Exercise 2

Educational opportunity in the NLSY

For this exercise, you will be working with an extract from the 1997 *National Longitudinal Survey of Youth* (NLSY97),[1] a major panel study tracking about 9,000 youth who were between 12 and 15 on December 31, 1996. NLSY97 is intended to follow young adults both as they complete their education and over their working lives. It's also a pretty messy data set to work with.

1. **Create your exercise directory**
    a. On your computer, create a folder for this seminar exercise.
    b. Download the NLSY97 data and extract the files into your seminar exercise folder.
2. **Perform the included processing of the NLSY data file**
    a. The NLSY provides some preliminary formatting to the raw data file (*nsl97.dat*) via a R script file (here *nsly97.R*). Open this script file in RStudio.
    b. To run the script, you will first need to uncomment the **setwd()** function and provide the path to your exercise directory.
    c. At the end of the script, uncomment the necessary lines to "rename variables using Qnames instead of Reference Numbers."
    d. Export the data frame *new_data* as *nlsy97.rds*.
    e. Save the *nls97.R* and begin working in a new working R script file, wherein you will complete the remainder of the exercise.
3. **Load the data and view the structure**
    a. In the new script file you've created, import *nlsy97.rds* dataframe as *nlsy97*.
    b. Look at the structure of the data frame using the **glimpse()** and **str()** functions.
    c. The list of column names is a bit hard to read in either approach - and is even truncated with **str()**. To make it easier to scan the column names, save them to a new data object formatted as a **tibble**.
4. **"Tidy" the dataset**: A key principle of data analysis is that data should follow the "tidy" data structure, wherein each row of a data set corresponds to an observation, and each column a variable.
    a. Notice that there are both time-varying and time-invariant variables. Rename the variables that are time-invariant as follows (to make your life easier in the next step):

Old Name	New Name
PUBID_1997	personid
KEY_SEX_1997	sex
KEY_RACE_ETHNICITY_1997	race
KEY_BDATE_Y_1997	birthyr
CV_HGC_RES_MOM_1997	motheredyrs

    b. Begin to tidy the data by using the **gather()** function to gather all variables that are time-varying

[1] Available at https://www.nlsinfo.org/investigator/. Keep in mind that this exercise is strictly pedagogical - we ignore things like missing value codes in the data, hence there are obvious errors.

into a key and value column. To do that, use the **gather()** function on all columns except those mentioned in step (d), with *variable* as the name of the key column and *value* as the name of the value column.

c. All time-varying variables should now appear in a single column and all corresponding values in another. But notice that we have another issue – each year of a variable has a different name. To fix this:

(i) Create a *year* column, with values that are the numbers from the *variable* column. You can do so as follows:

```
nlsy97$year <- str_extract(nlsy97$variable, "[0-9]+")
```

(ii) Notice that some of the year values are written in the wrong format. Recode the following values of *year*:

Old Value	New Value
9293	1993
9394	1994
9495	1995
9596	1996
9697	1997
9798	1998
9899	1999
9900	2000
0001	2001
0102	2002
0203	2003
0304	2004

(iii) Replace the numbers in the *variable* column with an empty string. You can do so as follows:

```
nlsy97$variable <- str_replace(nlsy97$variable, "[0-9]+","")
```

d. Now that the variable names and years are consistent, finish "tidying" the data by **spreading** each of the variables in *variable* to its own column.

5. **Rename and select variables**

a. Rename the time-varying variables as follows:

Old Name	New Name
CV_CENSUS_REGION_	region
CV_INCOME_GROSS_YR_	parentincome
CV_SCHOOL_TYPE_	schooltype
CV_HGC_EVER_EDT_	highestgrade
TRANS_AB_AY_HSTR	absences
TRANS_CRD_GPA_YR___HSTR	gpa

b. Keep only the variables you've renamed in 4(a) and 5(a) as well as the year variable.

6. **Finalize the data set**

a. Ensure that each variable has the correct data type. This means you'll probably have to change the data type for the following variables: *personid*, *year*, *sex*, *race*, *region*, and *schooltype*.

b. Create the following new variables:

- $age = year - birthyr$
- $gradesback = (age - 7) - highestgrade$
- A female indicator variable
- A non-white indicator variable

c. Rescale GPA: Redefine GPA so that it is the original value divided by 100.

d. Filter the data set to only include the years 1997-2007, for individuals from the South-East (region 4), with a highest grade completed (*highestgrade*) between 8 and 17.

7. **Create and view summary statistics**

a. By age, create and display summary statistics with average values of:[2]

- Highest grade completed (*highestgrade*)
- Number of grades below expected level (*gradesback*)

b. By highest grade completed, create and display summary statistics with average values of:

- Parental income (*parentincome*)
- Mother's education (*motheredyrs*)
- GPA (*gpa*)
- Share women
- Share nonwhite

Bonus! If you aren't ready to stop having fun, the steps needed to really make this analysis complete would be to consult the codebook (*.cdb*) file to recode the factor variables into readable values and to appropriately recode missing values using **na_if()**.

[2]For both summary statistics tables, you may need to use the na.rm = TRUE to remove missing values when calculating the averages.