Welcome to the NDACAN Summer training series!

- August 7th, 2019 - Strategies for Managing Data
- Presenter: Frank Edwards, Ph.D.
- The session will begin at 12pm.
- This session is being recorded.
NDACAN Summer Training series

National Data Archive on Child Abuse and Neglect
Bronfenbrenner Center for Translational Research
Cornell University
NDACAN Summer Training series schedule

• July 17\textsuperscript{th}, 2019 - Introduction to NDACAN
• July 24\textsuperscript{th}, 2019 - Overview of NCANDS Data
• July 31\textsuperscript{st}, 2019 - Overview of AFCARS and NYTD Data
• \textbf{August 7\textsuperscript{th}, 2019 - Strategies for Managing Data}
• August 14\textsuperscript{th}, 2019 - Linking NCANDS, AFCARS, and NYTD Data
• August 21\textsuperscript{st}, 2019 - Concluding Session
Session Overview

- Secondary data analysis requires a high level of data management skills, such as reshaping and collapsing data.
- This is especially true for larger datasets, such as the NCANDS dataset, where running code can be time and computing power intensive.
- This training should help you increase your data management skills and ease some of the difficulties in conducting secondary data analysis with large administrative datasets.
Strategies for managing data

Frank Edwards
• Use the R statistical programming language and sql-like commands to:
  • Aggregate (summarize) data to geographic or time units of analysis
  • Reshape data (wide, long)
  • Append variables or observations to existing data
  • Merge (join) datasets
  • Draw a random sample
Before we begin

• We use the free and open source R statistical programming language, and I use the RStudio IDE.
• You can install R from cran-rproject.org, and RStudio from rstudio.com
• I also use the tidyverse packages for data manipulation. Run `install.packages('tidyverse')` from the R console to install them
• These techniques are adapted from SQL principals, and share syntax and theory with SQL data management and wrangling principals
• These demos use AFCARS 2017 child file, with identifying variables removed
Set up

```r
library(tidyverse)

afcars_17 <- read_tsv("FC2017v2.tab")

afcars_17 <- afcars_17 %>%
  select(St, SEX, AgeAtEnd, InAtEnd, Entered) %>%
  filter(!is.na(SEX))
```
What are the natural groupings in this data?

<table>
<thead>
<tr>
<th>St</th>
<th>SEX</th>
<th>AgeAtEnd</th>
<th>InAtEnd</th>
<th>Entered</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>AL</td>
<td>1</td>
<td>10</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>AL</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>AL</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>AL</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>AL</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Summarizing the data

table1<-afcars_17%>%
  summarise(MeanAge = mean(AgeAtEnd),
             PctMale = sum(SEX==1,
                          na.rm=TRUE)/n(),
             TotalEntries=sum(Entered),
             Caseload = sum(InAtEnd))

<table>
<thead>
<tr>
<th>MeanAge</th>
<th>PctMale</th>
<th>TotalEntries</th>
<th>Caseload</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.098175</td>
<td>0.5157354</td>
<td>269732</td>
<td>442681</td>
</tr>
</tbody>
</table>

National Data Archive on Child Abuse and Neglect
Grouping and summarizing: by sex

table1 <- afcars_17 %>%
  group_by(SEX) %>%
  summarise(MeanAge = mean(AgeAtEnd),
            TotalEntries = sum(Entered),
            Caseload = sum(InAtEnd))

<table>
<thead>
<tr>
<th>SEX</th>
<th>MeanAge</th>
<th>TotalEntries</th>
<th>Caseload</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.021688</td>
<td>138083</td>
<td>228589</td>
</tr>
<tr>
<td>2</td>
<td>8.179632</td>
<td>131649</td>
<td>214092</td>
</tr>
</tbody>
</table>
Grouping and summarizing: by state

table1 <- afcars_17 %>%
group_by(St) %>%
summarise(
  MeanAge = mean(AgeAtEnd),
  TotalEntries = sum(Entered),
  Caseload = sum(InAtEnd)
)

<table>
<thead>
<tr>
<th>St</th>
<th>MeanAge</th>
<th>TotalEntries</th>
<th>Caseload</th>
</tr>
</thead>
<tbody>
<tr>
<td>AK</td>
<td>7.417093</td>
<td>1328</td>
<td>2766</td>
</tr>
<tr>
<td>AL</td>
<td>8.177410</td>
<td>4095</td>
<td>5631</td>
</tr>
<tr>
<td>AR</td>
<td>7.167915</td>
<td>3778</td>
<td>4776</td>
</tr>
<tr>
<td>AZ</td>
<td>7.758218</td>
<td>10054</td>
<td>15028</td>
</tr>
<tr>
<td>CA</td>
<td>8.283262</td>
<td>28015</td>
<td>51867</td>
</tr>
<tr>
<td>CO</td>
<td>9.352652</td>
<td>5134</td>
<td>5704</td>
</tr>
</tbody>
</table>
Grouping and summarizing: by state and sex

```r
table1<-afcars_17%>
  group_by(St, SEX)%>
  summarise(MeanAge = mean(AgeAtEnd),
             TotalEntries=sum(Entered),
             Caseload = sum(InAtEnd))
```

<table>
<thead>
<tr>
<th>St</th>
<th>SEX</th>
<th>MeanAge</th>
<th>TotalEntries</th>
<th>Caseload</th>
</tr>
</thead>
<tbody>
<tr>
<td>AK</td>
<td>1</td>
<td>7.415414</td>
<td>643</td>
<td>1375</td>
</tr>
<tr>
<td>AK</td>
<td>2</td>
<td>7.418787</td>
<td>685</td>
<td>1391</td>
</tr>
<tr>
<td>AL</td>
<td>1</td>
<td>7.747501</td>
<td>2070</td>
<td>2859</td>
</tr>
<tr>
<td>AL</td>
<td>2</td>
<td>8.625935</td>
<td>2025</td>
<td>2772</td>
</tr>
<tr>
<td>AR</td>
<td>1</td>
<td>6.961627</td>
<td>1907</td>
<td>2425</td>
</tr>
<tr>
<td>AR</td>
<td>2</td>
<td>7.379384</td>
<td>1871</td>
<td>2351</td>
</tr>
</tbody>
</table>
Moving from long to wide: Mean Age

```r
wide1 <- table1 %>%
  select(St, SEX, MeanAge) %>%
  spread(key = SEX, value = MeanAge)
```

<table>
<thead>
<tr>
<th>St</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>AK</td>
<td>7.415414</td>
<td>7.418787</td>
</tr>
<tr>
<td>AL</td>
<td>7.747501</td>
<td>8.625935</td>
</tr>
<tr>
<td>AR</td>
<td>6.961627</td>
<td>7.379384</td>
</tr>
<tr>
<td>AZ</td>
<td>7.832072</td>
<td>7.679496</td>
</tr>
<tr>
<td>CA</td>
<td>8.146840</td>
<td>8.424892</td>
</tr>
<tr>
<td>CO</td>
<td>9.984347</td>
<td>8.502218</td>
</tr>
</tbody>
</table>
Convert full data to wide by sex

```
wide_age <- table1 %>% 
  select(St, SEX, MeanAge) %>% 
  spread(key = SEX, value = MeanAge, 
         sep = "age")

wide_caseload <- table1 %>% 
  select(St, SEX, Caseload) %>% 
  spread(key = SEX, value = Caseload, 
         sep = "caseload")

wide_entries <- table1 %>% 
  select(St, SEX, TotalEntries) %>% 
  spread(key = SEX, value = TotalEntries, 
         sep = "entries")
```
How to merge them?

```r
names(wide_age)
## [1] "St" "SEXage1" "SEXage2"

names(wide_caseload)
## [1] "St" "SEXcaseload1" "SEXcaseload2"

names(wide_entries)
## [1] "St" "SEXentries1" "SEXentries2"
```
Use St as our key column

```r
wide_merge <- left_join(
  wide_age,
  wide_cases
)
## Joining, by = "St"

wide_merge <- left_join(
  wide_merge,
  wide_entries
)
## Joining, by = "St"
```
Adding population data

*From seer.cancer.gov/popdata*

```r
pop <- read_fwf("us.1990_2017.singleages.adjusted.txt",
                 fwf_widths(c(4, 2, 2, 3,
                              2, 1, 1, 1,
                              2, 8),
                 c("year", "state", "st_fips",
                   "cnty_fips", "reg", "race",
                   "hisp", "sex", "age",
                   "pop")))

mutate(age = as.numeric(age),
       pop = as.numeric(pop))

filter(age < 21, year == 2017)

rename(st = state)
```
Aggregate it to state, year, seX

\[
\text{pop}_{\text{st}} \leftarrow \text{pop} \%
\text{group_by(} \text{st, sex, age}) \%
\text{summarise(pop = sum(pop))}
\]
Let’s get per capita entry rates by age and sex

```r
percap <- left_join(
  afcars_17 %>%
  group_by(SEX, St, AgeAtEnd) %>%
  summarise(entries = sum(Entered)) %>%
  rename(st = St, sex = SEX, age = AgeAtEnd),
  pop_st) %>%
filter(age < 19)

## Joining, by = c("sex", "st", "age")
```
What did this make?

<table>
<thead>
<tr>
<th>sex</th>
<th>st</th>
<th>age</th>
<th>entries</th>
<th>pop</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AK</td>
<td>0</td>
<td>91</td>
<td>5673</td>
</tr>
<tr>
<td>1</td>
<td>AK</td>
<td>1</td>
<td>59</td>
<td>5595</td>
</tr>
<tr>
<td>1</td>
<td>AK</td>
<td>2</td>
<td>45</td>
<td>5369</td>
</tr>
<tr>
<td>1</td>
<td>AK</td>
<td>3</td>
<td>56</td>
<td>5433</td>
</tr>
<tr>
<td>1</td>
<td>AK</td>
<td>4</td>
<td>39</td>
<td>5555</td>
</tr>
<tr>
<td>1</td>
<td>AK</td>
<td>5</td>
<td>44</td>
<td>5344</td>
</tr>
</tbody>
</table>
Calculate per capita rates

```r
percap <- percap %>%
  mutate(entries_pc = entries / pop * 1e3)

ggplot(percap, aes(x = age,
              y = entries_pc,
              lty = factor(sex))) +
  geom_line() +
  facet_wrap(~st)
```
Age of entry by sex for every state in the US
Draw a random sample

- Useful when working with large datasets to test models (e.g. NCANDS, AFCARS)
- Random subsets make code development more efficient, as computation on large datasets can take a LONG time

```r
sample <- afcars %>%
  sample_frac(size = 0.1)
## Draw a 10 percent random sample
## Model placement stability for latest episode
Model_1 <- glm(NumPlep ~ AgeAtEnd + Entered + Sex,
               data = sample, family = "poisson")
```
QUESTIONS?
FRANK EDWARDS
NDACAN STATISTIAN
ASSISTANT PROFESSOR, Rutgers
frank.edwards@rutgers.edu
Next week...

August 14, 2019
Linking NCANDS, AFCARS, and NYTD Data
Presenter: Michael Dineen