Accessing NHANES Data

This page contains 3 ways of accessing NHANES data inside R.

1. Through the dockerized container created by the CCB.
2. Through the `nhanesA` package.
3. Through downloading individual files from the CDC website.

Through the Docker container (reccomended)

These steps will allow you to access the NHANES data via a builtin

Connect to RStudio in your browser as shown in the Accessing NHANES with Docker page.

Installing Phonto

First we need to install the Phonto package, which is the software used to access the NHANES database inside the container.

In the R console, run the command:

```
devtools::install_github("ccb-hms/phonto")
```

If successful you should see output similar to:

```
Downloading GitHub repo ccb-hms/phonto@HEAD
  checking for file ‘/tmp/RtmpjKcsPx/remotes8977d6e909/ccb-hms-phonto-121d255/DESCRIPTION’ .
  preparing ‘phonto’:
  checking DESCRIPTION meta-information
  checking for LF line-endings in source and make files and shell scripts
  checking for empty or unneeded directories
  building ‘phonto_0.0.0.0069.tar.gz’

* installing *source* package ‘phonto’ ...
** using staged installation
** R
** data
*** moving datasets to lazyload DB
** inst
** byte-compile and prepare package for lazy loading
EpiConductor Container Version: v0.0.4
Data Collection Date: 2023-06-28
```

```
** help
*** installing help indices
** building package indices
** installing vignettes
** testing if installed package can be loaded from temporary location
** testing if installed package can be loaded from final location
** testing if installed package keeps a record of temporary installation path
* DONE (phonto)
```

⚠ **Timeout Errors**

You may get a timeout error such as this:

```
> devtools::install_github("ccb-hms/phonto")
Error: Failed to install 'phonto' from GitHub:
  Timeout was reached: [api.github.com] Resolving timed out after 10000 milliseconds
```

Or a longer error related to login timing out like this:

```
> devtools::install_github("ccb-hms/phonto")
Downloading GitHub repo ccb-hms/phonto@HEAD
   checking for file ‘/tmp/RtmpjKcsPx/remotes895e1b4f91/ccb-hms-phonto-121d255/DESCRIPTIO
   preparing ‘phonto’:
   checking DESCRIPTION meta-information ...
   checking for LF line-endings in source and make files and shell scripts
   checking for empty or unneeded directories
   building ‘phonto_0.0.0.0069.tar.gz’

* installing *source* package ‘phonto’ ...
** using staged installation
** R
** data
*** moving datasets to lazyload DB
** inst
** byte-compile and prepare package for lazy loading
Error : nanodbc/nanodbc.cpp:1021: 00000: [Microsoft][ODBC Driver 17 for SQL Server]Login
Error: unable to load R code in package ‘phonto’
Execution halted
ERROR: lazy loading failed for package ‘phonto’
* removing ‘/usr/local/lib/R/library/phonto’
Warning message:
In i.p(...) :
  installation of package ‘/tmp/RtmpjKcsPx/file8930842adb/phonto_0.0.0.0069.tar.gz’ had
```

If you get one of these errors, please retry the installation command. It may take 2-3 tries for the docker container to establish a connection with Github.

Accessing the Phonto vignettes

Like many R packges, the best way to learn how to use Phonto is to go through the included vignettes.

You can access the vignettes inside the Docker container by pulling them from Github.

RStudio has an integrated Git user interface that makes it very easy to use both Git and GitHub. RStudio has integrated Git support which helps to streamline this process. To get a copy of phonto in RStudio do the following:

1. Click `File` → `New Project`
2. Select `Version Control` → `Git`

3. For the URL choose: https://github.com/ccb-hms/phonto.git
4. You can choose the name of the project directory.
5. Choose the folder in which you want to store the R project and Git (you can put this either inside your home directory or navigate to an attached volume).
6. Click `Create Project`
7. Check the `Files` tab to see if you have successfully created the project.
8. Navigate to `phonto/vignettes` or directly open `phonto/vignettes/quick_start.Rmd` or `phonto/vignettes/VariableClassification.Rmd`.

Whenever you are working in an RStudio project that has a dedicated Git repository, you can interact with Git through the Git tab (same pane as Environment tab)

Using Phonto

While the above vignettes include a variety of examples in how to use phonto to access the NHANES data, let's take a look at how we accessed the data used in the Beheshti paper.

Searching NHANES

To start, if we don't know the variable we're interested in, we can search for keywords using ‘`nhanesSearch`.

```
hba1c = nhanesSearch("glycohemoglobin", ignore.case=TRUE, ystart = 2005, ystop=2010, names
```

```
 Variable.Name Variable.Description Data.File.Name Data.File.Description Begin.Year EndYear
1         LBXGH  Glycohemoglobin (%)          GHB_D       Glycohemoglobin       2005    2006
2         LBXGH  Glycohemoglobin (%)          GHB_E       Glycohemoglobin       2007    2008
3         LBXGH  Glycohemoglobin (%)          GHB_F       Glycohemoglobin       2009    2010
```

Alternatively, we can look up table definitions.

```
res = nhanesSearchTableNames("DEM", details=TRUE)
```

```
  TableName     Years
1      DEMO 1999-2000
2    DEMO_B 2001-2002
3    DEMO_C 2003-2004
4    DEMO_D 2005-2006
5    DEMO_E 2007-2008
6    DEMO_F 2009-2010
```

```
7     DEMO_G 2011-2012
8     DEMO_H 2013-2014
9     DEMO_I 2015-2016
10    DEMO_J 2017-2018
```

And then check column names.

```
nhanesColnames("DEMO_D")
```

```
 [1] "SEQN"     "SDDSRVYR" "RIDSTATR" "RIDEXMON" "RIAGENDR" "RIDAGEYR" "RIDAGEMN" "RIDAGEEX"
[13] "DMDYRSUS" "DMDEDUC3" "DMDEDUC2" "DMDSCHOL" "DMDMARTL" "DMDHHSIZ" "DMDFMSIZ" "INDHHINC"
[25] "DMDHRAGE" "DMDHRBRN" "DMDHREDU" "DMDHRMAR" "DMDHSEDU" "SIALANG"  "SIAPROXY" "SIAINTRP"
[37] "MIAPROXY" "MIAINTRP" "AIALANG"  "WTINT2YR" "WTMEC2YR" "SDMVPSU"  "SDMVSTRA" "SEQN"
[49] "RIDRETH1" "DMQMILIT" "DMDBORN"  "DMDCITZN" "DMDYRSUS" "DMDEDUC3" "DMDEDUC2" "DMDSCHOL"
[61] "DMDHRGND" "DMDHRBRN" "DMDHREDU" "DMDHRMAR" "DMDHSEDU" "SIALANG"  "SIAPROXY" "SIAINTRP"
[73] "MIAPROXY" "MIAINTRP" "AIALANG"  "RIDAGEYR" "RIDAGEMN" "RIDAGEEX" "DMDHHSIZ" "DMDFMSIZ"
[85] "SDMVPSU"  "SDMVSTRA"
```

Defining a query

We can define a quary to NHANES by creating a list of column names for each table we're interested in. It is reccomended to only get a single servey cycle with a single query.

```
cols_d = list(DEMO_D= c("RIDAGEYR","RIAGENDR","RIDRETH1",
                        "DMDBORN", "INDFMPIR", "SDMVPSU",
                        "SDMVSTRA", "WTINT2YR", "WTMEC2YR"),
              OHX_D = c("OHXDECAY", "OHXREST"),
              GLU_D = c("LBXGLU", "WTSAF2YR"), GHB_D = "LBXGH",
              BMX_D= "BMXBMI"
)
cols_e = list(DEMO_E= c("RIDAGEYR","RIAGENDR","RIDRETH1",
                        "DMDBORN2", "INDFMPIR", "SDMVPSU",
                        "SDMVSTRA", "WTINT2YR", "WTMEC2YR"),
              OHX_E = c("OHXDECAY", "OHXREST"),
              GLU_E = c("LBXGLU", "WTSAF2YR"),
              GHB_E = "LBXGH",
              BMX_E = "BMXBMI"
)

cols_f = list(DEMO_F= c("RIDAGEYR","RIAGENDR","RIDRETH1",
                        "DMDBORN2", "INDFMPIR", "SDMVPSU",
                       "SDMVSTRA", "WTINT2YR", "WTMEC2YR"),
              OHXDEN_F = c("OHXDECAY", "OHXREST"),
              GLU_F = c("LBXGLU", "WTSAF2YR"),
              GHB_F = "LBXGH",
              BMX_F = "BMXBMI"
)
```

Getting Metadata

We can get metadata on each column by calling `dataDescription`. Here we combine years, but note that only unique variable names and variable descriptions are returned, i.e., if the list contains the same questionnaire/variables across different survey years, and if all metadata is consistent, then only one row for this variable will be return.

```
all_cols <- c(cols_d, cols_e, cols_f)
metadata <- dataDescription(all_cols)
tail(metadata)
```

```
   VariableName                            SASLabel
19     DMDBORN2           Country of Birth - Recode                              In what cou
20     INDFMPIR   Ratio of family income to poverty                        A ratio of family
21      SDMVPSU          Masked Variance Pseudo-PSU     Masked Variance Unit Pseudo-PSU varia
22     SDMVSTRA      Masked Variance Pseudo-Stratum Masked Variance Unit Pseudo-Stratum varia
23     WTINT2YR Full Sample 2 Year Interview Weight
24     WTMEC2YR  Full Sample 2 Year MEC Exam Weight                    Both Interviewed and
```

Getting Data

We can use `jointQuery` to get data from NHANES. This will return all columns in the query already translated and combined into a single dataframe for us.

```
base_df_d <- jointQuery(cols_d)
base_df_e <- jointQuery(cols_e)
base_df_f <- jointQuery(cols_f)

head(base_df_d)
```

```
   SEQN RIDAGEYR RIAGENDR           RIDRETH1                        DMDBORN INDFMPIR SDMVPSU
1 31127        0     Male Non-Hispanic White "Born in 50 US States or Washi     0.75       2
2 31128       11   Female Non-Hispanic Black "Born in 50 US States or Washi     0.77       1
```

```
3 31129       15     Male Non-Hispanic Black "Born in 50 US States or Washi     2.71        1
4 31130       85   Female Non-Hispanic White "Born in 50 US States or Washi     1.99        2
5 31131       44   Female Non-Hispanic Black "Born in 50 US States or Washi     4.65        1
6 31132       70     Male Non-Hispanic White "Born in 50 US States or Washi     5.00        2
```

Through nhanesA

With `nhanesA`, we can easily download entire tables from NHANES. However, there are some extra processing steps we'll have to perform compared to using the dockerized database. You can learn more about using `nhanesA` here.

```
library(nhanesA)
library(tidyverse)
```

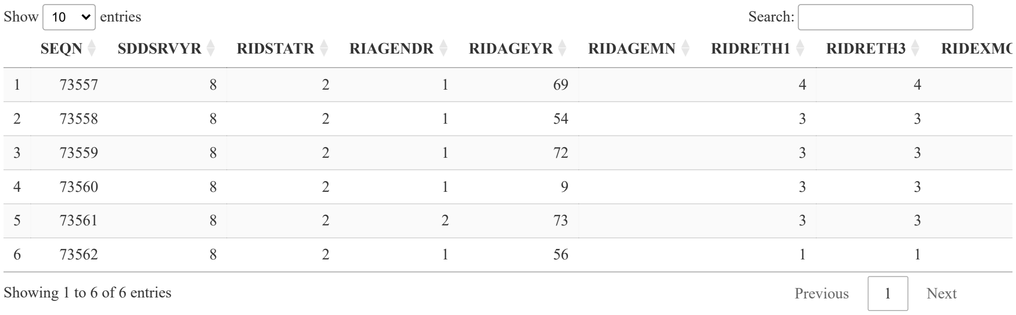
```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.2     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.0
v ggplot2   3.4.2     v tibble    3.2.1
v lubridate 1.9.2     v tidyr     1.3.0
v purrr     1.0.1
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
library(DT)
# Get data with nahnesA
DEMO_H = nhanes('DEMO_H')
DEMO_I = nhanes('DEMO_I')
DPQ_H = nhanes('DPQ_H')
DPQ_I = nhanes('DPQ_I')

# Append Files
DEMO <- bind_rows(DEMO_H, DEMO_I)
DPQ <- bind_rows(DPQ_H, DPQ_I)

datatable(head(DEMO))
```

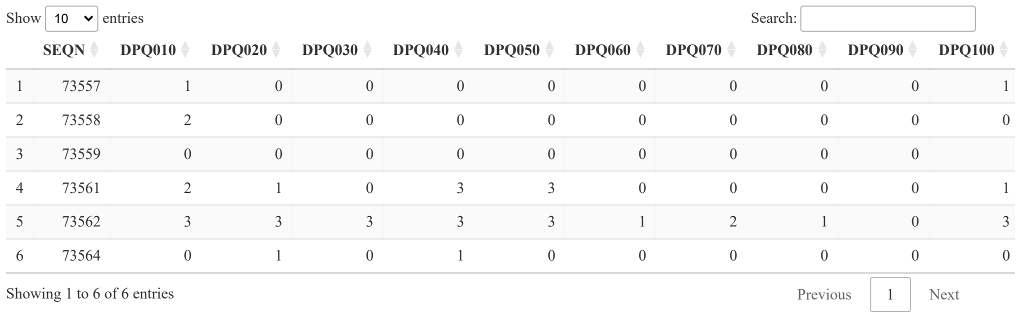
```
PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is installe
```



	SEQN	SDDSRVYR	RIDSTATR	RIAGENDR	RIDAGEYR	RIDAGEMN	RIDRETH1	RIDRETH3	RIDEXM
1	73557	8	2	1	69		4	4	
2	73558	8	2	1	54		3	3	
3	73559	8	2	1	72		3	3	
4	73560	8	2	1	9		3	3	
5	73561	8	2	2	73		3	3	
6	73562	8	2	1	56		1	1	

Showing 1 to 6 of 6 entries

```
datatable(head(DPQ))
```



	SEQN	DPQ010	DPQ020	DPQ030	DPQ040	DPQ050	DPQ060	DPQ070	DPQ080	DPQ090	DPQ100
1	73557	1	0	0	0	0	0	0	0	0	1
2	73558	2	0	0	0	0	0	0	0	0	0
3	73559	0	0	0	0	0	0	0	0	0	
4	73561	2	1	0	3	3	0	0	0	0	1
5	73562	3	3	3	3	3	1	2	1	0	3
6	73564	0	1	0	1	0	0	0	0	0	0

Showing 1 to 6 of 6 entries

There are a few differences between this data and the processed data we've been using. First, we have to `join` the DEMO and DPQ tables. We'll learn more about joining or merging data in the last week of class.

Second, the values in the raw NHANES tables are numeric encodings for each variable. With nhanesA we can lookup the code using `nhanesCodebook` and convert numeric codes using `nhanesTranslate`.

```
nhanesCodebook('DEMO_H', 'RIAGENDR')
```

```
$`Variable Name:`
[1] "RIAGENDR"

$`SAS Label:`
[1] "Gender"

$`English Text:`
[1] "Gender of the participant."

$`Target:`
[1] "Both males and females 0 YEARS -\r 150 YEARS"

$RIAGENDR
# A tibble: 3 x 5
  `Code or Value` `Value Description` Count Cumulative `Skip to Item`
  <chr>           <chr>               <int>      <int> <lgl>
1 1               Male                 5003       5003 NA
2 2               Female               5172      10175 NA
3 .               Missing                 0      10175 NA
```

```
nhanesTranslate(DEMO_H)
```

```
Column name is required

NULL
```

Downloading individual files

If all else fails, individual files can be downloaded from the CDC website and read into R using the foreign package. This example is taken from an example analysis put out by the CDC here.

```
#' Prevalence of Depression Among Adults Aged 20 and Over: United States, 2013-2016
#' Brody DJ, Pratt LA, Hughes JP. Prevalence of Depression Among Adults Aged 20 and Over:
#' States, 2013-2016. NCHS Data Brief. No 303. Hyattsville, MD: National Center for Health

#' # Data preparation
# Download & Read SAS Transport Files
# Demographic (DEMO)
download.file("https://wwwn.cdc.gov/nchs/nhanes/2013-2014/DEMO_H.XPT", tf <- tempfile(), m
DEMO_H <- foreign::read.xport(tf)[,c("SEQN","RIAGENDR","RIDAGEYR","SDMVSTRA","SDMVPSU","WT
download.file("https://wwwn.cdc.gov/nchs/nhanes/2015-2016/DEMO_I.XPT", tf <- tempfile(), m
DEMO_I <- foreign::read.xport(tf)[,c("SEQN","RIAGENDR","RIDAGEYR","SDMVSTRA","SDMVPSU","WT

# Mental Health - Depression Screener (DPQ)
download.file("http://wwwn.cdc.gov/nchs/nhanes/2013-2014/DPQ_H.XPT", tf <- tempfile(), mod
DPQ_H <- foreign::read.xport(tf)
download.file("http://wwwn.cdc.gov/nchs/nhanes/2015-2016/DPQ_I.XPT", tf <- tempfile(), mod
DPQ_I <- foreign::read.xport(tf)
```