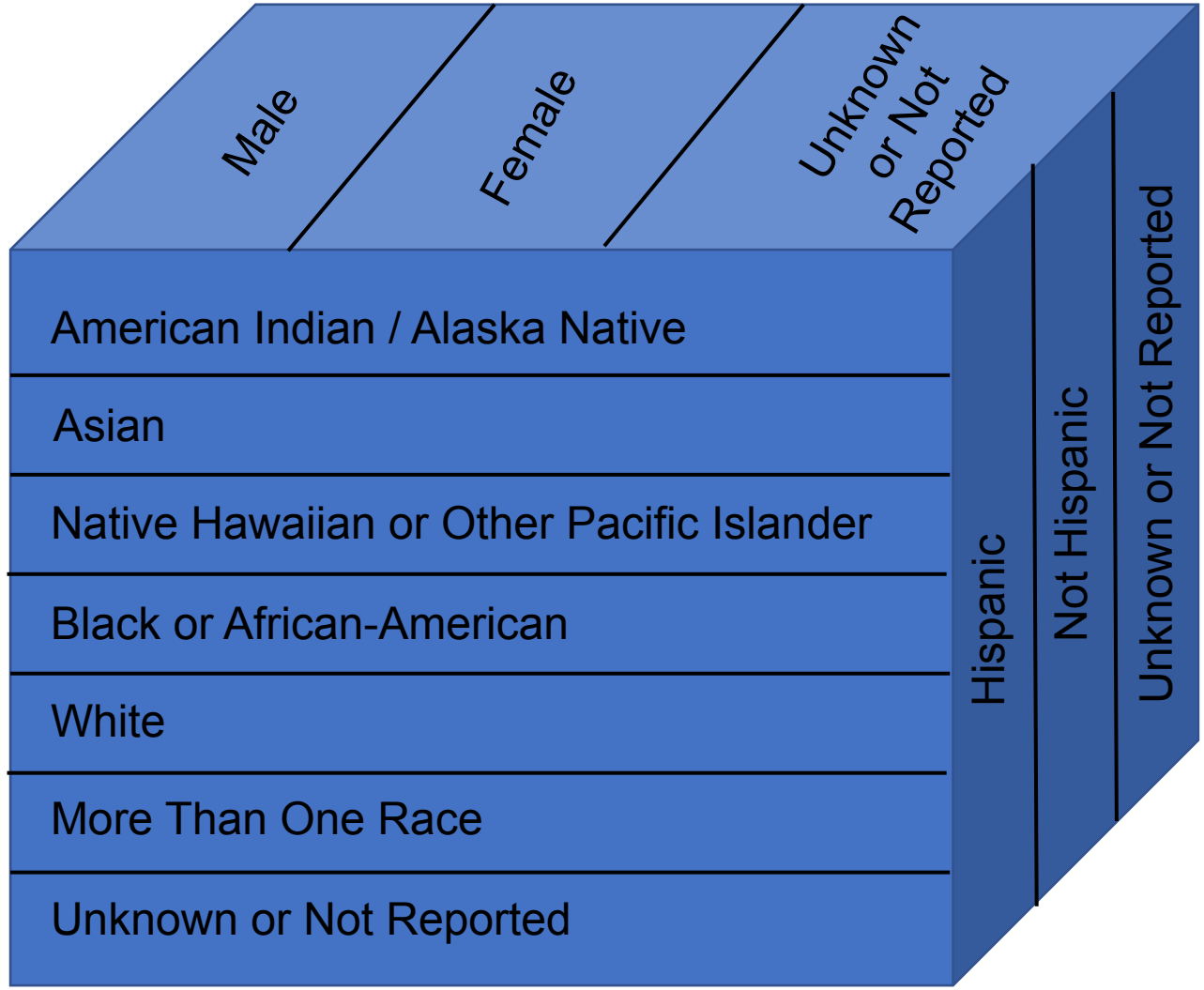
What is REDCap?

- Research Electronic Data CAPture
- HIPAA compliant web database
  - Health Insurance Portability and Accountability Act
  - PHI (Personal Health Information) is protected
- Enables secure data entry from multiple sites
- Allows real-time tracking of enrollment
- Allows real-time tracking of data quality

Used in 2,992 institutions in 192 countries worldwide
Export data to R, SAS, SPSS

What is an NIH Enrollment Report?

- When you have an NIH grant that includes enrolling patients
- Every year you have to provide a standard enrollment report in your annual progress report
- This requires you to count enrolled subjects and divides them on three dimensions
  - Race: 7 categories
  - Ethnicity: 3 categories
  - Sex: 3 categories
- To produce a 3 dimensional matrix of 63 cells of counts
- Then submit this as a 2 dimensional (very untidy) table
- With totals for both rows and columns



63 cell table

	Ethnic Categories									
	Not Hispanic or Latino			Hispanic or Latino			Unknown/Not Reported Ethnicity			Total
Racial Categories	Female	Male	Unknown/ Not Reported	Female	Male	Unknown/ Not Reported	Female	Male	Unknown/ Not Reported	
American Indian/Alaska Native	1	0	1	0	0	0	0	0	0	2
Asian	1	0	0	0	0	0	0	0	0	1
Native Hawaiian or Other Pacific Islander	0	0	0	0	0	0	0	0	0	0
Black or African American	1	1	0	0	1	0	0	0	0	2
White	10	11	0	2	1	0	1	2	0	27
More than One Race	2	3	0	1	0	0	1	1	0	8
Unknown or Not Reported	2	1	0	0	0	0	0	0	0	3
Total	17	16	1	3	2	0	1	3	0	0

Format of an NIH Enrollment Report

	Ethnic Categories									
	Not Hispanic or Latino			Hispanic or Latino			Unknown/Not Reported Ethnicity			Total
Racial Categories	Female	Male	Unknown/ Not Reported	Female	Male	Unknown/ Not Reported	Female	Male	Unknown/ Not Reported	
American Indian/ Alaska Native	1	0	1	0	0	0	0	0	0	2
Asian	1	0	0	0	0	0	0	0	0	1
Native Hawaiian or Other Pacific Islander	0	0	0	0	0	0	0	0	0	0
Black or African American	1	1	0	0	1	0	0	0	0	2
White	10	11	0	2	1	0	1	2	0	27
More than One Race	2	3	0	1	0	0	1	1	0	8
Unknown or Not Reported	2	1	0	0	0	0	0	0	0	3
Total	17	16	1	3	2	0	1	3	0	43

Actual Scenario

- Hi Peter.
- Do you remember when you got some funds from the Peptide Center grant for the IBD databank?
- The annual progress report is due tomorrow, and it turns out that we need an NIH Enrollment Report for this. Could you get this filled out by 5 PM today?

REDCap Data Access with keyring

- First you need to get access
- Request a read-only API key
  - Email your local REDCap administrator to request
  - Looks something like "9A8126EA46645C4E5F03728B8AC3AA7B”
  - You don’t want this in your code.

```
install_packages(‘keyring’)
library(keyring)
```

Can then try:

key_list() OR
glimpse(key_list())

Mine has 195 entries,
mostly old WiFi connections
from hotel stays

Secure Access to REDCap with keyring

```
url <- 'https://redcap-p-a.umms.med.umich.edu/api/’
keyring::key_set(service = url
        username = "your_username")
```

- When prompted with a dialog box
- Enter your API token with no quotes around it
- This is now stored in your keyring(Mac) or wincred (Windows) or backend_secret_service (Linux)

Secure Access to REDCap data

- Now to securely access your data in code without revealing your API token

```
url <- 'https://redcap-p-a.umms.med.umich.edu/api/’
token <- keyring::key_get(
        service = url,
        username = ‘your_username’)
library(REDCapR)
df <- redcap_read(redcap_uri = url, token = token)$data
```

What Does REDCap Data look like?

ibd_table2 | nih_table.R | ibd | Untitled1* | biobank_data_import.R

Filter

	subject_id	redcap_event_name	sex	marital_status	race___0	race___1	race__
1	1052	baseline_visit_arm_1	0	2	1	0	
2	IBD1001	baseline_visit_arm_1	0	2	1	0	
3	IBD1001	year_1_visit_arm_1	NA	NA	NA	NA	
4	IBD1002	baseline_visit_arm_1	1	2	1	0	
5	IBD1003	baseline_visit_arm_1	0	2	1	0	
6	IBD1003	year_1_visit_arm_1	NA	NA	NA	NA	

Problems :
- More than one visit per subject
- Need to decode sex
- Need to decode ethnicity
- Race has 7 columns to coalesce into 1
- Unnecessary columns to filter out (subject_id, marital status)

Wrangling with dplyr & magrittr

```
ibd %>%
  filter(redcap_event_name == "baseline_visit_arm_1") %>%
  select(sex, race___0:ethnicity) %>%
  mutate(race  =  case_when(
            .$race___0 == 1 ~ "White",
            .$race___1 == 1 ~ "Black or African-American",
            .$race___2 == 1 ~ "Asian",
            .$race___3 == 1 ~ "Native Hawaiian or Other Pacific Islander",
            .$race___4 == 1 ~ "American Indian or Alaska Native",
            .$race___5 == 1 ~ "More Than One Race",
            .$race___999 == 1 ~ "Unknown or Not Reported",
            TRUE ~ "Unknown or Not Reported")) %>%
```

Wrangling with dplyr & magrittr

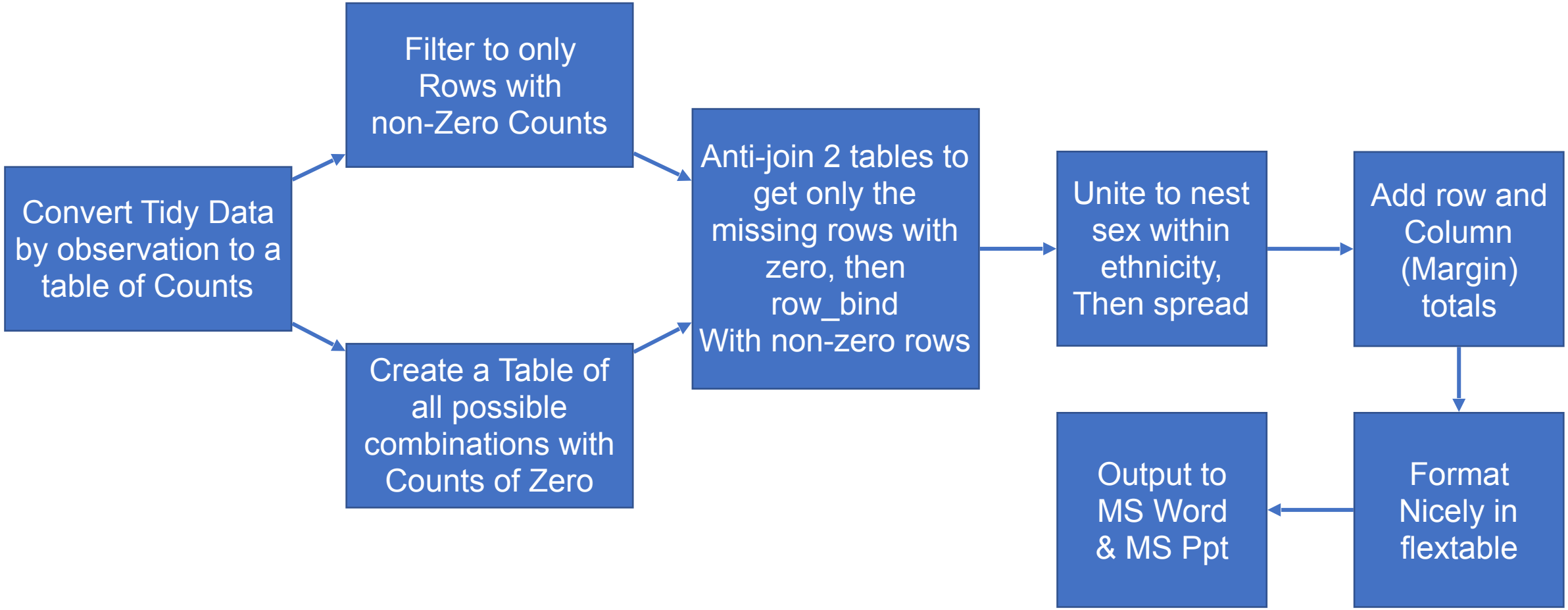
```
mutate(ethnic_cat = case_when(
    .$ethnicity == 1 ~ "Hispanic or Latino",
    .$ethnicity == 0 ~ "Not Hispanic or Latino",
    TRUE ~ "Unknown or Not Reported Ethnicity")) %>%
  mutate(sex2 = case_when(
    .$sex == 1 ~ "Male",
    .$sex == 0 ~ "Female",
    TRUE ~ "Female")) %>%
  select(sex2, race, ethnic_cat) ->
ibd
```

Now you have tidy data

	sex2	race	ethnic_cat
1	Female	White	Hispanic or Latino
2	Female	White	Not Hispanic or Latino
3	Male	White	Not Hispanic or Latino
4	Female	White	Not Hispanic or Latino
5	Female	White	Not Hispanic or Latino
6	Male	More Than One Race	Not Hispanic or Latino
7	Male	White	Not Hispanic or Latino
8	Female	White	Not Hispanic or Latino
9	Male	White	Not Hispanic or Latino
10	Female	White	Not Hispanic or Latino
11	Male	White	Unknown or Not Reported Ethnicity

Problems : Need to convert from single observation per row to counts of all combinations.
Lots of missing combinations, i.e. Asian/Hispanic/Female – need 63 cells.
These need to be present, and need to be assigned a count of zero.

Data Wrangling Plan



Convert to a table of counts

```
ibd_table <- ibd %>%
        tabyl(race, sex2, ethnic_cat) %>%  # creates list of 3 tables
        reduce(left_join, by = "race")  # purrr reduces to one table
        # 17 rows have non-zero counts
```

- **DEMO – create table of counts**
  tabyl x 2, tabyl x 3, tabyl x 3 with reduce

	race	Female.x	Male.x	Female.y	Male.y	Female	Male
1	Asian	0	1	3	5	0	0
2	Black or African-American	0	0	1	1	0	0
3	More Than One Race	1	0	1	1	0	0
4	Unknown or Not Reported	0	0	0	1	2	1
5	White	5	1	43	46	1	3

Make a Table of all possible combinations of categories, with all counts = 0

```
# make a list of four vectors of length 7
l <- list(race =  c("White",   "Black or African-American", "Asian",
          "Native Hawaiian or Other Pacific Islander",  "American Indian or Alaska Native", "More Than One Race", "Unknown or Not Reported"),
      sex = c("male", "female", "Unknown or Not Reported Sex", "male", "female", "male", "female"),
     ethnicity = c("Hispanic", "Not", "Unknown", "Hispanic", "Not", "Hispanic", "Not"),
       count = rep(0,7))
```

```
> as_tibble(l)
# A tibble: 7 x 4
  race                          sex                ethnicity count
  <chr>                         <chr>              <chr>     <dbl>
1 White                         male               Hispanic      0
2 Black or African-American     female             Not           0
3 Asian                         Unknown or Not Re… Unknown       0
4 Native Hawaiian or Other P…   male               Hispanic      0
5 American Indian or Alaska     female             Not           0
6 More Than One Race            male               Hispanic      0
7 Unknown or Not Reported       female             Not           0
```

rownames(x, do.NULL = TRUE, prefix = "row")

Make a Table of all possible combinations of categories, with all counts = 0

```
empty_table <- as_tibble(l) %>%
    tidyr::complete(race,
      nesting(sex), nesting(ethnicity),fill=list(count = 0))
```

```
# A tibble: 63 x 4
   race                                sex          ethnicity count
   <chr>                               <chr>        <chr>     <dbl>
 1 American Indian or Alaska Native    female       Hispanic      0
 2 American Indian or Alaska Native    female       Not           0
 3 American Indian or Alaska Native    female       Unknown       0
 4 American Indian or Alaska Native    male         Hispanic      0
 5 American Indian or Alaska Native    male         Not           0
 6 American Indian or Alaska Native    male         Unknown       0
 7 American Indian or Alaska Native    Unknown or … Hispanic      0
 8 American Indian or Alaska Native    Unknown or … Not           0
 9 American Indian or Alaska Native    Unknown or … Unknown       0
```

Now filter 2 tables

- Filter actual counts to only have non-zero rows

```
ibd_table2 <- gather(ibd_table, key= sex.eth, value = count, -race) %>%
  separate(sex.eth, into = c('sex', 'ethnicity')) %>%
  filter(count != 0)  # 17 non-zero rows
```

- Anti-join with empty table to get only the needed zero rows

```
complement <- anti_join(empty_table, ibd_table2,
                        by = c('race', 'sex', ‘ethnicity’))
          # complement is 46 rows with zeros
```

Now make full 63 row table

full_table <- bind_rows(ibd_table2, complement)

***DEMO* – create table of counts**

tabyl x 2, tabyl x 3, tabyl x 3 with reduce

- Now it is tidy and complete.
- Now to make it un-tidy for standard formatting
  - Sex is nested within ethnicity in the NIH table
  - Still need to add row totals and column totals

Wrangling nested sex, ethnicity

- Unite to nest sex within ethnicity as eth.sex
- Then spread

```
ibd_table <- full_table %>%
  unite(col = "eth.sex", c('ethnicity', "sex"), sep=".") %>%
# three cols - race, eth.sex, count
  spread(key = eth.sex, value = count)
# now spread to 10 cols
```

DEMO wrangle nested sex within ethnicity

Adding Margin totals

```
# convert race col to rownames to make numbers into a matrix
m <- as.matrix(ibd_table[ ,-1]) # removes col1 (race) in m
rownames(m) <- ibd_table$race   # saves race in rownames

ibd_table2 <- addmargins(m, FUN=c(Total=sum), quiet = T)

ibd_table <- rownames_to_column(as.data.frame(ibd_table2), "Racial Categories")  #
puts race back into dataframe from rownames
```

DEMO add margin totals

Now to make nice tables

- ***Flextable*** is a package by David Gohel (officer, reporters)
- To make nicely formatted tables for Word and Powerpoint
- Very well documented, multiple vignettes
- Functionality similar to the kableExtra package
- Works well with Rmd (HTML), .docx, .pptx
  - Not so much with LaTex, PDF if that is your thing
  - ***DEMO make flextable***
    show myft after add each header, before mergeh, mergev, final
  - https://github.com/higgi13425/nih_enrollment_table

Output to Word, Powerpoint

- ***officer*** package

```
doc <- read_docx()
doc <- body_add_flextable(doc, value = myft)
print(doc, target = "/path/file.docx")

ppt <- read_pptx()
ppt <- add_slide(ppt, layout = "Title and Content",
                 master = "Office Theme")
ppt <- ph_with_flextable(ppt, value = myft, type= "body")
print(ppt, target = "/path/file.docx")
```

Questions?

Thanks for your interest!

Problems to fix

- Mention Cmd-Shift-R in Rstudio for Section Names
- Update Github
- Fix names of ibd_tables – make more descriptive than 2, 3
- Standardize capitalization of sex, ethnicity
- Make all unknown = unknown (not longer)
- Keep race names short until end