The R EdSurvey Package

Using R and Generative AI for NAEP Data Analysis: Part One

Presenters: Sinan Yavuz, Ting Zhang, Paul Bailey, Blue Webb and Emmanuel Sikali

April 2024

NAEP
NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS

Goal

Provide participants with an overview of the methods used to analyze large-scale assessment data using the R package `EdSurvey` and generative AI chatbot-EdSurveyGPT

Outline of EdSurvey Session: Part One

- EdSurveyGPT
- Why R and Why EdSurvey
- Get to Know the R Environment
- Data Processing
- Meet Your Data (`searchSDF`, `levelSDF`, `showCodebook`)
- Data Manipulation

EdSurveyGPT

NAEP
NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS

Why R and Why EdSurvey

NAEP

NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS

Why R?

1. **Free:** Users can legally use and edit R package code
2. **Extensible:** Large variety of contributed packages that expand its functionality
3. **Transparent:** Codes are open, so it provides full transparency in its mechanism and the formulas behind each functionality
4. **Reproducible:** Automated data analysis
5. **Designed by and for researchers:** Robust ecosystem to translate data into analyses, visualizations, and summary reports with one software

Why EdSurvey?

1. **One-stop shop**: For data downloading, processing, manipulation, and analysis of survey data.
2. **Automated**: Weights and complex sampling design calculations are automated following standard NCES methodology.
3. **Simple**: For example a regression with 62 replicate weights and 20 plausible values requires only a few lines of code.
4. **Flexible**: You can use functions that rely on `EdSurvey` methods or get the data and use traditional `R`.
5. **Expansible**: Has the capacity to expand functions and data supports to meet the needs of analysts of various levels of expertise
6. **Minimizes memory footprint**: by only reading in required data.

Basic R Infrastructure

CRAN stores packages

Basic R Infrastructure

CRAN stores packages

Accessed via

```
install.packages("ggplot2")
```

and loaded into R on your machine

```
library("ggplot2")
```

Basic R Infrastructure

CRAN stores packages

Accessed via

```
install.packages("ggplot2")
```

and loaded into R on your machine

```
library("ggplot2")
```

These package libraries consist of functions

```
ggplot()
geom_point()
...
```

Basic R Infrastructure

CRAN stores packages

Accessed via

```
install.packages("ggplot2")
```

and loaded into R on your machine

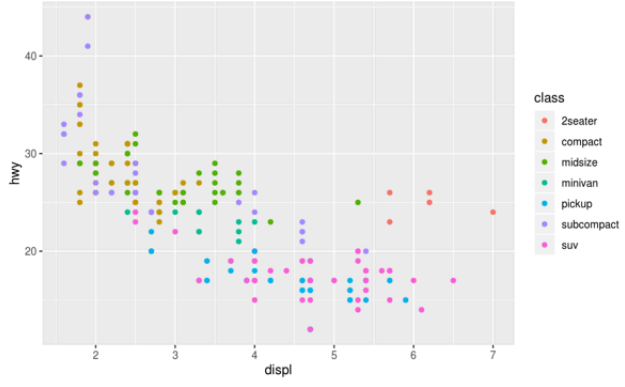
```
library("ggplot2")
```

These package libraries consist of functions

```
ggplot()
geom_point()
...
```

That can be used to analyze data

```
ggplot(mpg, aes(displ, hwy, colour = class)) +
  geom_point()
```

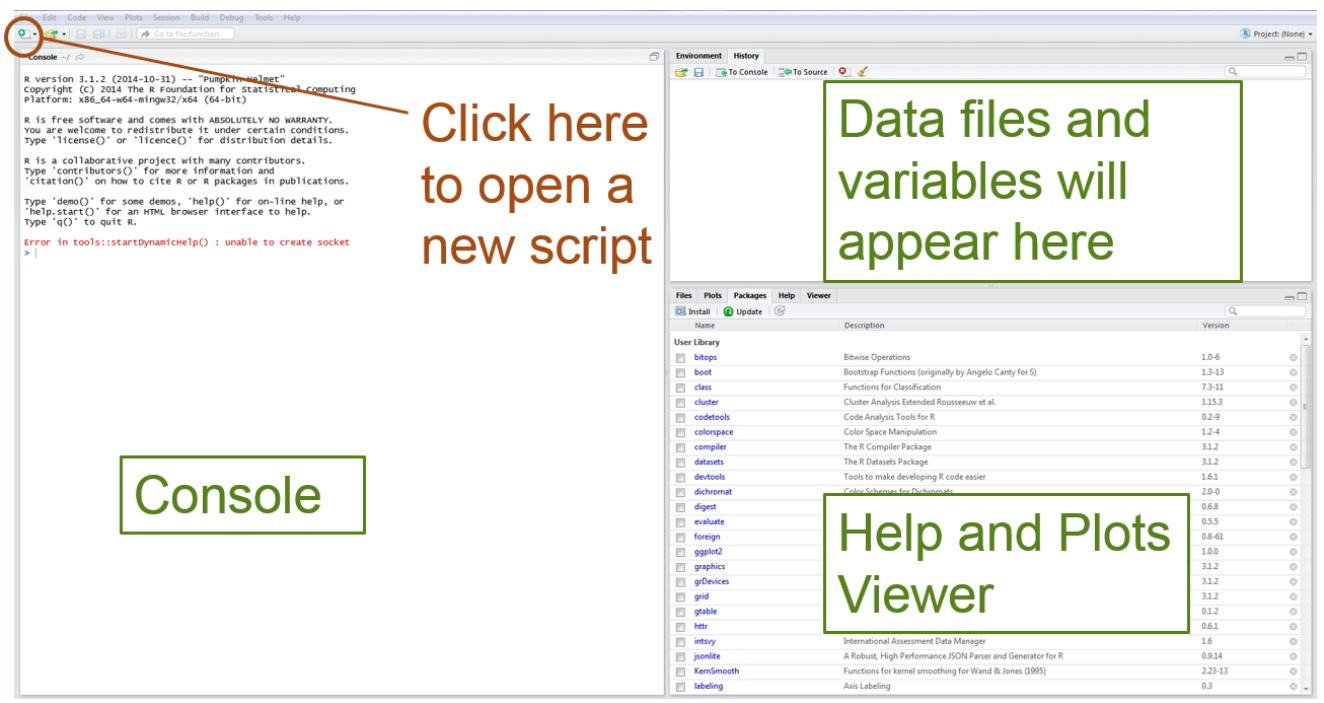


Get to Know the R Environment

NAEP
NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS

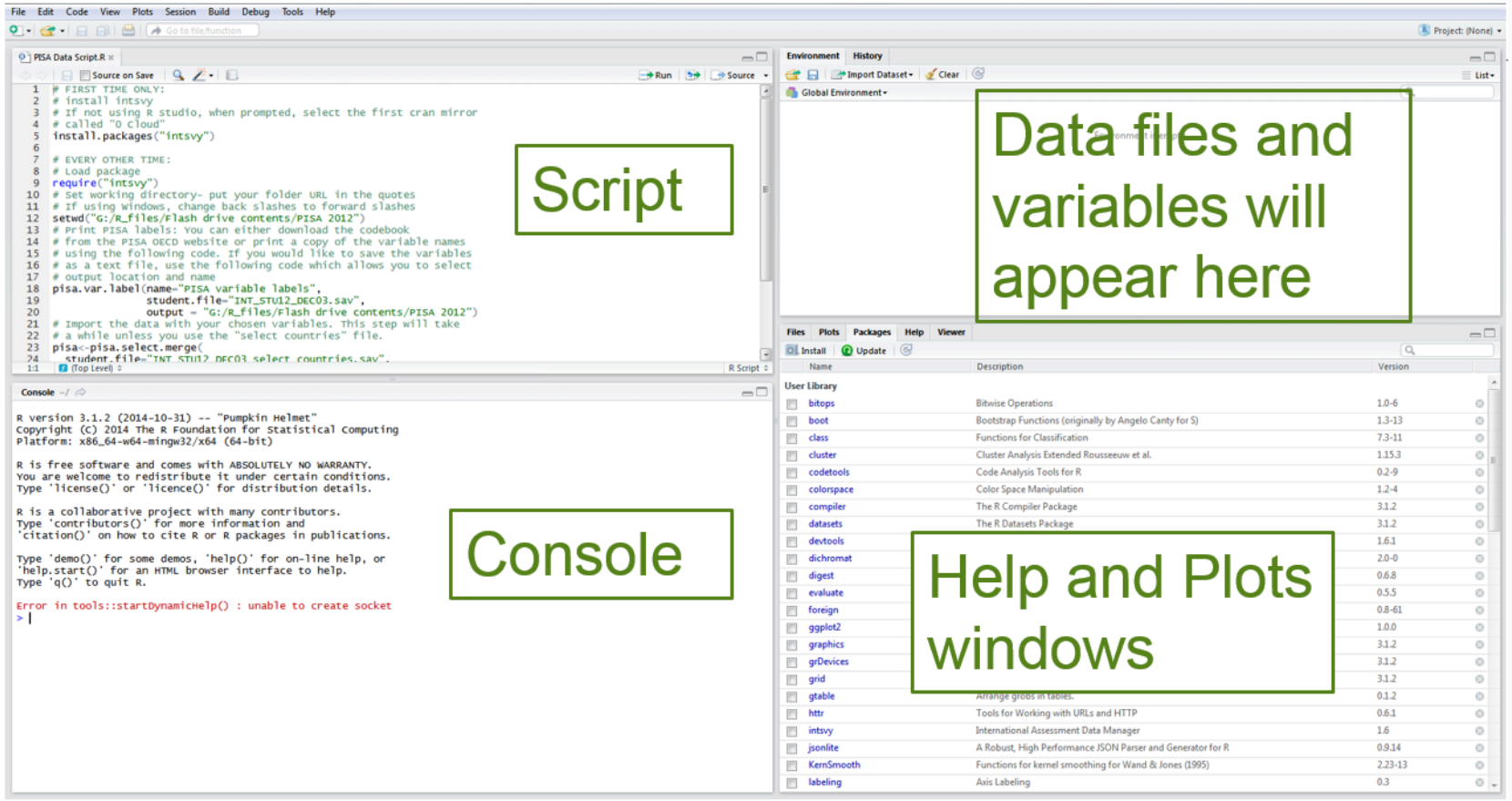
RStudio

- If it's your first time using R, note that it will ask you to select a mirror
  - Try the first one on the list: "0-cloud"



Using R Scripts

Open script



Notes About Using R

- Comment character is a hash

```
# this line is not executed
```

- Variables are assigned with an equals or `<-`

```
x <- 12
x
```

```
## [1] 12
```

- In file names on Windows use a forward slash
  - `C:/`
- R is case sensitive!

```
j <- 12
J
```

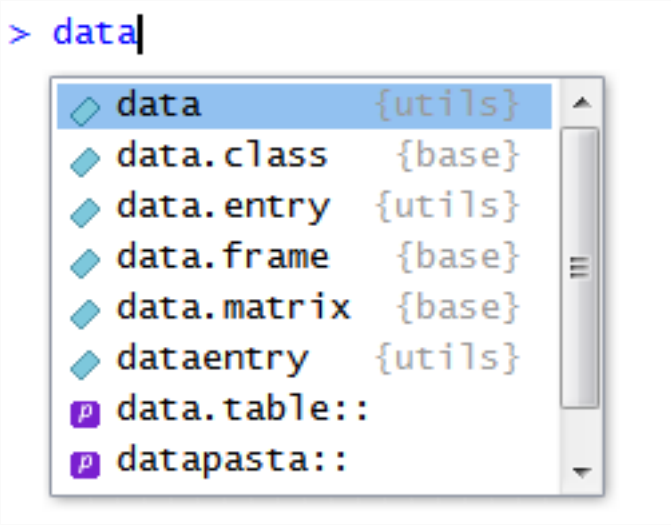
```
## Error in eval(expr, envir, enclos): object 'J' not found
```

Notes About Using R

- Any command can be input with a question mark preceding it to open the help guide

```
?mean
```

- Use the up arrow on your keyboard to copy your previous lines of code
- Try tab completion: type "data" then hit the tab key



Using R Functions

- `c()` function combines values separated by a comma into a vector

```
colors <- c("red", "green", "blue")
colors
```

```
## [1] "red"   "green" "blue"
```

```
numbers <- c(1, 2, 3)
numbers
```

```
## [1] 1 2 3
```

- In the `EdSurvey` package we'll use vectors to combine the names of variables in our analyses

Using R Functions

- Arguments can be explicitly or implicitly named

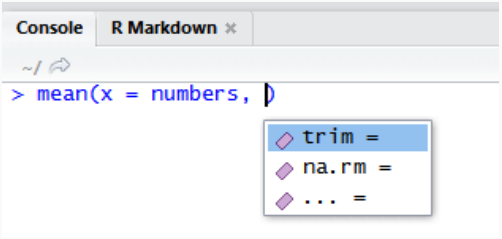
```
mean(x = numbers)
```

```
## [1] 2
```

```
mean(numbers)
```

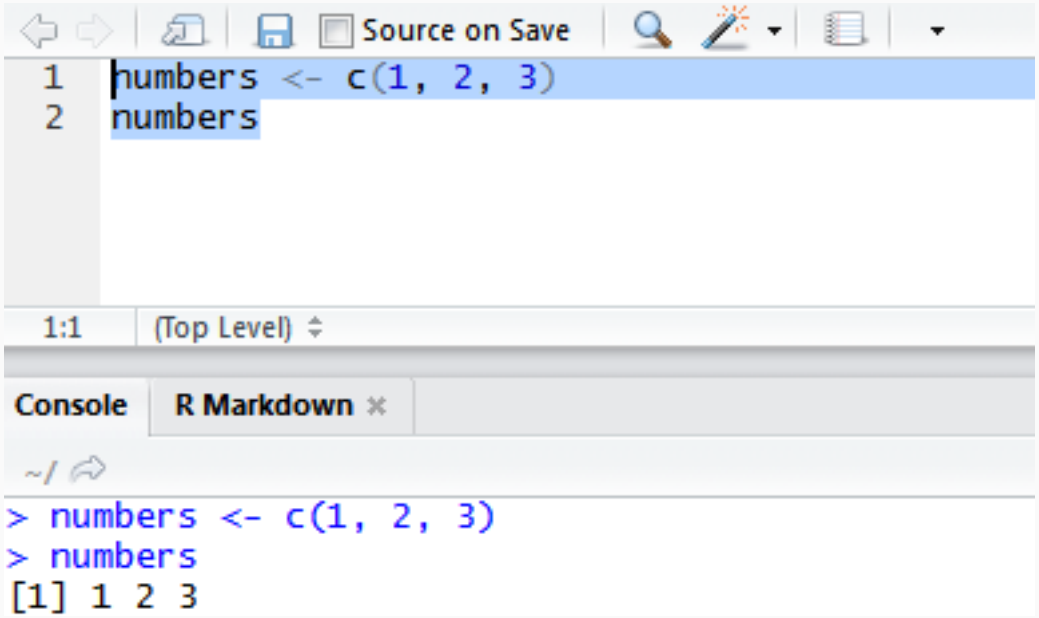
```
## [1] 2
```

- Arguments are separated by commas



Follow Along: R Scripts

- Highlight and press `ctrl` + `enter` executes code to console



Follow along in edsurvey_training_part1.R

Installing the EdSurvey Package

- After opening up RStudio, run the following scripts in the console to download and initialize the `EdSurvey` package:

```
# to install the package
install.packages("EdSurvey")

# to load the package
library(EdSurvey)
```

Learning EdSurvey

- Reading vignettes provided in training materials

```
vignette("introduction", package="EdSurvey")
```

- R help

```
help(package = "EdSurvey")
```

- EdSurvey User Guide
- EdSurvey Website
- EdSurvey Github
- EdSurvey Data Training Workshops

Think, Pair, Share: R Functions

What are the arguments of the function `readNAEP()`? What are some examples of acceptable values for each argument?

Think, Pair, Share: R Functions

`readNAEP()` arguments from R documentation (*type* `?readNAEP` *in the console*)

Usage

```
readNAEP(
  path,
  defaultWeight = "origwt",
  defaultPvs = "composite",
  omittedLevels = c("Multiple", NA, "Omitted"),
  frPath = NULL,
  xmlPath = NULL
)
```

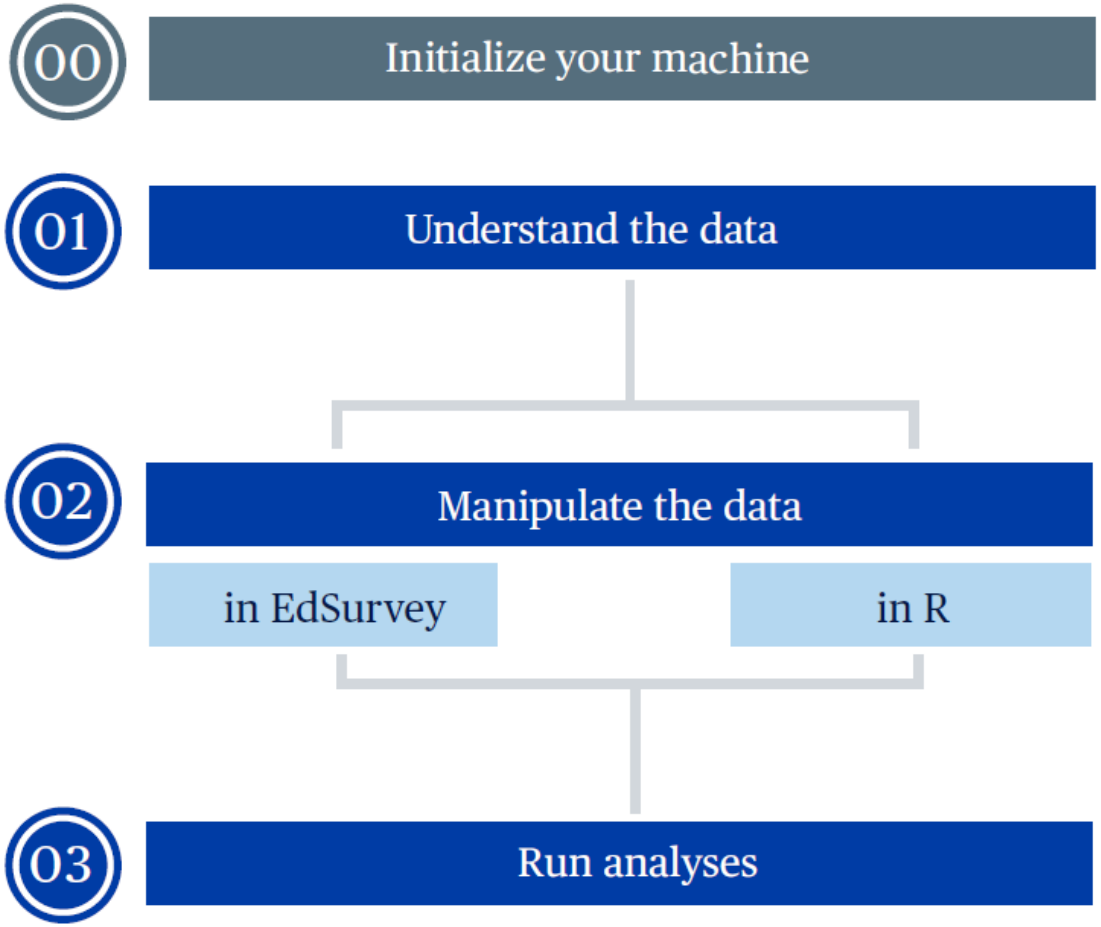
Arguments

`path`	a character value indicating the full filepath location and name of the (.dat) data file
`defaultWeight`	a character value that indicates the default weight specified in the resulting `edsurvey.data.frame`. Default value is `origwt` if not specified.
`defaultPvs`	a character value that indicates the default plausible value specified in the resulting `edsurvey.data.frame`. Default value is `composite` if not specified.
`omittedLevels`	a character vector indicating which factor levels/labels should be excluded. When set to the default value of `c('Multiple',NA,'Omitted')`, adds the vector to the `edsurvey.data.frame`.
`frPath`	a character value indicating the file location of the `.fr2` parameter layout file included with the data companion to parse the specified `path` data file. The default value of `NULL` will attempt to search the parent directory for the corresponding `.fr2` file for the specified `path` data file.
`xmlPath`	a character value indicating the file path of the `.xml` parameter layout file included as part of the NAEPEX companion. This file provides necessary information required to read and parse the (.dat) data file. The default value of `NULL` will attempt to search the parent directory for the corresponding `.xml` file for the specified `path` data file.

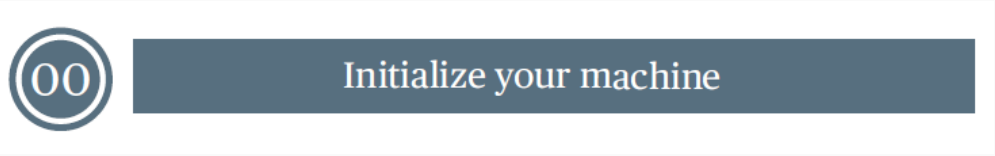
Data Processing

NAEP
NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS

Data Processing: EdSurvey Workflow



Data Processing: EdSurvey Workflow



- Install R and EdSurvey
- Download and read in data

Example functions:

- `readTIMSS`, `readPIRLS`, `readPISA`, `readTALIS`
- `downloadTIMSS`, `downloadPIRLS`, `downloadPISA`, `downloadTALIS`

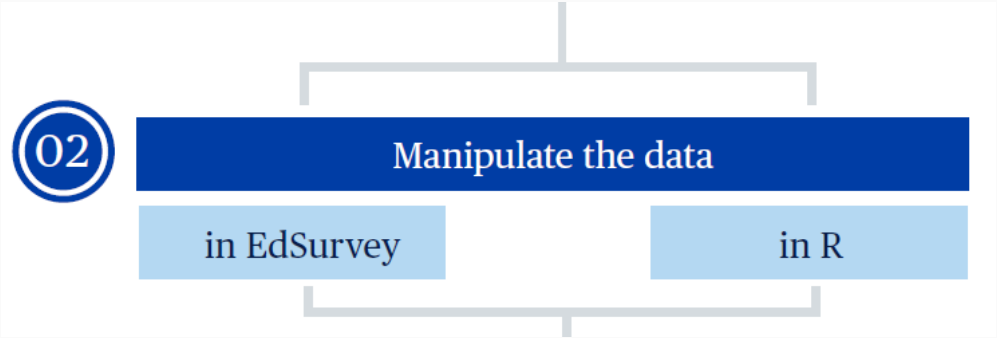
Data Processing: EdSurvey Workflow

- **Explore**: Explore the codebook, see the variables with plausible values, see weights
- **Search**: Search variables
- **Expand**: See variable levels, tabulate response percentages, see assessment scores by response category, summarize continuous variables

Example functions:

- `showCodebook`, `showPlausibleValues`, `showWeights`
- `searchSDF`, `levelsSDF`
- `summary2`, `edsurveyTable`

Data Processing: EdSurvey Workflow



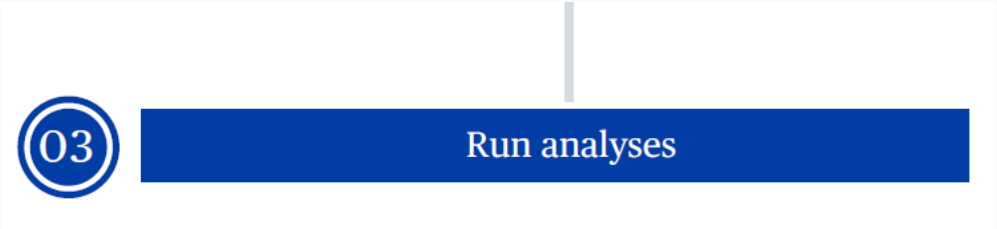
In **EdSurvey**: Clean and manipulate data with built-in subset and recode features

In **R**: Extract and manipulate data as a data frame (for experienced users)

Example functions:

- `subset`
- recode with `ifelse`
- `getData`, `rebindAttributes`

Data Processing: EdSurvey Workflow

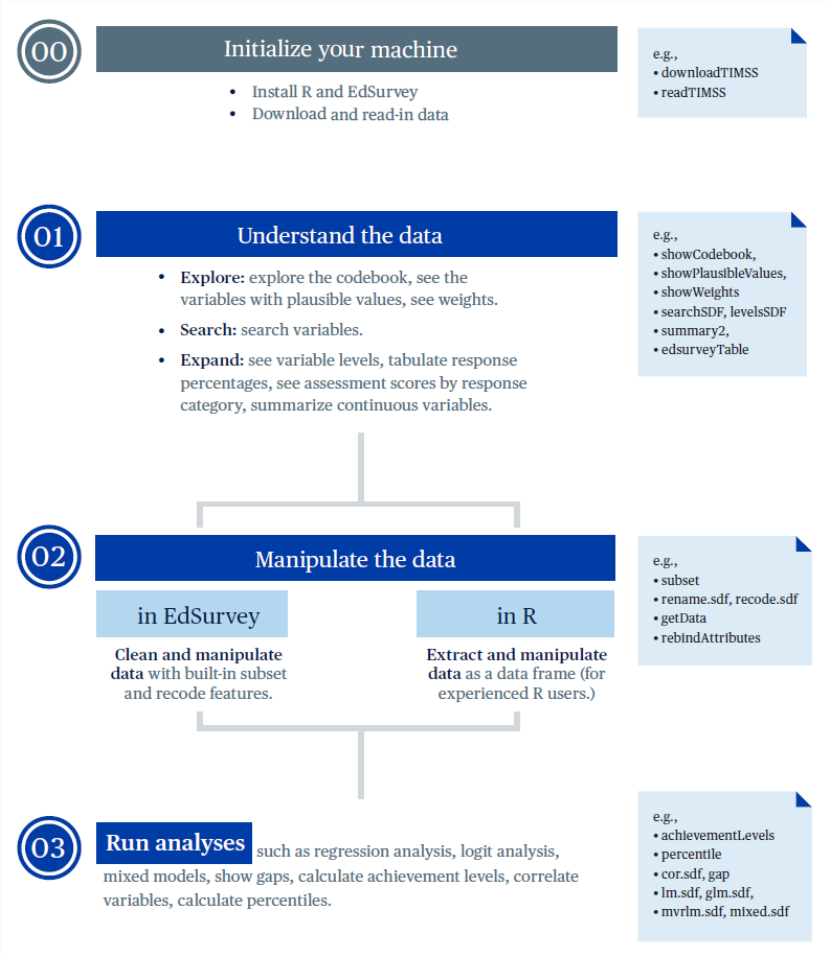


- **Run analyses** such as regression analysis, logit analysis, mixed models, show gaps, calculate achievement levels, correlate variables, calculate percentiles.

Example functions:

- `achievementLevels`, `percentile`
- `cor.sdf`
- `gap`
- `lm.sdf`, `glm.sdf`
- `mvrlm.sdf`, `mixed.sdf`

Data Processing: EdSurvey Workflow

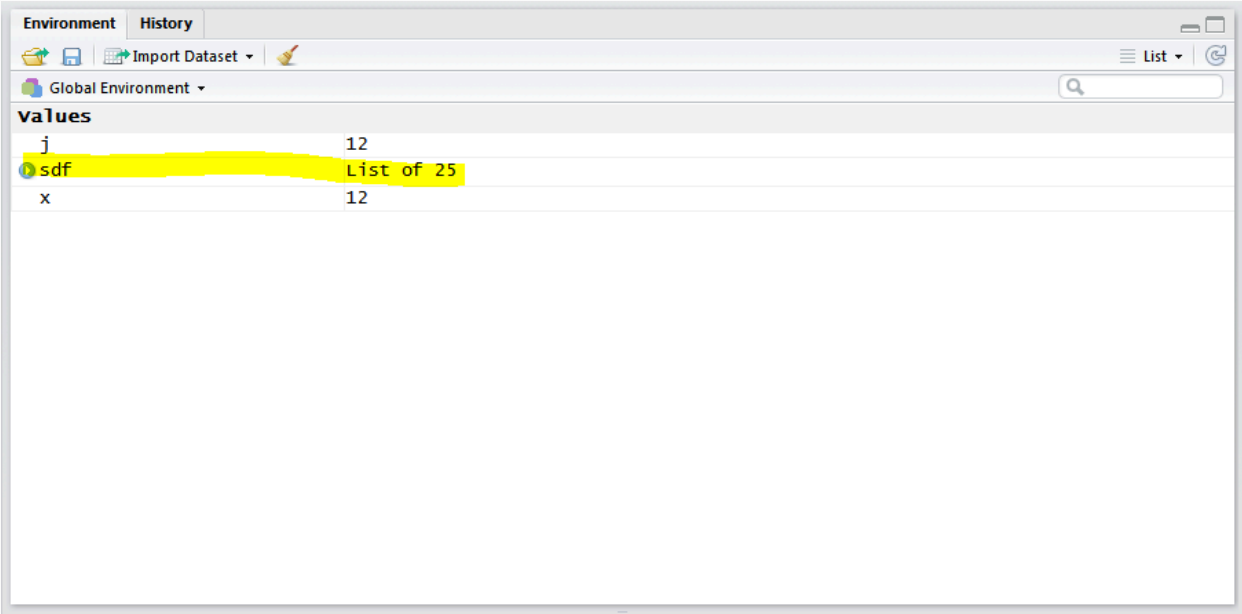


- Related documentation: EdSurvey User Guide

Data Processing: Reading NAEP Data

- First, read in the publicly available NAEP data from the `NAEPprimer` package. `NAEPprimer` is a mini sample of the 2005 NAEP mathematics Grade 8 assessment for public schools.

```
sdf <- readNAEP(system.file("extdata/data", "M36NT2PM.dat", package
```



Data Processing: Reading NAEP Data

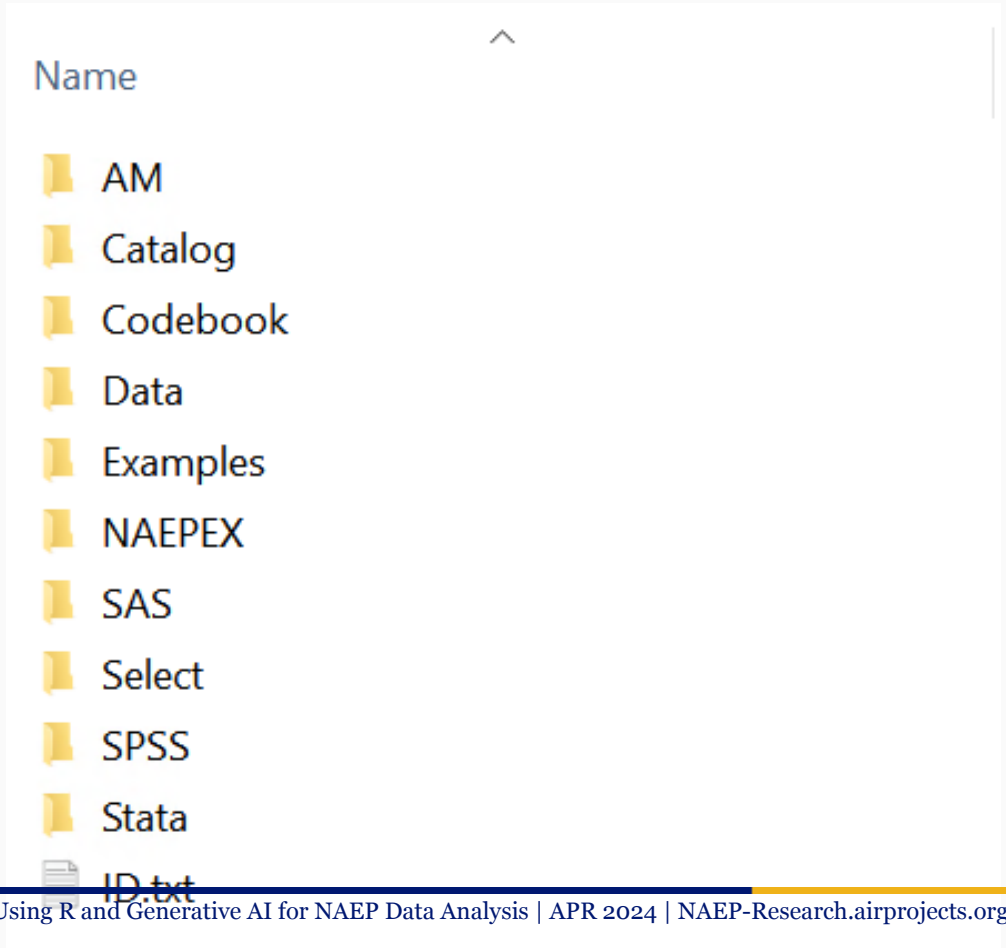
- Can also read in other restricted-use NAEP data by naming the file location

```
math19 <- readNAEP("//path_to_directory/Data/M50NT2AT.dat")
```

- The first character indicates the subject: `M` (Math)
- The second and third characters indicate the NAEP year: `50` (2019 – 1969 = 50)
- The fourth character indicates the component: `N` (National)
- The fifth character indicates the type of data: `T` (Student)
- The sixth character indicates the grade cohort: `2` (8th)
- The seventh and eighth characters indicate the sample: `AT` (Main NAEP)

Data Processing: Reading NAEP Data

NOTE: The data file requires its intact data folder directory in order to be read in correctly. It must contain both the student- and school-level files to merge data.



Meet Your Data

NAEP
NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS

Quick Terminology Notes

The `edsurvey.data.frame` class stores information about survey data via a data connection, which allows for:

- correct calculation of relevant statistics.
- limited working memory usage.

`_____SDF`

The EdSurvey package uses the acronym `SDF` in the names of several functions to signify their relationship to **S**urvey **D**ata **F**rames.

Meet Your Data: print

`print()`

- The `print` function returns detailed data file information:

```
print(sdf)
```

```
## edsurvey.data.frame for 2005 NAEP National - Primer (Mathematics; Grade 8) in USA
## Dimensions: 17606 rows and 303 columns.
##
## There is 1 full sample weight in this edsurvey.data.frame:
##   'origwt' with 62 JK replicate weights (the default).
##
##
## There are 6 subject scale(s) or subscale(s) in this edsurvey.data.frame:
## 'num_oper' subject scale or subscale with 5 plausible values.
##
## 'measurement' subject scale or subscale with 5 plausible values.
##
## 'geometry' subject scale or subscale with 5 plausible values.
##
## 'data_anal_prob' subject scale or subscale with 5 plausible values.
##
```

Meet Your Data: colnames

`colnames()`

- Prints the names of all variables in the data:

```
colnames(sdf)
```

```
##   [1] "ROWID"   "year"    "cohort"  "scrpsu"  "dsex"    "iep"     "lep"     "ell3"    "sdracem" "pared"   "b003501" "b0
##  [13] "b013801" "b017001" "b017101" "b018101" "b018201" "b017451" "m815401" "m815501" "m815601" "m815801" "m815701" "rp
##  [25] "repgrp1" "repgrp2" "jkunit"  "origwt"  "srwt01"  "srwt02"  "srwt03"  "srwt04"  "srwt05"  "srwt06"  "srwt07"  "sr
##  [37] "srwt09"  "srwt10"  "srwt11"  "srwt12"  "srwt13"  "srwt14"  "srwt15"  "srwt16"  "srwt17"  "srwt18"  "srwt19"  "sr
##  [49] "srwt21"  "srwt22"  "srwt23"  "srwt24"  "srwt25"  "srwt26"  "srwt27"  "srwt28"  "srwt29"  "srwt30"  "srwt31"  "sr
##  [61] "srwt33"  "srwt34"  "srwt35"  "srwt36"  "srwt37"  "srwt38"  "srwt39"  "srwt40"  "srwt41"  "srwt42"  "srwt43"  "sr
##  [73] "srwt45"  "srwt46"  "srwt47"  "srwt48"  "srwt49"  "srwt50"  "srwt51"  "srwt52"  "srwt53"  "srwt54"  "srwt55"  "sr
##  [85] "srwt57"  "srwt58"  "srwt59"  "srwt60"  "srwt61"  "srwt62"  "smsrswt" "mrps11"  "mrps12"  "mrps13"  "mrps14"  "mr
##  [97] "mrps21"  "mrps22"  "mrps23"  "mrps24"  "mrps25"  "mrps31"  "mrps32"  "mrps33"  "mrps34"  "mrps35"  "mrps41"  "mr
## [109] "mrps43"  "mrps44"  "mrps45"  "mrps51"  "mrps52"  "mrps53"  "mrps54"  "mrps55"  "mrpcm1"  "mrpcm2"  "mrpcm3"  "mr
## [121] "mrpcm5"  "m075201" "m075401" "m075601" "m019901" "m066201" "m047301" "m046201" "m066401" "m020101" "m067401" "m0
## [133] "m047701" "m067301" "m048001" "m093701" "m086001" "m051901" "m076001" "m046001" "m046101" "m067701" "m046701" "m0
## [145] "m047201" "m046601" "m046801" "m067801" "m066601" "m067201" "m068003" "m068005" "m068008" "m068007" "m068006" "m0
## [157] "m053001" "m047801" "m086301" "m085701" "m085901" "m085601" "m085501" "m085801" "m019701" "m020001" "m046301" "m0
## [169] "m046501" "m066501" "m047101" "m066301" "m067901" "m019601" "m051501" "m047901" "m053101" "m143601" "m143701" "m1
## [181] "m143901" "m144001" "m144101" "m144201" "m144301" "m144401" "m144501" "m144601" "m144701" "m144801" "m144901" "m1
```

Meet Your Data: searchSDF

`searchSDF()` searches the survey data frame by character strings:

```
searchSDF("education", data = sdf)
```

```
##   variableName                                  Labels fileFormat
## 1        pared Parental education level (from 2 questions)    Student
## 2     b003501                   Mother's education level    Student
## 3     b003601                   Father's education level    Student
## 4     c044007               Percent in special education    Student
```

- Add argument `levels = TRUE` to return variable levels:

```
searchSDF("b003501", data = sdf, levels = TRUE)
```

```
## Variable: b003501
## Label: Mother's education level
## Levels (Lowest level first):
##      1. Did not finish H.S.
##      2. Graduated H.S.
##      3. Some ed after H.S.
##      5. I don't know
```

- What occurs with an empty string?

```
searchSDF("", data = sdf)
```

Meet Your Data: levelsSDF

`levelsSDF()`

- Shows the levels of a variable:

```
levelsSDF(varnames = "b018201", data = sdf)
```

```
## Levels for Variable 'b018201' (Lowest level first):
##     1. Never (n = 9524)
##     2. Once in a while (n = 3328)
##     3. Half the time (n = 1178)
##     4. All or most of time (n = 2133)
##     8. Omitted* (n = 741)
##     0. Multiple* (n = 11)
##     NOTE: * indicates an omitted level.
```

Meet Your Data: showCodebook

`showCodebook()`

- Shows the levels of all variables printed to the console. If there are lots of variables, this can be hard to read.

```
showCodebook(data = sdf) # or showCodebook(sdf)
```

- The output can be assigned to a variable and is returned as a `data.frame`.

```
book <- showCodebook(sdf)
```

- `View()` shows a preview of a selected data set in a new 'Excel'-like tab in RStudio (*RStudio specific).

```
View(showCodebook(sdf))
```

Meet Your Data: showPlausibleValues

`showPlausibleValues()` prints all plausible values:

```
showPlausibleValues(sdf)
```

```
## There are 6 subject scale(s) or subscale(s) in this edsurvey.data.frame:
## 'num_oper' subject scale or subscale with 5 plausible values.
##
## 'measurement' subject scale or subscale with 5 plausible values.
##
## 'geometry' subject scale or subscale with 5 plausible values.
##
## 'data_anal_prob' subject scale or subscale with 5 plausible values.
##
## 'algebra' subject scale or subscale with 5 plausible values.
##
```

- add `verbose = TRUE` for all detail:

```
showPlausibleValues(sdf, verbose = TRUE)
```

```
## There are 6 subject scale(s) or subscale(s) in this edsurvey.data.frame:
## 'num_oper' subject scale or subscale with 5 plausible values.
##   The plausible value variables are: 'mrps11', 'mrps12', 'mrps13', 'mrps14', and 'mrps15'
##
## 'measurement' subject scale or subscale with 5 plausible values.
##   The plausible value variables are: 'mrps21', 'mrps22', 'mrps23', 'mrps24', and 'mrps25'
##
```

Meet Your Data: showWeights

`showWeights()` prints all weights:

```
showWeights(sdf)
```

```
## There is 1 full sample weight in this edsurvey.data.frame:
##   'origwt' with 62 JK replicate weights (the default).
```

- add `verbose = TRUE` to print the complete list of jackknife replicate weights associated with each full sample weight:

```
showWeights(sdf, verbose = TRUE)
```

```
## There is 1 full sample weight in this edsurvey.data.frame:
##   'origwt' with 62 JK replicate weights (the default).
##     Jackknife replicate weight variables associated with the full sample weight 'origwt':
##     'srwt01', 'srwt02', 'srwt03', 'srwt04', 'srwt05', 'srwt06', 'srwt07', 'srwt08', 'srwt09', 'srwt10', 'srwt11',
##     'srwt12', 'srwt13', 'srwt14', 'srwt15', 'srwt16', 'srwt17', 'srwt18', 'srwt19', 'srwt20', 'srwt21', 'srwt22',
##     'srwt23', 'srwt24', 'srwt25', 'srwt26', 'srwt27', 'srwt28', 'srwt29', 'srwt30', 'srwt31', 'srwt32', 'srwt33',
##     'srwt34', 'srwt35', 'srwt36', 'srwt37', 'srwt38', 'srwt39', 'srwt40', 'srwt41', 'srwt42', 'srwt43', 'srwt44',
##     'srwt45', 'srwt46', 'srwt47', 'srwt48', 'srwt49', 'srwt50', 'srwt51', 'srwt52', 'srwt53', 'srwt54', 'srwt55',
##     'srwt56', 'srwt57', 'srwt58', 'srwt59', 'srwt60', 'srwt61', and 'srwt62'
```

Meet Your Data: Omitted Levels

- Levels of the variables that will be omitted (excluded) by default from `edsurvey.data.frame` results
- By default, all analytical functions apply list-wise deletions for omitted levels (the default settings can be changed)

```
> sdf
edsurvey.data.frame with 17606 rows and 302 columns.

There are 1 full sample weight(s) in this edsurvey.data.frame
  'origwt' with 62 JK replicate weights (the default).

There are 6 subject scale(s) or subscale(s) in this edsurvey.data.frame
  'num_oper' subject scale or subscale with 5 plausible values.
  'measurement' subject scale or subscale with 5 plausible values.
  'geometry' subject scale or subscale with 5 plausible values.
  'data_anal_prob' subject scale or subscale with 5 plausible values.
  'algebra' subject scale or subscale with 5 plausible values.
  'composite' subject scale or subscale with 5 plausible values (the default).

Omitted Levels: 'Multiple', 'NA', 'Omitted'

Default Conditions:
  tolower(rptsamp) == "reporting sample"

Achievement Levels:
  Basic:      262
  Proficient: 299
  Advanced:   333

Survey: NAEP
```

Meet Your Data: Default Conditions

- Special considerations for a particular `edsurvey.data.frame`
- For NAEP, this defaults the 'reporting sample,' which excludes selected students who did not respond

```
> sdf
edsurvey.data.frame with 17606 rows and 302 columns.

There are 1 full sample weight(s) in this edsurvey.data.frame
  'origwt' with 62 JK replicate weights (the default).

There are 6 subject scale(s) or subscale(s) in this edsurvey.data.frame
  'num_oper' subject scale or subscale with 5 plausible values.
  'measurement' subject scale or subscale with 5 plausible values.
  'geometry' subject scale or subscale with 5 plausible values.
  'data_anal_prob' subject scale or subscale with 5 plausible values.
  'algebra' subject scale or subscale with 5 plausible values.
  'composite' subject scale or subscale with 5 plausible values (the default).

Omitted Levels: 'Multiple', 'NA', 'Omitted'

Default Conditions:
  tolower(rptsamp) == "reporting sample"

Achievement Levels:
  Basic:      262
  Proficient: 299
  Advanced:   333

Survey: NAEP
```

Think, Pair, Share: Meet your data

Explore the NAEP Primer data and discuss what subscales are available in this data set.

What are the variables related to computer usage?

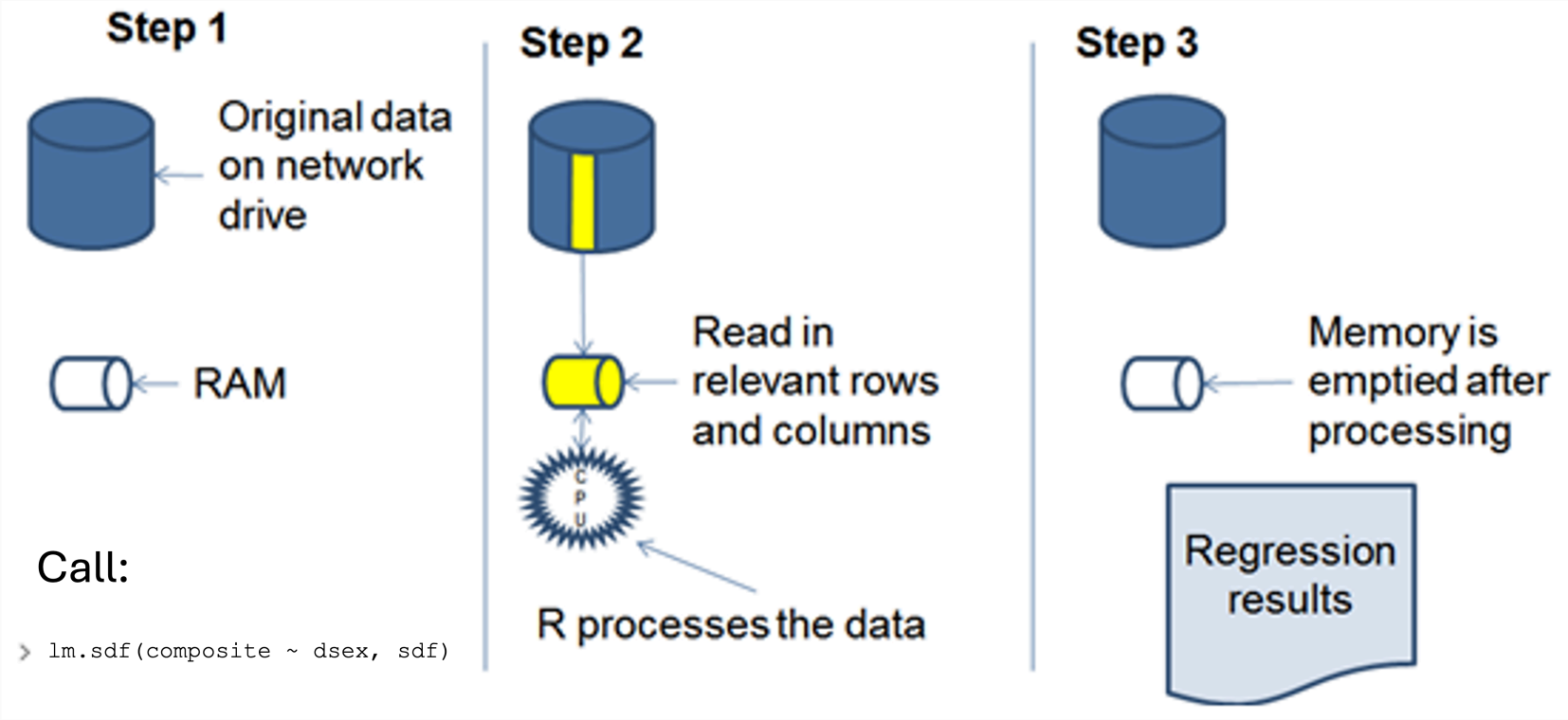
Think of your research question and find the variables in the data that could support it.

Data Manipulation

NAEP
NATIONAL ASSESSMENT OF EDUCATIONAL PROGRESS

EdSurvey Calls Network Connection

Small Memory Footprint



Data Manipulation: $

- You can manipulate variables as you would with a normal data frame, with `sdf$` and assignment works as well

```
# table(sdf$b017451)
sdf$b017451_recode <- ifelse(sdf$b017451 %in% c("Omitted",
                                                "Multiple"),
                             "Omitted or multiple",
                             as.character(sdf$b017451))
# table(sdf$b017451_recode, sdf$b017451)
```

Data Manipulation: Just Give Me the Data

- If you want to work with a data frame, you can easily get one

```
dat <- sdf[,c("b017451", "ell3")]
```

- Get all the columns

```
dat <- sdf[,colnames(sdf)]
```

Data Manipulation: getData

`getData()` reads in selected variables and sampling weights from the EdSurvey database and returns a `light.EdSurvey.data.frame` (a data-frame-like object) into the `.GlobalEnv` environment.

Functionality:

- Retrieve variables by call
- Manipulate the resulting `light.EdSurvey.data.frame`:
  - subset
  - recode
  - drop levels
- Use `EdSurvey` package functions on `light.EdSurvey.data.frames`.
- Related documentation: EdSurvey-getData.pdf.

Data Manipulation: getData

`getData()`

```
gddat <- getData(sdf, varnames = c('dsex', 'sdracem', 'b018201', 'b0
                                   'composite', 'geometry', 'origwt'
                 addAttributes = TRUE, dropOmittedLevels = FALSE)
```

A vector of variable names, including:

- `dsex` (Gender), `sdracem` (Race/ethnicity), `b018201` (Language other than English spoken in home) and `b017451` (Frequency of talk about studies at home)
- All plausible values associated with the `composite` scale (i.e., NAEP mathematics composite scale scores of 8th grade students) and the `geometry` subscale
- The sampling weight for this data frame: `origwt`, and its associated replicate weights

Data Manipulation: getData

`getData()`

```
gddat <- getData(sdf, varnames = c('dsex', 'sdracem', 'b018201', 'b0
                                   'composite', 'geometry', 'origwt'
                 addAttributes = TRUE, dropOmittedLevels = FALSE)
```

- A few important things to note:
  - "`addAttributes = TRUE`" allows the `data.frame` to be passed to `EdSurvey` package functions
  - all of the jackknife replicates are automatically returned (`srwt01` to `srwt62`)
  - "`dropOmittedLevels = FALSE`" returns variables with special values (such as multiple entries or NAs)

Data Manipulation: subset

`subset()` Returns only the data-matching elements from a variable.

- Subset the connection to the data for all analyses:

```
subsetSDF <- subset(gddat, dsex %in% c("Male"))
```

- As expected, the `subsetSDF` contains about half the rows of the original:

```
dim(sdf)
```

```
## [1] 17606   304
```

```
dim(subsetSDF)
```

```
## [1] 8486   77
```

Data Manipulation: adding dplyr

- `filter()` is a dplyr function used to subset data frames. There are many other data manipulation commands in `dplyr`. Try `?dplyr` for more information.

```
library(dplyr)
library(tidyEdSurvey)
# with tidyEdSurvey loaded, you can just use dplyr
subsetSDF <- sdf %>% filter(sdracem != "White")
head(subsetSDF[,1:10], 6)
```

```
##   ROWID       b017451_recode year cohort scrpsu   dsex iep lep ell3              sdracem
## 1    16    About once a week   36 Age 13      5   Male  No Yes  Yes Asian/Pacific Island
## 2    21           Every day   36 Age 13      5   Male Yes  No   No             Hispanic
## 3    22 Once every few weeks   36 Age 13      5   Male  No  No   No Amer Ind/Alaska Natv
## 4    26 Never or hardly ever   36 Age 13      5 Female  No  No   No             Hispanic
## 5    47  2 or 3 times a week   36 Age 13     45 Female  No  No   No Asian/Pacific Island
## 6    59 Once every few weeks   36 Age 13     56 Female  No  No   No                Black
```

Data Manipulation: caution with attributes

```
showWeights(subsetSDF)
```

```
## There is 1 full sample weight in this edsurvey.data.frame:
##   'origwt' with 62 JK replicate weights (the default).
```

```
names(attributes(subsetSDF))
```

```
##  [1] "names"              "class"              "row.names"            "userConditions"       "defaultConditions"
##  [6] "dataList"           "weights"            "pvvars"               "subject"              "year"
## [11] "assessmentCode"     "dataType"           "gradeLevel"           "achievementLevels"    "omittedLevels"
## [16] "survey"             "country"            "psuVar"               "stratumVar"           "jkSumMultiplier"
## [21] "recodes"            "dim0"               "validateFactorLabels" "cacheDataLevelName"   "reqDecimalConversion"
## [26] "fr2Path"            "dichotParamTab"     "polyParamTab"         "adjustedData"         "testData"
## [31] "scoreCard"          "scoreDict"          "scoreFunction"
```

Data Manipulation: caution with attributes

```
subsetSDF <- sdf %>%
  filter(sdracem != "White") %>%
  rebindAttributes(sdf) # adding rebindAttributes
showWeights(subsetSDF)
```

```
## There is 1 full sample weight in this edsurvey.data.frame:
##   'origwt' with 62 JK replicate weights (the default).
```

```
names(attributes(subsetSDF))
```

```
##  [1] "names"               "row.names"           "class"               "userConditions"      "defaultConditions"
##  [6] "dataList"            "weights"             "pvvars"              "subject"             "year"
## [11] "assessmentCode"      "dataType"            "gradeLevel"          "achievementLevels"   "omittedLevels"
## [16] "survey"              "country"             "psuVar"              "stratumVar"          "jkSumMultiplier"
## [21] "dim0"                "validateFactorLabels" "cacheDataLevelName" "reqDecimalConversion" "fr2Path"
## [26] "dichotParamTab"      "polyParamTab"        "adjustedData"        "testData"            "scoreCard"
## [31] "scoreDict"           "scoreFunction"       "cache"
```

Data Manipulation: caution with attributes

- Sometimes functions from packages like `dplyr` can change a `light.edsurvey.data.frame` to a `data.frame`.
- This breaks the data frames compatibility with `EdSurvey` functions.
- `rebindAttributes()` fixes this by re-adding `light.edsurvey.data.frame` attributes to the returned `data.frame`.

Data Manipulation: recoding variables

You can use `recode.sdf()` but another option is `ifelse` to recode:

```
sdf$b017451_edited <- ifelse(sdf$b017451 %in% c("Never or hardly eve
                             "Infrequently",
                             sdf$b017451)
#table(sdf$b017451_edited, sdf$b017451)
```

Think, Pair, Share: Data Manipulation

How would you recode `t091502` (and what is that variable?) or another variable of your own interest in the NAEP Primer data?

Data Visualization: loading ggplot

`install.package` and `library` ggplot2:

```
# R code for loading ggplot2
install.packages("ggplot2") # only necessary once, or if updating

# R code for loading ggplot2
library(ggplot2)
library(tidyEdSurvey)
```

Data Visualization: ggplot function

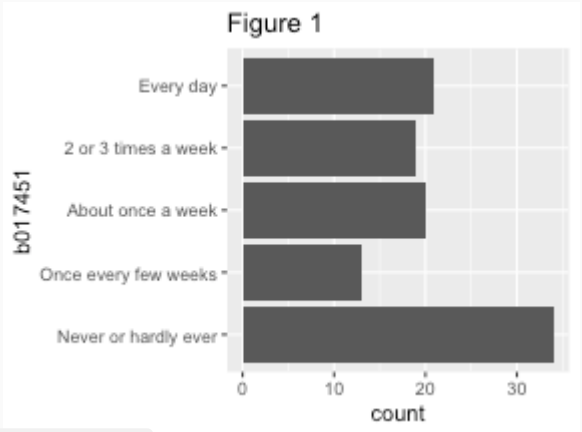
```
ggplot(data=sdf, aes(x=b017451)) + geom_bar() +
coord_flip() + labs(title = "Figure 1")
```

Functionality

- supply data to plot, and aesthetic mapping (axis variables and groupings)
- add layers with `+` to create different kinds of graphs or add other graph elements:
  - `geom_bar()`: bar charts
  - `geom_histogram()`: histograms
  - `coord_flip()`: flip coordinates
  - `labs()`: add labels
- Related documentation: EdSurvey Vignette or ggplot2
- Note: The charts presented are all **quick and dirty**—unless noted, they are not weighted, and in the case of plausible values, only one set is used.

Data Visualization: plotting categorical

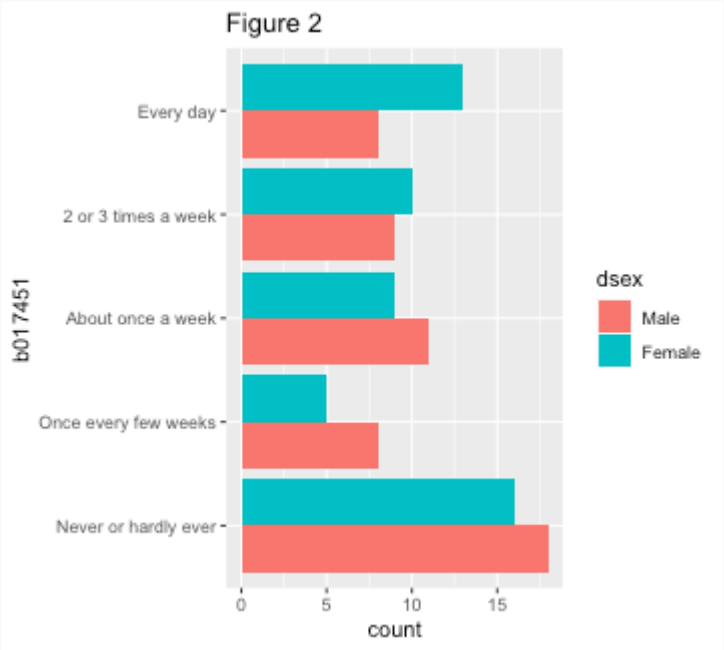
```r
library(tidyEdSurvey)
# R code generating Figure 1
bar1 <- ggplot(data=sdf, aes(x=b017451)) +
  geom_bar() +
  coord_flip() +
  labs(title = "Figure 1")
bar1
```



Note: loading `tidyEdSurvey` a few slides ago allows us to use `sdf` directly in `ggplot2` calls, without that you'll need to change the code to use `gddat`.

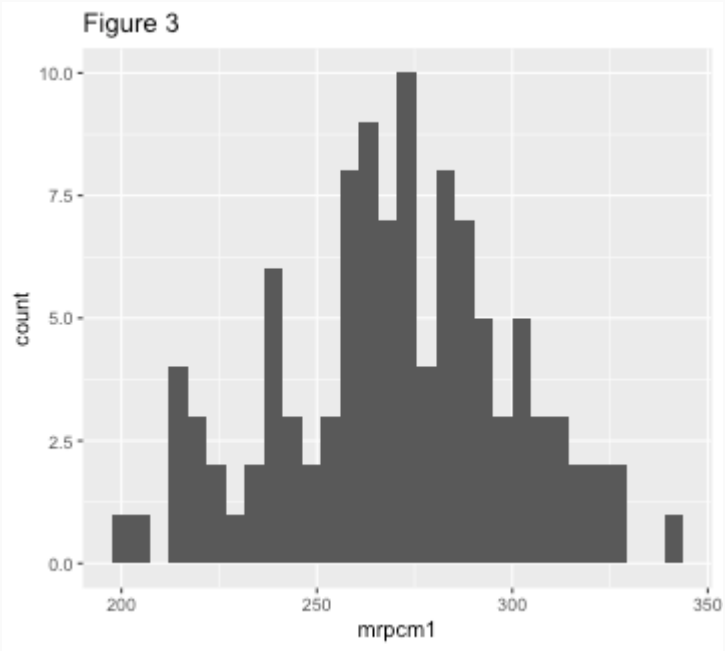
Data Visualization: catorgical by group

```
# R code for generating Figure 2
bar2 <- ggplot(data=sdf, aes(x=b017451, fill=dsex)) +
  geom_bar(position='dodge') +
  coord_flip() +
  labs(title = "Figure 2")
bar2
```



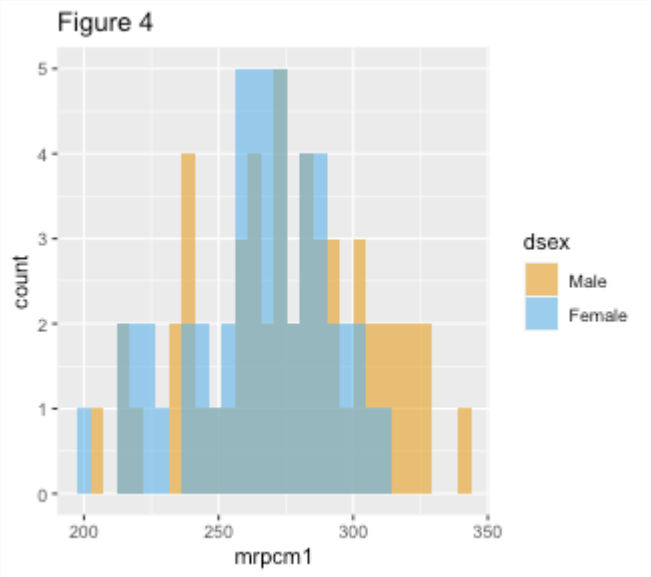
Data Visualization: plotting continuous

```r
# R code for generating Figure 3
hist1 <- ggplot(sdf, aes(x=mrpcm1)) +
    geom_histogram(bins=30) + # geom type changed
    labs(title = "Figure 3")
hist1
```



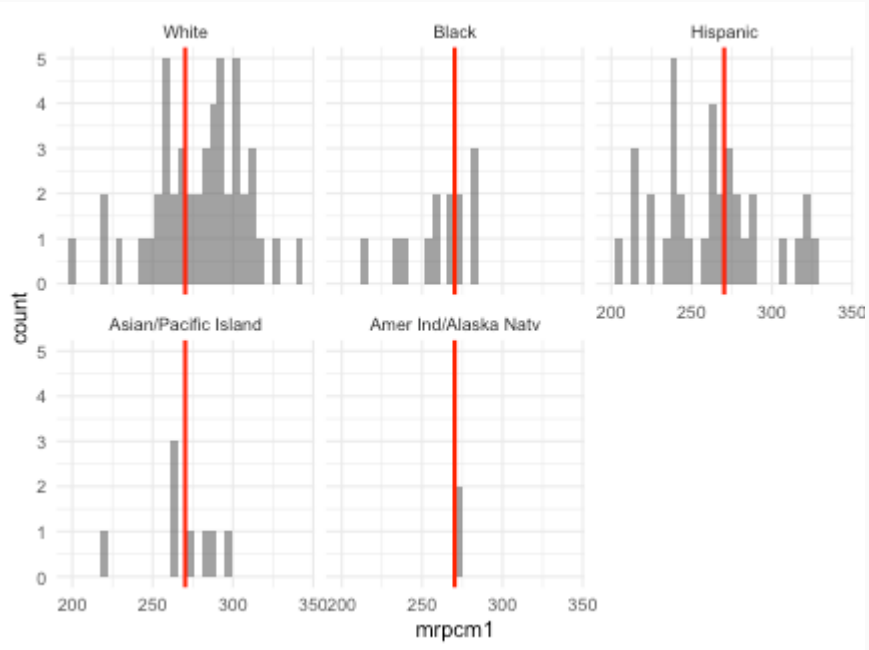
Data Visualization: histogram with grouping

```
hist2 <- ggplot(sdf, aes(x=mrpcm1, fill = dsex)) +
     geom_histogram(position="identity", alpha = 0.6, bins=30) + # a
     labs(title = "Figure 4") +
     scale_fill_manual(values=c("#E69F00", "#56B4E9")) # scale fill
hist2
```



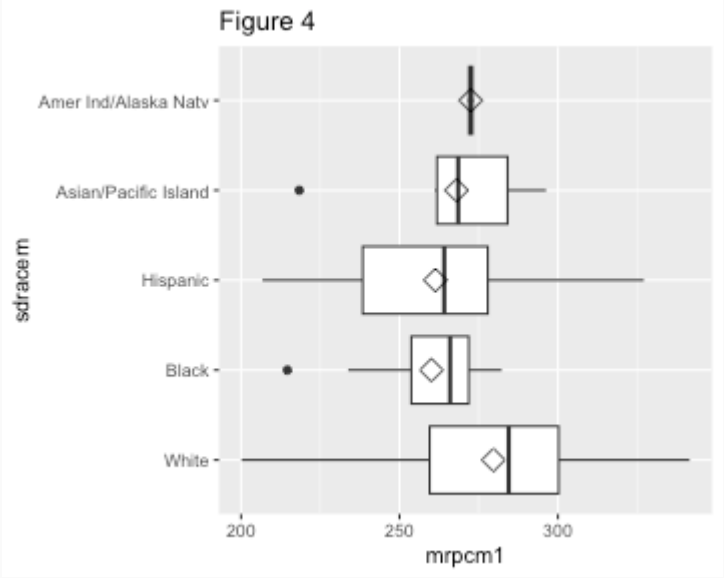
Data Visualization: histogram with facets

```
ggplot(sdf, aes(x=mrpcm1)) +
     geom_histogram(position="identity", alpha = 0.6, bins=30) + # a
     geom_vline(aes(xintercept = mean(mrpcm1)),col='red',linewidth=1
     theme_minimal() +
     labs(y = "count") +
     facet_wrap(~ sdracem, ncol = 3 ) # great graph faceted by group
```



Data Visualization: box-plots

```r
# R code for generating Figure 4
box1 <- ggplot(sdf, aes(x=sdracem, y=mrpcm1)) +
  geom_boxplot() +
  stat_summary(fun=mean, geom="point", shape=23, size=4) + # transor
  coord_flip() +
  labs(title = "Figure 4")
box1
```



Data Visualization: box-plots

```r
# R code for generating Figure 4
box1 <- ggplot(sdf, aes(x=sdracem, y=mrpcm1), fill = dsex) +
  geom_boxplot(aes(fill = dsex)) +
  stat_summary(fun=mean, geom="point", shape=23, size=4) + # transor
  coord_flip() +
  labs(title = "Figure 4")
box1
```