Harmonized Cultural Access & Participation Dataset

```
library(retroharmonize)
cap_files <- c('ZA4529_v3-0-1.sav', 'ZA5688_v6-0-0.sav', 'ZA6925_v1-0-0.sav')
```

- European Commission (2012). Eurobarometer 67.1 (Feb-Mar 2007). GESIS Data Archive, Cologne. ZA4529 Data file Version 3.0.1, https://doi.org/10.4232/1.10983.
- European Commission, Brussels (2016). Eurobarometer 79.2 (2013). GESIS Data Archive, Cologne. ZA5688 Data file Version 6.0.0, https://doi.org/10.4232/1.12577.
- European Commission and European Parliament, Brussels (2018). Eurobarometer 88.1 (2017). GESIS Data Archive, Cologne. ZA6925 Data file Version 1.0.0, https://doi.org/10.4232/1.12959.

To replicate this example, you must acquire these files from GESIS after accepting their terms of use. The `retroharmonize` project is not affiliated with GESIS.

```
## This is a dummy path. You should use the path where you saved these files.
gesis_dir <- file.path("C:", "your_data", "gesis_files")
```

```
## Create full paths to the three selected files
cap_files <- file.path(gesis_dir, cap_files)
```

Inventory

```
document_cap_files <- document_surveys(survey_path = cap_files, .f = "read_spss")
```

```
knitr::kable(document_cap_files)
```

The files are rather large. We have `sum(document_cap_files$nrow)` of `sum(document_cap_files$ncol)` variables. We will only harmonize a few of them.

Importing

It is likely that the surveys will fit into your computer's memory, so you can omit the first step. If you work with many files, and you want to keep working sequentially with survey files, it is a good idea to convert them to R objects.

Concepts to harmonize

```
cap_metadata <- metadata_create(survey_files = cap_files)
```

Harmonization of Variables

Weights

```
weight_variables <- eb_metadata %>%
  filter ( .data$var_name_orig %in% c("isocntry", "wex", "wextra", "v47", "v7"))
```

```
weigthing_crosswalk_table <- crosswalk_table_create(weight_variables) %>%
  select (
```

```
    -all_of(c("val_numeric_orig", "val_numeric_target", "val_label_orig", "val_label_target"))
    ) %>%
  mutate ( var_name_target = case_when (
    .data$var_name_target %in% c("wex", "wextra", "v47") ~ 'wex',
    .data$var_name_orig == "rowid"  ~ 'rowid',  # do not forget to keep the unique row IDs
    TRUE ~ "geo"),
    class_target = ifelse(.data$var_name_target %in% c("geo", "v47"), "factor", "numeric")
    )
```

The crosswalk table contains the original (source) variable names that must be converted to the target variable names, i.e. `v7` and `isocntry` to `geo` and `v47`, `wextra`, and `wex` to `wex`.

```
weigthing_crosswalk_table %>%
  select ( all_of(c("filename", "var_name_orig", "var_name_target")))
```

The Eurobarometer surveys contains separate samples for the former West Germany (`DE-W`), East Germany (`DE-E`), Northern Ireland (`GB-NIR`), and Great Britain, i.e. England, Scotland and Wales (`GB-GBN`). For the variable `geo` the appropriate post-stratification weight is the `w1` variable. For the two German subsamples, it is `w_de`, and for the two United Kingdom subsamples it is the `w_uk`. We create two new variables, the `country_code` and `w`. When the two UK and the two German samples are joined, the appropriate post-stratification weight is `w`.

```
weight_vars <- weight_vars %>%
  mutate ( country_code = substr(geo, 1,2),
           w = case_when (
             country_code == "DE" ~  w_de,
             country_code == "GB" ~  w_uk,
             TRUE ~ w1 )) %>%
  mutate ( year_survey = case_when(
    .data$id == "ZA4529_v3-0-1" ~ '2007',
    .data$id == "ZA5688_v6-0-0" ~ '2013',
    .data$id == "ZA6925_v1-0-0" ~ '2017'
  )) %>%
  mutate ( year_survey = as.factor(.data$year_survey))
```

Demography

We will use two important demography variables: the age of the respondent, and the age of school leaving of the respondent. This latter, `age_education` variable is an important, *ex ante* harmonized variable in Eurobarometer. Because each European country has different education systems, furthermore, because education systems are not the same across different demographic groups (for example, people schooled before the World War II often went to very different schools), a more precise education level would require a very complicated mapping with education-specific knowledge.

The age_education variable has three special values: 1. the value for declined answers, which should be coded in a numeric representation to the special `NA` value or R;

2. An integer outside the valid range of responses, which means that the person is still tudying. We will recode the still studying answers to 0, and later we will further recode them to the current age of the person. We will also mark this special group as students—in their case, the `age_education` has a different meaning. A 17 year-old respondent in Europe is likely to study further; a middle aged person who replied with 17 has a low education level according to the current norms. In this case, a student's response of 17 is not the same as the middle aged person's;

3. And at last, the special value `No formal education` means that the person did not finish the primary school. Different countries define differently the minimum mandatory age when a person can leave the

school system. We code these answers to 14, which is lower than the lowest age in our sample (15). This is also a special group, so we will mark them with a dummy variable, too.

```
demography_crosstable <- eb_metadata %>%
  filter ( .data$var_name_orig == "rowid" |
             .data$var_label_orig %in% c("age_exact", "age_education")) %>%
  crosswalk_table_create() %>%
  mutate ( var_name_target = case_when(
    # Do not leave out the unique row identifiers!
    .data$var_name_orig == "rowid" ~ "rowid",
    TRUE ~ .data$var_label_orig
  )) %>%
  mutate ( na_label_target = case_when(
    .data$na_label_orig %in% c("DK", "Refusal") ~ "Declined",
    TRUE ~ .data$na_label_target
  )) %>%
  mutate ( val_numeric_target = case_when(
    .data$val_label_orig == "No full-time education" ~ 14,
    .data$val_label_orig == "Still studying"         ~ 0,
    TRUE ~ .data$val_numeric_target
  )) %>%
  mutate ( na_numeric_target = case_when (
    .data$na_label_target == "Declined" ~ 99999,
    TRUE ~ NA_real_
  )) %>%
  mutate ( class_target = case_when (
    .data$var_name_target == "rowid" ~ 'character',
    TRUE ~ "numeric"
  ))
```

Cultural Access & Participation Variables

Our variables of interest are visiting frequencies to concerts and public libraries. In the Eurobarometer CAP surveys, the answers have

1. `Never in the last twelve months` or `None` will be coded as 0 visits, and their factor label will be harmonized to `never`. In this case, the questionnaire reveals that whilst this was an *ex ante* harmonized question, the answers were not consistently labelled in the three files. We can safely bring these responses to the same value
2. `1-2 times` will be coded to the programatically easier to use `1_2_times`, and it will get a numeric value of 1.5. Later, we will try to establish a better numeric value, now it is important that all responses labelled as `1-2 times` should have the same numeric code.
3. `3-5 times` will be coded to the numeric value of 3.5 and 3_5_times.
4. `More than 5 times` will be `more_than_5` and have the numeric value of 6.
5. The special case of `DK` will be coded as `declined` (to anwer) and the *Inap...* lables to `inap`. These are special codes in Eurobarometer meaning that the item is inappropriate, i.e. due to some filtering, this particular question was not asked from the respondant. While respondents who consciously decline an answer usually form a rather homogenous category of their own, the ianppropriate answers are truly missing answers. Their numeric representation will be both `NA` (f.e. they should be omitted from a numeric average.)

```
search_term_labels <- c("rowid|cultural_activities_public_library|cultural_activities_museums_galleries
cultural_activities_freq_public_library|cultural_activities_freq_library_or_archive|cultural_activities_

cap_vars_crosstable <- eb_metadata %>%
```

```
  filter ( .data$var_name_orig == "rowid" |
             grepl(search_term_labels, .data$var_label_orig) ) %>%
  crosswalk_table_create () %>%
  mutate ( var_name_target = case_when(
    .data$var_name_orig == "rowid" ~ "rowid",
    grepl("concert", .data$var_label_orig) ~ "visit_concert",
    grepl("library", .data$var_label_orig) ~ "visit_library",
    grepl("museum", .data$var_label_orig)  ~ "visit_museum",
    TRUE ~ .data$var_label_orig
  )) %>%
  mutate ( na_label_target = case_when(
    .data$na_label_orig %in% c("DK", "Refusal") ~ "declined",
    grepl("^Inap", .data$na_label_orig)         ~ "inap",
    .data$val_label_orig == "DK"                ~ "declined",
    TRUE ~ .data$na_label_target
  )) %>%
  mutate ( val_label_target = case_when (
    grepl("^more", tolower(.data$val_label_orig)) ~ "more_than_5",
    grepl("never|^not|none", tolower(.data$val_label_orig))  ~ "never",
    tolower(.data$val_label_orig) == "1-2 times" ~ "1_2_times",
    tolower(.data$val_label_orig) == "3-5 times" ~ "3_5_times",
    !is.na(.data$na_label_target) ~  .data$na_label_target,
    # do not forget the rowid, it should not be labelled
    .data$var_name_target == "rowid" ~ NA_character_,
    TRUE ~ "coding_error"
  )) %>%
  mutate ( na_numeric_target = case_when (
    .data$na_label_target == "declined" ~ 99999,
    .data$na_label_target == "inap"     ~ 99998,
    TRUE ~ NA_real_
  )) %>%
  mutate ( val_numeric_target = case_when (
    .data$val_label_target == "never"        ~ 0,
    .data$val_label_target == "1_2_times"    ~ 1.5,
    .data$val_label_target == "3_5_times"    ~ 3.5,
    .data$val_label_target == "more_than_5" ~ 6,
    TRUE ~ .data$na_numeric_target
  ))
```

We create two version of the crosswalk table. One will rename the vist_ variables to fct_visit, and give them a factor representation.

```
cap_vars_crosstable_fct <- cap_vars_crosstable %>%
  mutate ( var_name_target = ifelse(
            test = grepl("^visit", .data$var_name_target),
            yes  = gsub("visit", "fct_visit", .data$var_name_target),
            no   = .data$var_name_target),
            class_target = ifelse(
              test  = .data$var_name_target == 'rowid',
              yes = "character", # the rowid remains a character
              no  = "factor" )
            )

cap_vars_fct <- crosswalk(
```

```
  crosswalk_table = cap_vars_crosstable_fct,
  survey_list = cap)
```

The other will give these variable a numeric representation.

```
cap_vars_crosstable_num <- cap_vars_crosstable %>%
   mutate( class_target = ifelse(
      test  = .data$var_name_target == 'rowid',
      yes = "character",  # the rowid remains a character
      no = "numeric" )
  )

cap_vars_num <- crosswalk(
  crosswalk_table = cap_vars_crosstable_num,
  survey_list = cap)
```

Join the harmonized CAP dataset

```
harmonized_cap_file <- cap_vars_fct %>%
  left_join ( cap_vars_num,    by = c("id", "rowid") ) %>%
  left_join ( demography_vars, by = c("id", "rowid") ) %>%
  left_join ( weight_vars,     by = c("id", "rowid") )
```

Unit testing and corrections

Weights

The average post-stratification weight must be 1 for `w1`, and for `w_de` and `w_uk` whenever they are not zero values. The `wex` variable has no naturally defined mean value.

```
weight_vars %>%
  group_by ( .data$id ) %>%
  mutate ( across(starts_with("w"), function(x) ifelse(x==0, NA, x))) %>%
  dplyr::summarise( across(starts_with('w'), mean, na.rm=TRUE))
```

Missing labels

All concert visits labelled as `declined` and `inap` must be coded to a numeric `NA` value. Furthermore, there must not be other `NA` values than cases in the `ZA6925_v1-0-0` survey, where this question was not asked. For consistency, these observations should be also coded as `inap` in the factor representation.

In case of the `visit_library` variable all missing values should have a factor representation of `declined` or `inap`.

```
harmonized_cap_file %>%
  select ( starts_with(c("id", "fct", "visit")) ) %>%
  select ( -contains("museum"), -contains("library") ) %>%
  filter ( is.na(.data$visit_concert) ) %>%
  distinct_all()

harmonized_cap_file %>%
  select ( starts_with(c("id", "fct", "visit")) ) %>%
  select ( -contains("museum"), -contains("concert") ) %>%
  filter ( is.na(.data$visit_library) ) %>%
  distinct_all()
```

```
harmonized_cap_file <- harmonized_cap_file %>%
  mutate ( fct_visit_museum = ifelse(.data$id == "harmonized_cap_file", "inap", .data$fct_visit_museum)
  mutate ( is_visit_concert = ifelse(.data$visit_concert == 0, 0, 1),
           is_visit_library = ifelse(.data$visit_library == 0, 0, 1),
           is_visit_museum  = ifelse(.data$visit_museum  == 0, 0, 1))
```

```
harmonized_cap_file %>%
  group_by ( .data$id) %>%
  dplyr::summarise ( across(starts_with("visit"),     mean, na.rm=TRUE),
                     across(starts_with("is_visit"),  mean, na.rm=TRUE),
                     .groups = "keep")
```

Exporting

```
saveRDS(harmonized_cap_file,  file.path(tempdir(), "harmonized_cap_dataset.rds"))
write.csv(harmonized_cap_file,
          file.path(tempdir(), "harmonized_cap_file_20211214_doi_10_5281_zenodo_5781672.csv"),
          row.names = FALSE)
```

You can find this harmonized dataset on Zenodo in the Digital Music Observatory and the Cultural Creative Sectors Industries Data Observatory repositories under the DOI `10.5281/zenodo.5781672`.