Package: Wavelength (via r-universe)

September 12, 2024

Title Wavelength

Version 2.4.1

Description USAID OHA Office. Munging of mission weekly HFR data.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/USAID-OHA-SI/Wavelength

BugReports https://github.com/USAID-OHA-SI/Wavelength/issues

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Depends R (>= 3.1.2)

Imports crayon, dplyr(>= 1.0.0), lubridate, magrittr, purrr, readr, readxl, stats, stringr, tibble, tidyr, vroom

Suggests curl, glue, googledrive, googlesheets4, httr, jsonlite, openxlsx, usethis, plyr, janitor

Remotes USAID-OHA-SI/glamr, USAID-OHA-SI/grabr, USAID-OHA-SI/gagglr

Repository https://usaid-oha-si.r-universe.dev

RemoteUrl https://github.com/USAID-OHA-SI/Wavelength

RemoteRef HEAD

RemoteSha 270e88233a0316f521d64adefe70112954c9ab33

Contents

apply_filetimestamp *Apply Time Stamp to file and*

Description

Apply Time Stamp to file and

Usage

```
apply_filetimestamp(df, gdrive_rename = TRUE)
```

Arguments

df	dataframe from identify_newfiles()
gdrive_rename	rename on Google drive? defaule = TRUE

check_content *Validate output content*

Description

Additional/optional validation against DATIM tables.

Usage

```
check_content(df, output_path, datim_path)
```

Arguments

df	HFR data framed created by hfr_process_template()
datim_path	path to look up files from pull_hierarchy,pull_mech, pull_mer

Value

df updated HFR dataframe

See Also

Other validation: check_dates(), check_disaggs(), check_orgunituids(), check_output_cols(), is_hfrtab(), validate_import(), validate_initial(), validate_output()

check_dates *Validate dates*

Description

Check whether there are any rows/records with missing dates. Provides readout on this as well as on whether the submission covers multiple periods and the dates covered in the file.

Usage

```
check_dates(df)
```

Arguments

df	HFR data framed created by hfr_process_template()

See Also

Other validation: check_content(), check_disaggs(), check_orgunituids(), check_output_cols(), is_hfrtab(), validate_import(), validate_initial(), validate_output()

check_disaggs *Validate disaggs for export*

Description

Check whether there are any rows/records with missing disaggs and provides readout

Usage

```
check_disaggs(df)
```

Arguments

df	HFR data framed created by hfr_process_template()

See Also

Other validation: check_content(), check_dates(), check_orgunituids(), check_output_cols(), is_hfrtab(), validate_import(), validate_initial(), validate_output()

check_inds *Validate indicators for export*

Description

Check whether there are any rows/records with missing indicators and provides readout

Usage

```
check_inds(df)
```

Arguments

df	HFR data framed created by hfr_process_template()

check_mechs *Validate mechanisms for export*

Description

Check whether there are any rows/records with missing mechanisms and provides readout of the mechanisms included

Usage

```
check_mechs(df)
```

Arguments

df	HFR data framed created by hfr_process_template()

check_orgunituids *Validate orgunituids for export*

Description

Check whether there are any rows/records with missing orgunits and provides readout

Usage

```
check_orgunituids(df)
```

Arguments

df	HFR data framed created by hfr_process_template()

See Also

Other validation: check_content(), check_dates(), check_disaggs(), check_output_cols(), is_hfrtab(), validate_import(), validate_initial(), validate_output()

`check_output_cols` *Validate columns for export*

Description

Ensure all expected columns exist before exporting

Usage

```
check_output_cols(df)
```

Arguments

df	HFR data framed created by hfr_process_template()

See Also

Other validation: check_content(), check_dates(), check_disaggs(), check_orgunituids(), is_hfrtab(), validate_import(), validate_initial(), validate_output()

`confirm_validations` *Confirm validations of processed files*

Description

Confirm validations of processed files

Usage

```
confirm_validations(hfr_data, hfr_errors, dir_files)
```

Arguments

hfr_data	content of processed files
hfr_errors	errors detected from files content
dir_files	location of processed files

Examples

```
## Not run:
confirm_validations(hfr_data, hfr_errors, dir_files)

## End(Not run)
```

curr_fy *Current Fiscal Year*

Description

Current Fiscal Year

Usage

```
curr_fy
```

Format

An object of class `numeric` of length 1.

Value

Current Fiscal Year

ddcpv_check *DDC Pre-Validation Check*

Description

This code should be run on HFR submission prior to loading into DDC/s3. Currently, DDC cannot handle two issues - (1) tabs with only one row of data and (2) tabs not ordered from least to greatest This code resolves the first by creating a second row of data with the first value and the second by reordering tabs using openxlsx.

Usage

```
ddcpv_check(filepath)
```

Arguments

`filepath`	path to HFR submission

Value

print out of checks and

Examples

```
## Not run:
files <- list.files("ou_submissions/", "xlsx", full.names = TRUE)
ddcpv_check(files[1])
purrr::walk(files, ddcpv_check)
## End(Not run)
```

download_new *Download New Submission to upload*

Description

Download New Submission to upload

Usage

```
download_new(df)
```

Arguments

df	dataframe from apply_filetimestamp

extract_iso3code *Extract OU ISO3 code*

Description

Extract OU ISO3 code

Usage

```
extract_iso3code(pfile)
```

Arguments

pfile	processed file

Value

ISO3 3 character iso code

Examples

```
## Not run:
  extract_iso3code('HFR_2020.99_XAR_100000_processed_20200528.csv')

## End(Not run)
```

extract_mechcode *Extract mechanism code*

Description

Extract mechanism code

Usage

```
extract_mechcode(pfile)
```

Arguments

pfile	processed file

Value

mech_code mechanism code

Examples

```
## Not run:
  extract_mechcode('HFR_2020.99_XAR_100000_processed_20200528.csv')

## End(Not run)
```

gen_url *Generate a API URL*

Description

Generate a API URL

Usage

```
gen_url(
  ou_uid,
  org_lvl,
  org_type = "facility",
  value_type = "results",
  is_hts = FALSE,
  fy_pd = NULL,
  baseurl = "https://final.datim.org/"
)
```

Arguments

ou_uid	UID for the country, recommend using identify_ouuids()
org_lvl	org hierarchy level, eg facility is level 7 in country X, recommend using identify_levels()
org_type	organization type, either facility (default) or community
value_type	results (default) or targets
is_hts	is the API for HTS indicators (HTS_TST or HTS_TST_POS), default = FALSE
fy_pd	fiscal year(s) to cover, default will be current FY if not provided
baseurl	API base url, default = https://final.datim.org/

Examples

```
## Not run:
 #get OU UID
  ouuid <- identify_ouuids() %>% dplyr::filter(ou == "Ghana")
 #get facility level
  faclvl <- identify_levels("Ghana", "facility", username = myuser, password = mypwd())
 #gen url
  myurl <- gen_url(ouuid, faclvl, org_type = facility)
## End(Not run)
```

---

get_datim_data *DATIM API Call for Targets*

Description

DATIM API Call for Targets

Usage

```
get_datim_data(url, username, password)
```

Arguments

url	supply url forAPI call, recommend usinggen_url()
username	DATIM username
password	DATIM password, recommend using mypwd()

Examples

```
## Not run:
 myurl <- paste0(baseurl, "api/29/analytics.json?
                 dimension=LxhLO68FcXm:udCop657yzi&
                 dimension=ou:LEVEL-4;HfVjCurKxh2&
                 filter=pe:2018Oct&
                 displayProperty=SHORTNAME&outputIdScheme=CODE")
 myuser <- "UserX"
 df_datim <- get_datim_data(myurl, myuser, mypwd(myuser))
## End(Not run)
```

get_datim_targets *DATIM API Call for Targets*

Description

DATIM API Call for Targets

Usage

```
get_datim_targets(url, username, password)
```

Arguments

url	supply url forAPI call, recommend usinggen_url()
username	DATIM username
password	DATIM password, recommend using mypwd()

Examples

```
## Not run:
 myurl <- paste0(baseurl, "api/29/analytics.json?
                 dimension=LxhLO68FcXm:udCop657yzi&
                 dimension=ou:LEVEL-4;HfVjCurKxh2&
                 filter=pe:2018Oct&
                 displayProperty=SHORTNAME&outputIdScheme=CODE")
 myuser <- "UserX"
 df_targets <- get_datim_targets(myurl, myuser, mypwd(myuser))
## End(Not run)
```

get_operatingunit *Get operating unit name*

Description

Get operating unit name

Usage

```
get_operatingunit(iso_code, orglevels)
```

Arguments

iso_code	iso3 code
orglevels	df org levels

Value

operating unit

Examples

```
## Not run:
  get_operatingunit(org_levels, 'XWA')

## End(Not run)
```

`guess_operatingunit` *Get operating unit name*

Description

Get operating unit name

Usage

```
guess_operatingunit(pfile, levels, ims)
```

Arguments

`pfile`	processed file
`levels`	org levels
`ims`	mechanisms df

Value

operating unit name

Examples

```
## Not run:
  get_mech_ou(ims, 'HFR_2020.99_XAR_100000_processed_20200528.csv')

## End(Not run)
```

hfr_aggr *Aggregate HFR dataframe*

Description

Aggregate data frame to combine rows where needed (minimize row count). Multiple lines may be entered for the same unique reporting combination/id

Usage

```
hfr_aggr(df)
```

Arguments

df	HFR data frame imported via `hfr_import()`

hfr_append_sources *#' Append HFR and DATIM Data*

Description

#' Append HFR and DATIM Data

Usage

```
hfr_append_sources(
  folderpath_hfr,
  folderpath_datim,
  start_date,
  weeks = 4,
  max_date = TRUE,
  folderpath_output
)
```

Arguments

folderpath_hfr	folder path to HFR processed data
folderpath_datim	folder path to DATIM extracts, see `extract_datim()`
start_date	start date of HFR period, YYYY-MM-DD format
weeks	number of weeks to create, default = 4
max_date	cut off data at max date? default = NULL
folderpath_output	folder path for saving the output

hfr_assign_pds *Add HFR Period column*

Description

Creates new columns for specifying the HFR reporting period - fy (20XX) and hfr_pd (fiscal month), 1-12

Usage

```
hfr_assign_pds(df)
```

Arguments

df	HFR data frame with date

Examples

```
## Not run:
 df <- hfr_assign_pds(df)
## End(Not run)
```

hfr_export *Export High Frequency Data*

Description

Function to export different data frames with standardized naming formats

Usage

```
hfr_export(
  df,
  folderpath_output = NULL,
  type = "processed",
  by_mech = FALSE,
  quarters_complete = NULL
)
```

Arguments

df	structured High Frequency Data Frame
folderpath_output	provide the full path to the folder for saving
type	type of data being saved, default = processed
by_mech	export by mechanism, default = FALSE
quarters_complete	FOR DATIM ONLY: # of quarters completed through FY to determine weeks left in year

Examples

```
## Not run:
 #write output
   hfr_export(df_tza, "~/WeeklyData")
## End(Not run)
```

---

hfr_extract_meta *Extract Meta Data Information about Template*

Description

Useful for pulling information about the template, whether It be the Operating Unit (OU), Period, template version, or type, eg wide or long.

Usage

```
hfr_extract_meta(filepath, meta_type = "type")
```

Arguments

filepath	filepath to sumbitted template
meta_type	type of meta data requesting: ou, period, version, type (default)

Examples

```
## Not run:
#identify whether template is long or wide
  filepath <- "~/WeeklyData/Raw/KEN_Weekly.xlsx"
  hfr_extract_meta(filepath, meta_type = "type")
#identify period
  hfr_extract_meta(filepath, meta_type = "period")
#identify OU
  hfr_extract_meta(filepath, meta_type = "ou")
## End(Not run)
```

hfr_filter_pd *Filter to Select HFR Period*

Description

Limits data frame to only the current reporting period when the fiscal year and period are provided.

Usage

```
hfr_filter_pd(df, hfr_pd_sel = NULL, hfr_fy_sel = NULL)
```

Arguments

df	HFR data frame imported via hfr_import()
hfr_pd_sel	HFR reporting period, 1-13, no filter when NULL, default = NULL
hfr_fy_sel	fiscal year, default = NULL

hfr_fix_date *Convert dates to date format*

Description

The submission templates do not have date validations stored in the Excel files, so there can be a number of date types submitted that we attempt to account for outside of the normal, ISO format of YYYY-MM-DD. This function handles each identfied date type as a separate dataframe and then binds them back together. Date formats include - Excel, ISO, character dates. If the round_hfrdate param is TRUE, the function rounds to the start of the week or month and aggregates any mid-week/month submission.

Usage

```
hfr_fix_date(df, round_hfrdate = FALSE)
```

Arguments

df	HFR data frame imported via hfr_import()
round_hfrdate	rounds date to the nearest HFRweek start (for non-compliance), default = FALSE

hfr_fix_noncompliance *Resolve issues with non-standard entries*

Description

A number of issues crop up frequently enough around indicators, disaggregates, values, etc, that worth coding a resolution into a singular place. This function aims to be a place to rectify any non-standard issue that arises more than a couple of times.

Usage

```
hfr_fix_noncompliance(df)
```

Arguments

df	HFR data frame imported via hfr_import()

hfr_gap_target *Generate Gap Targets*

Description

Generate Gap Targets

Usage

```
hfr_gap_target(df, quarters_complete)
```

Arguments

df	data frame created by extract_datim()
quarters_complete	MER quarters with data available

Examples

```
## Not run:
 myuser <- "UserX"
 mech_x_targets <- extract_datim(00001, myuser, mypwd(myuser))
 mech_x_targets <- hfr_gap_target(mech_x_targets, 2)
## End(Not run)
```

hfr_gather *Reshape HFR Data frame long*

Description

Tidy Wide or Wide-LIMITED submissions by pivoting them long and separating column name into relevant parts.

Usage

```
hfr_gather(df)
```

Arguments

df	HFR data frame imported via `hfr_import()`

hfr_group_wkly *Create Monthly Aggregate for Weekly data*

Description

Changes frequency from week to month agg and rounds up to month

Usage

```
hfr_group_wkly(df)
```

Arguments

df	HFR data frame imported via `hfr_import()`

hfr_gs_statrep *DDC Status Report Google Sheet ID*

Description

DDC Status Report Google Sheet ID

Usage

```
hfr_gs_statrep
```

Format

An object of class `character` of length 1.

Value

google sheet id

`hfr_identify_freq` *Identify reporting frequency*

Description

Flags the frequency of reporting, adding this to the data frame for tracking and for use if rounding and aggregating.

Usage

```
hfr_identify_freq(df)
```

Arguments

`df`	HFR data frame from `hfr_fix_date()`

`hfr_identify_pds` *Create a data frame of HFR weeks and periods*

Description

Create a data frame of HFR weeks and periods

Usage

```
hfr_identify_pds(fy = NULL, week = FALSE)
```

Arguments

`fy`	fiscal year
`week`	are periods weekly? default = FALSE

hfr_import *Import template sheet(s)*

Description

Read in a standard HFR submission, binding all tabs into one. Note that all tabs must be in the same template format - all long, wide, or wide-limited.

Usage

```
hfr_import(filepath)
```

Arguments

filepath	filepath to submitted template

Examples

```
## Not run:
#identify whether template is long or wide
  filepath <- "~/HFR/HFR_FY23_Oct_Moldova_20221015.xlsx"
  df_hfr <- hfr_import(filepath)
## End(Not run)
```

hfr_munge_string *Standardized Text*

Description

Clean up/standardize string text for indicators and disaggregates

Usage

```
hfr_munge_string(df)
```

Arguments

df	HFR data frame imported via `hfr_import()`

hfr_orgunit_search *Search Org Hierarchy for Org Unit*

Description

A look up for an partial orgunit name against the DATIM list of orgunits, when trying to find the correct or missing orgunituid

Usage

```
hfr_orgunit_search(df, orgunit_name, ou = NULL)
```

Arguments

df	org hierarchy, created in pull_hierarchy()
orgunit_name	full or partial orgunit name for matching
ou	operating unit; if added searches only that OU default = NULL

Examples

```
## Not run:
load_secrets("datim")
org <- pull_hierarchy(datim_user(), datim_pwd())
# orgunit - "Kewot"
hfr_orgunit_search(org, "Kew", "Ethiopia")
## End(Not run)
```

hfr_process_template *Import and Munge HFR Standard Templates*

Description

Run validation check on country HFR submission while it processes and tidies the submission for inclusion into the database.

Usage

```
hfr_process_template(
  filepath,
  round_hfrdate = FALSE,
  hfr_pd_sel = NULL,
  folderpath_output = NULL,
  datim_path = NULL
)
```

Arguments

filepath	filepath to sumbited template
round_hfrdate	rounds date to the nearest HFRweek start (for non-compliance), default = FALSE
hfr_pd_sel	filter for HFR reporting period, 1-13, no filter when NULL, default = NULL
folderpath_output	if a csv output is desired, provide the full path to the folder
datim_path	path to DATIM lookup files for full validation

Examples

```
## Not run:
#file path for the
  path <- "~/WeeklyData/Saturn"
  output_folder <- "~/WeeklyData/Output"
#process Excel file for Saturn
  hfr_process_template(path, output_folder)
#process Excel file for Saturn with full validation
  datim_folder <- "~/Datim"
  hfr_process_template(path, output_folder, datim_path = datim_folder)

## End(Not run)
```

---

hfr_read *Read in HFR output file*

Description

Read in HFR output file

Usage

```
hfr_read(filepath)
```

Arguments

filepath	filepath of an HFR output file

Examples

```
## Not run:
 path <- "~/data/HFR_2020.01_Global_output_20191204.1705.csv"
 df <- hfr_read(path)

## End(Not run)
```

hfr_read_all *Batch read hfr files*

Description

Batch read hfr files

Usage

```
hfr_read_all(pfolder, pattern, source = FALSE)
```

Arguments

pfolder	processed file folder
pattern	filename pattern
source	append source filename?

Value

df

Examples

```
## Not run:
  hfr_read_all('./Data/2020.05', pattern='HFR_FY2020.05')

## End(Not run)
```

hfr_rectify_date *Rectify Incorrectly Submitted Dates*

Description

Without access to change data directly in the database, the only means of changing data is to resubmit a zeroed out version of the original submission. This function replaces any cell with entered data with zero so that it can be resubmitted and processed by Trifacta. The original, correct version should be re-processed after the zeroed dataset has been processes

Usage

```
hfr_rectify_date(subm_file, subm_tab, folderpath_templates = "templates/")
```

Arguments

subm_file	submission file with incorrect dates
subm_tab	tab with incorrect dates
folderpath_templates	folder path where current HFR templates are located

Value

exports two files - one with zeroed out data for wrong date and one with corrected date

Examples

```
## Not run:
#store file paths for looping over to read in
  df_files_tabs <- list.files(folderpath, full.names = TRUE) %>%
    purrr::map_dfr(~ tibble::tibble(file = .x,
                                    tabs = readxl::excel_sheets(.x))) %>%
    dplyr::filter(stringr::str_detect(tabs, "HFR"))
#fix date and create zeroed version for wrong date (to clean from DB)
 pwalk(df_files_tabs, ~hfr_rectify_date(..1, ..2))
## End(Not run)
```

---

hfr_restrict_cols *Restrict HFR data frame columns*

Description

Limits the datafram to only have the exact columsn found in the template, removing any extras

Usage

```
hfr_restrict_cols(df)
```

Arguments

df	HFR data frame imported via `hfr_import()`

hfr_round_date *Round Date Values*

Description

Round values to nearest HFR date

Usage

```
hfr_round_date(df)
```

Arguments

df	df HFR data frame imported via `hfr_import()`

hierarchy_clean *Clean up DATIM Hierarchy Path*

Description

Clean up DATIM Hierarchy Path

Usage

```
hierarchy_clean(df)
```

Arguments

df	data frame created by `hierarchy_extract()`

hierarchy_extract *Pull Hierarchy Data from DATIM*

Description

Pull Hierarchy Data from DATIM

Usage

```
hierarchy_extract(
  ou_uid,
  username,
  password,
  baseurl = "https://final.datim.org/"
)
```

Arguments

ou_uid	UID for the country, recommend using identify_ouuids()
username	DATIM username
password	DATIM password, recommend using mypwd()
baseurl	API base url, default = https://final.datim.org/

Examples

```
## Not run:
#get OU UID
  ouuid <- identify_ouuids() %>% dplyr::filter(ou == "Kenya")
#pull hierarchy (paths are all UIDs)
  df <- hierarchy_extract(ouuid, username = myuser, password = mypwd(myuser))
## End(Not run)
```

---

hierarchy_identify_ctry *Extract country name from OU or country name*

Description

Extract country name from OU or country name

Usage

```
hierarchy_identify_ctry(df)
```

Arguments

df	data frame created by hierarchy_extract() %>% hierarchy_clean()

---

hierarchy_rename *Rename Hierarchy from Levels to OU/SNU1/PSNU/Facility*

Description

Rename Hierarchy from Levels to OU/SNU1/PSNU/Facility

Usage

```
hierarchy_rename(
  df,
  country,
  username,
  password,
  baseurl = "https://final.datim.org/"
)
```

Arguments

df	data frame created by `hierarchy_extract() %>% hierarchy_clean()`
country	county name, eg "Malawi" or "Nepal"
username	DATIM username
password	DATIM password, recommend using `mypwd()`
baseurl	API base url, default = https://final.datim.org/

---

identify_levels *Identify Facility/Community levels in org hierarchy*

Description

Identify Facility/Community levels in org hierarchy

Usage

```
identify_levels(
  ou = NULL,
  username,
  password,
  baseurl = "https://final.datim.org/"
)
```

Arguments

ou	operating unit name
username	DATIM username
password	DATIM password, recommend using `mypwd()`
baseurl	base API url, default = https://final.datim.org/

Examples

```
## Not run:
 #table for all OUs
  myuser <- "UserX"
  identify_levels(username = myuser, password = mypwd())
 #table for just Kenya
   identify_levels("Kenya", username = myuser, password = mypwd())
## End(Not run)
```

identify_newfiles *Identify New Submissions on Google Drive*

Description

Identify New Submissions on Google Drive

Usage

```
identify_newfiles(print_files = TRUE, id_modified = TRUE)
```

Arguments

print_files	print out list of new files
id_modified	removed Modified from filename

Value

new files, along with submission (df_submissions) and s3 files from the archive folder (df_archive)

identify_ouuids *Pull OU UIDS*

Description

Pull OU UIDS

Usage

```
identify_ouuids(username, password, baseurl = "https://final.datim.org/")
```

Arguments

username	DATIM Username
password	DATIM password, recommend using mypwd()
baseurl	base url for the API, default = https://final.datim.org/

Examples

```
## Not run:
 ous <- identify_ouuids("userx", mypwd("userx"))
## End(Not run)
```

iso_map *ISO Code Mapping Table*

Description

Table that cross references the ISO Country codes to the PEPFAR Operating Units and Countries. This table is used during the export process for file naming. ISO codes were originall pulled from DATIM.

Usage

```
data(iso_map)
```

Format

A data frame with each PEPFAR OU and associated ISO code.

operatingunit PEPFAR Operating Unit or country

iso 3 letter ISO code for the Operatingunit or country

regional TRUE if operatingunit is a country under a regional program

Source

"https://final.datim.org/api/dataStore/dataSetAssignments/orgUnitLevels"

is_date_valid *Validate reporting dates*

Description

Validate reporting dates

Usage

```
is_date_valid(.data, df_dates)
```

Arguments

.data	df
df_dates	df_dates

Value

boolean

Examples

```
## Not run:
  data %>% is_date_valid(df_dates)

## End(Not run)
```

`is_hfrtab` *Determine if there are tabs to import*

Description

Determine if there are tabs to import

Usage

```
is_hfrtab(filepath)
```

Arguments

`filepath`	filepath to submitted template

See Also

Other validation: `check_content()`, `check_dates()`, `check_disaggs()`, `check_orgunituids()`, `check_output_cols()`, `validate_import()`, `validate_initial()`, `validate_output()`

`is_mech4ou` *Validate mechanism code*

Description

Validate mechanism code

Usage

```
is_mech4ou(.data, df_mechs)
```

Arguments

`.data`	df
`df_mechs`	mechs

Value

boolean

Examples

```
## Not run:
  data %>% is_mech4ou_valid(df_mechs)

## End(Not run)
```

`is_mech_valid` *Validate mechanism code*

Description

Validate mechanism code

Usage

```
is_mech_valid(.data, df_mechs)
```

Arguments

`.data`	df
`df_mechs`	mechs

Value

boolean

Examples

```
## Not run:
  data %>% is_mech_valid(df_dates)

## End(Not run)
```

`is_metatab` *Determine whether meta tab exists*

Description

Determine whether meta tab exists

Usage

```
is_metatab(filepath)
```

Arguments

`filepath`	filepath to sumbitted template

See Also

Other utility: package_check(), var_exists()

`is_orgunituid4ou` *Check if orgunituid exist in operating unit*

Description

Check if orgunituid exist in operating unit

Usage

```
is_orgunituid4ou(.data, df_orgs)
```

Arguments

df_orgs	org_hierarchy df
data	df

Value

data with new column: T/F

Examples

```
## Not run:
  data %>% orgunituid4ou(df_orgs)

## End(Not run)
```

`is_orgunituid_valid` *Check if Operating Unit name is valid*

Description

Check if Operating Unit name is valid

Usage

```
is_orgunituid_valid(.data, df_orgs)
```

Arguments

df_orgs	orgs
data	df

Value

boolean

Examples

```
## Not run:
  data %>% is_orgunituid_valid(df_orgs)

## End(Not run)
```

`is_ou_valid` *Check if OperatingUnit is valid*

Description

Check if OperatingUnit is valid

Usage

```
is_ou_valid(.data, df_orgs)
```

Arguments

`df_orgs`	df_orgs
`data`	df

Value

boolean

Examples

```
## Not run:
  data %>% is_ou_valid(df_orgs)

## End(Not run)
```

load_lookups *Load DATIM Look up tables*

Description

Load DATIM Look up tables

Usage

```
load_lookups(datim_path = "./Data/", local = TRUE, user = NULL)
```

Arguments

datim_path	HFR Data folder ID
local	Read data from local directory?
user	User email address

Examples

```
## Not run:
 #load look up data
 load_lookups(datim_path = "./Data/datim")
 load_lookups(datim_path = "999#1aw####")

## End(Not run)
```

match_ignoredfiles *Identify S3 file names matching Ignored files*

Description

Identify S3 file names matching Ignored files

Usage

```
match_ignoredfiles(file_name, df_ref, ref_name = "name")
```

Arguments

file_name	Filename to be used as look up
df_ref	Reference data frame, Recommend using files present in S3 but not in gdrive, Eg: df_ghosts
ref_name	Column name to be used for lookup

Examples

```
## Not run:

  library(tidyverse)
  library(Wavelength)
  library(googlesheets4)

  ss_sbm <- as_sheets_id("<xyz>")

  df_ignore_files <- read_sheet(ss = ss_sbm, "ignore_files")

  df_new <- identify_newfiles() # this will also generate df_submissions & df_archive

  # Ghost files
  df_ghosts <- df_submissions %>%
     full_join(df_archive, by = c("name" = "sys_data_object")) %>%
     filter(is.na(exists_gdrive))

  # Match ignored files
  df_ignore_files <- df_ignore_files %>%
    mutate(names_s3 = match_ignoredfiles(name_googledrive, df_ghosts, "name")

  # Update ignore files table
  range_write(ss = sbm_form,
              data = df_ignore_files,
              sheet = "ignore_files",
              col_names = TRUE)

## End(Not run)
```

`opm_holiday` *OPM Holiday*

Description

Table of OPM Holidays for 2021-26 for scheduling submission dates.

Usage

```
data(opm_holiday)
```

Format

A data frame with each Federaly observed holiday from 2021-2026.

date_holiday_obs date holiday is observered

holiday holiday

Source

"https://www.opm.gov/policy-data-oversight/pay-leave/federal-holidays"

---

package_check *Check if package exists*

Description

Check if package exists

Usage

```
package_check(pkg)
```

Arguments

pkg	package name

Value

warning message if package is not installed

See Also

Other utility: `is_metatab()`, `var_exists()`

---

paint_blue *Paint console text in blue*

Description

Paint console text in blue

Usage

```
paint_blue(txt)
```

Arguments

txt	text to be printed

See Also

Other text_color: `paint_green()`, `paint_red()`, `paint_yellow()`

paint_green *Paint console text in green*

Description

Paint console text in green

Usage

```
paint_green(txt)
```

Arguments

txt	text to be printed

See Also

Other text_color: paint_blue(), paint_red(), paint_yellow()

paint_red *Paint console text in red*

Description

Paint console text in red

Usage

```
paint_red(txt)
```

Arguments

txt	text to be printed

See Also

Other text_color: paint_blue(), paint_green(), paint_yellow()

paint_yellow *Paint console text in yellow*

Description

Paint console text in yellow

Usage

```
paint_yellow(txt)
```

Arguments

txt	text to be printed

See Also

Other text_color: paint_blue(), paint_green(), paint_red()

parse_submission *Parse out submitted file components*

Description

Parse out submitted file components

Usage

```
parse_submission(pfile)
```

Arguments

sfile	processed file

Value

component As vector c("hfr_pd", "iso3", "mech_code", "pdate")

Examples

```
## Not run:
  parse_submission("HFR_2020.99_XWH_100000_processed_20200101.csv")

## End(Not run)
```

pull_hierarchy *Compile PEPFAR Hierarchy*

Description

Compile PEPFAR Hierarchy

Usage

```
pull_hierarchy(
  ou_uid,
  username,
  password,
  baseurl = "https://final.datim.org/",
  folderpath_output = NULL
)
```

Arguments

ou_uid	UID for the country, recommend using identify_ouuids()
username	DATIM username
password	DATIM password, recommend using mypwd()
baseurl	API base url, default = https://final.datim.org/
folderpath_output	provide the full path to the folder for saving

Examples

```
## Not run:
#get OU UID
  ouuid <- identify_ouuids() %>% dplyr::filter(ou == "Kenya")
#pull hierarchy (paths are all UIDs)
  df <- pull_hierarchy(ouuid, username = myuser, password = mypwd(myuser))
## End(Not run)
```

pull_mech *Pull Partner/Mechanism Info from DATIM*

Description

Pull Partner/Mechanism Info from DATIM

Usage

```
pull_mech(usaid_only = TRUE, ou_sel = NULL, folderpath_output = NULL)
```

Arguments

usaid_only	specify if only USAID mechansism should be returned, default = TRUE
ou_sel	option to specify an operating unit, default = NULL
folderpath_output	provide the full path to the folder for saving

Examples

```
## Not run:
#pull mechanism/partner information
df <- pull_mech()
## End(Not run)
```

---

pull_mer *Extract DATIM Results and Targets (DATIM API Call)*

Description

Extract DATIM Results and Targets (DATIM API Call)

Usage

```
pull_mer(
  ou_name = NULL,
  username,
  password,
  baseurl = "https://final.datim.org/",
  fy_pd = NULL,
  quarters_complete = NULL,
  folderpath_output = NULL
)
```

Arguments

ou_name	Operating Unit name, if mechanism is not specified
username	DATIM username
password	DATIM password, recommend using `mypwd()`
baseurl	API base url, default = https://final.datim.org/
fy_pd	fiscal year(s) to cover, default will be current FY if not provided
quarters_complete	no. of quarters completed through FY to determine weeks left in year
folderpath_output	folder path to store DATIM output, default = NULL

Examples

```
## Not run:
 #ou mer data
 myuser <- "UserX"
 mech_x_dta <- pull_mer(ou_name = "Namibia", username = myuser, password = mypwd(myuser))

## End(Not run)
```

`report_submissions_errors` *Report files validation*

Description

Report files validation

Usage

```
report_submissions_errors(df_files, mechanisms, export = FALSE)
```

Arguments

`df_files`	df of filename validation
`mechanisms`	mechs

Value

void

Examples

```
## Not run:
  report_file_errors(files, ims, processed)

## End(Not run)
```

revert_validations *Revert validated file to processed files*

Description

Revert validated file to processed files

Usage

```
revert_validations(dir_files)
```

Arguments

dir_files	location of processed files

Examples

```
## Not run:
revert_validations(dir_files)

## End(Not run)
```

stash_outgoing *Download from S3 and push to Google Drive*

Description

Download from S3 and push to Google Drive

Usage

```
stash_outgoing(
  prefix = c("HFR_Tableau", "HFR_Submission", "Mechanism", "Detailed"),
  outputfolder,
  gdrive = FALSE
)
```

Arguments

prefix	file prefix - "HFR_Tableau","HFR_Submission", "Mechanism", "Detailed"
outputfolder	folder path to store file
gdrive	whether to upload to Google Drive, default = FALSE

Examples

```
## Not run:
stash_outgoing("HFR_Tableau", "out/joint")
## End(Not run)
```

`template_cols_long` *Column Headers for HFR Long Template*

Description

List of column headers for the HFR Long Template

Usage

```
data(template_cols_long)
```

Format

A list of all headers

template_cols_long column names

`template_cols_meta` *Column Headers for HFR Indicator Meta Data*

Description

List of column headers for the HFR Indicator Meta Data

Usage

```
data(template_cols_meta)
```

Format

A list of all headers

template_cols_meta column names

template_cols_wide *Column Headers for HFR Wide Template*

Description

List of column headers for the HFR Wide Template

Usage

```
data(template_cols_wide)
```

Format

A list of all headers

template_cols_wide column names

template_cols_wide_lim *Column Headers for HFR Wide - Limited Template*

Description

List of column headers for the HFR Wide Template

Usage

```
data(template_cols_wide_lim)
```

Format

A list of all headers

template_cols_wide_lim column names

tidy_sitelist *Import and munge submitted site list*

Description

Import and munge submitted site list

Usage

```
tidy_sitelist(filepath, folderpath_output = NULL)
```

Arguments

filepath	path to sitelist file
folderpath_output	if output is desired, full folder path

Examples

```
## Not run:
  path <- "~/Data/HFR_FY21_SiteValidation_Kenya.xlsx"
  df_sites <- tidy_sitelist(path)
## End(Not run)
```

update_meta_mechs *Update USAID Mechanism meta table*

Description

Update USAID Mechanism meta table

Usage

```
update_meta_mechs(savefolder = "out/DATIM", upload = FALSE)
```

Arguments

savefolder	folderpath to save, default = "out/DATIM"
upload	should the new table be pushed to Google Drive and s3? default = FALSE

update_meta_mer *Update MER meta table*

Description

Update MER meta table

Usage

```
update_meta_mer(fy_pd = NULL, savefolder = "out/DATIM", upload = FALSE)
```

Arguments

fy_pd	fiscal year(s) to cover, default will be current FY if not provided
savefolder	folderpath to save, default = "out/DATIM"
upload	should the new table be pushed to s3? default = FALSE

update_meta_orgs *Update Organization Hierarchy meta table*

Description

Update Organization Hierarchy meta table

Usage

```
update_meta_orgs(savefolder = "out/DATIM", upload = FALSE)
```

Arguments

savefolder	folderpath to save, default = "out/DATIM"
upload	should the new table be pushed to Google Drive and s3? default = FALSE

update_meta_targets *Update MER targets from MSD (prior to site results being available)*

Description

Update MER targets from MSD (prior to site results being available)

Usage

```
update_meta_targets(fy, savefolder = "out/DATIM", upload = FALSE)
```

Arguments

fy	fiscal year
savefolder	folderpath to save, default = "out/DATIM"
upload	should the new table be pushed to s3? default = FALSE

update_operatingunits *Update invalid operating units*

Description

Update invalid operating units

Usage

```
update_operatingunits(hfr_data, levels, orgs, ims = NULL)
```

Arguments

hfr_data	processed hfr data
levels	datim org levels
orgs	datim org hierarchy
ims	datim mechanisms

Value

hfr_data df

Examples

```
## Not run:
  update_operatingunits(hfr_df, levels=org_levels, orgs=org_hierarchy, ims=mechanisms)

## End(Not run)
```

upload_meta_table *Push meta tables to Google Drive and s3*

Description

Push meta tables to Google Drive and s3

Usage

```
upload_meta_table(type = c("mech", "org"), folder = "out/DATIM")
```

Arguments

type	table type, "mech" or "org"
folder	where is the file stored? default = "out/DATIM"

Examples

```
## Not run:
 #pull updated mechanism table
   pull_mech(folderpath_output = savefolder)
 #upload to Google Drive and s3
   upload_meta_table("mech")
## End(Not run)
```

validate_date *Validate HFR PD Date*

Description

Validate HFR PD Date

Usage

```
validate_date(df_pds, pdate, pd)
```

Arguments

df_pds	hfr period dates
pdate	period date
pd	reporting period

Value

valid: True / False

Examples

```
## Not run:
  validate_date(df_pds = valid_dates, '2020-01-27', 5)

## End(Not run)
```

---

validate_hfr_data *Validate processed hfr data*

Description

Validate processed hfr data

Usage

```
validate_hfr_data(hfr_data, orgs, ims, dates, keep_values = FALSE)
```

Arguments

hfr_data	processed hfr data
orgs	datim org hierarchy
ims	datim mechanisms
dates	hfr valid dates
keep_values	Keep values along the error flags

Value

errors data frame

Examples

```
## Not run:
 validate_hfr_data(df_hfr_data, orgs=org_hierarchy, ims=df_mechanisms, dates=df_hfr_dates)
 validate_hfr_data(df_hfr_data, orgs=org_hierarchy, ims=df_mechanisms, dates=df_hfr_dates, keep_values = TRUE)

## End(Not run)
```

validate_import *Validation on import*

Description

Runs validation on data after reading in the dataset - making sure it has the required column based on the template type and check if there is one ore more operating units.

Usage

```
validate_import(df)
```

Arguments

df	df create during hfr_import()

See Also

Other validation: check_content(), check_dates(), check_disaggs(), check_orgunituids(), check_output_cols(), is_hfrtab(), validate_initial(), validate_output()

validate_initial *Initial Validation on Submitted template*

Description

Performs basic check before importing data - checking if there are any HFR labeled tabs to import and then outputting info from the meta tab (country, filename, and template version) and then about the tabs (what will be imported or excluded)

Usage

```
validate_initial(filepath)
```

Arguments

filepath	filepath to submitted template

See Also

Other validation: check_content(), check_dates(), check_disaggs(), check_orgunituids(), check_output_cols(), is_hfrtab(), validate_import(), validate_output()

validate_mechanism *Validate mechanism code*

Description

Validate mechanism code

Usage

```
validate_mechanism(mechanisms, ou, mcode)
```

Arguments

mechanisms	df of mechs
mech_code	mech code

Value

vector c(valid_im, mech_name)

Examples

```
## Not run:
  validate_mechanism(ims, 'Angola', 16172)

## End(Not run)
```

validate_orgunit *Validate org unit uid*

Description

Validate org unit uid

Usage

```
validate_orgunit(df_orgs, ou, uid)
```

Arguments

df_orgs	org hierarchy
ou	operating unit
uid	orgunituid

Value

valid as a vector c(valid_uid, valid_uid_ou)

Examples

```
## Not run:
  validate_orgunit(df_orgs, 'Eswatini', 'g48XD8px8NN')

## End(Not run)
```

`validate_output` *Validation Checks*

Description

Runs a number of validations after the tidying has occured. Additional, optional validations against DATIM data can be run if data are available, from `pull_hierarchy`,`pull_mech`, `pull_mer`

Usage

```
validate_output(df, output_path, datim_path = NULL)
```

Arguments

`df`	HFR data framed created by `hfr_process_template()`
`datim_path`	path to look up files from `pull_hierarchy`,`pull_mech`, `pull_mer`

See Also

Other validation: `check_content()`, `check_dates()`, `check_disaggs()`, `check_orgunituids()`, `check_output_cols()`, `is_hfrtab()`, `validate_import()`, `validate_initial()`

`validate_submission` *Validate submitted files*

Description

Validate submitted files

Usage

```
validate_submission(pfile, levels, ims)
```

Arguments

levels	org levels
ims	mechanisms df
sfile	processed file

Value

vector c("fy", "hfr_pd", "iso3", "operatingunit", "mech_code", "mech_valid", "mech_name", "pdate", "name")

Examples

```
## Not run:
  validate_submissions("HFR_2020.99_XWH_100000_processed_20200101.csv")

## End(Not run)
```

---

validate_submissions *Validate files*

Description

Validate files

Usage

```
validate_submissions(pfolder, levels, ims, pattern = NULL)
```

Arguments

levels	org levels
ims	mechanisms df
pattern	filename pattern
folder	pfolder

Value

dataframe

Examples

```
## Not run:
  validate_submissions(dir_hfr_pd205, pattern = "HFR_2020.05")

## End(Not run)
```

`var_exists` *Check if variable exist*

Description

Check if variable exist

Usage

```
var_exists(df, var)
```

Arguments

`df`	data frame to check against
`var`	quoted variable of interest

See Also

Other utility: `is_metatab()`, `package_check()`

Examples

```
## Not run:
var_exists(df, "val")
## End(Not run)
```

Index