Package 'soilDB'

August 20, 2022

Type Package

Title Soil Database Interface

Version 2.7.3

Author Dylan Beaudette [aut],
Jay Skovlin [aut],
Stephen Roecker [aut],
Andrew Brown [aut, cre]

Maintainer Andrew Brown <andrew.g.brown@usda.gov>

Description A collection of functions for reading data from USDA-NCSS soil databases.

License GPL (>= 3)

LazyLoad yes

Depends R (>= 3.5.0)

Imports grDevices, graphics, stats, utils, methods, aqp, data.table,
DBI, curl

Suggests jsonlite, httr, xml2, rvest, sf, wk, terra, raster, odbc,
RSQLite, testthat, scales

Repository CRAN

URL http://ncss-tech.github.io/soilDB/,
http://ncss-tech.github.io/AQP/

BugReports https://github.com/ncss-tech/soilDB/issues

RoxygenNote 7.2.1

Encoding UTF-8

LazyData false

NeedsCompilation no

Date/Publication 2022-08-20 00:10:06 UTC

R topics documented:

soilDB-package *Soil Database Interface*

Description

A collection of functions for reading data from USDA-NCSS soil databases.

Details

This package provides methods for extracting soils information from local PedonPC and AKSite databases (MS Access format), local NASIS databases (MS SQL Server), Soil Data Access and various other soil-related web services.

Author(s)

J.M. Skovlin, D.E. Beaudette, S.M Roecker, A.G. Brown @seealso `fetchPedonPC`, `fetchNASIS`, `SDA_query`, `loafercreek`

createSSURGO *Create a SQLite database or GeoPackage from one or more SSURGO Exports*

Description

Create a SQLite database or GeoPackage from one or more SSURGO Exports

Usage

```
createSSURGO(
  filename,
  exdir,
  pattern = NULL,
  include_spatial = TRUE,
  overwrite = FALSE,
  header = FALSE,
  quiet = TRUE,
  ...
)
```

Arguments

`filename`	Output file name (e.g. `'db.sqlite'` or `'db.gpkg'`)
`exdir`	Path containing containing SSURGO spatial (.shp) and tabular (.txt) files.
`pattern`	Character. Optional regular expression to use to filter subdirectories of `exdir`. Default: `NULL` will search all subdirectories for SSURGO export files.

include_spatial
: Logical. Include spatial data layers in database? Default: TRUE.

overwrite
: Logical. Overwrite existing layers? Default FALSE will append to existing tables/layers.

header
: Logical. Passed to read.delim() for reading pipe-delimited (|) text files containing tabular data.

quiet
: Logical. Suppress messages and other output from database read/write operations?

...
: Additional arguments passed to write_sf() for writing spatial layers.

Value

Character. Vector of layer/table names in filename.

Examples

```
## Not run:
 downloadSSURGO("areasymbol IN ('CA067', 'CA077', 'CA632')", destdir = "SSURGO_test")
 createSSURGO("test.gpkg", "SSURGO_test")

## End(Not run)
```

---

createStaticNASIS *Create a memory or file-based instance of NASIS database*

Description

Create a memory or file-based instance of NASIS database for selected tables.

Usage

```
createStaticNASIS(
  tables = NULL,
  new_names = NULL,
  SS = TRUE,
  dsn = NULL,
  output_path = NULL,
  verbose = FALSE
)
```

Arguments

tables
: Character vector of target tables. Default: NULL is whatever tables are listed by DBI::dbListTables for the connection typ being used.

new_names
: Optional: new table names (should match length of vector of matching tables in dsn)

SS	Logical. Include "selected set" tables (ending with suffix "_View_1"). Default: TRUE
dsn	Optional: path to SQLite database containing NASIS table structure; or a DBIConnection. Default: NULL
output_path	Optional: path to new/existing SQLite database to write tables to. Default: NULL returns table results as named list.
verbose	Show error messages from attempts to dump individual tables? Default FALSE

Value

A named list of results from calling dbQueryNASIS for all columns in each NASIS table.

---

dbConnectNASIS *Create local NASIS database connection*

Description

Create a connection to a local NASIS database with DBI

Usage

```
dbConnectNASIS(dsn = NULL)

NASIS(dsn = NULL)
```

Arguments

dsn	Optional: path to SQLite database containing NASIS table structure; Default: NULL

Value

A DBIConnection object, as returned by DBI::dbConnect(). If dsn is a DBIConnection, the attribute isUserDefined of the result is set to TRUE. If the DBIConnection is created by the internal NASIS connection process, isUserDefined is set to FALSE.

dbQueryNASIS *Query a NASIS DBIConnection*

Description

Send queries to a NASIS DBIConnection

Usage

```
dbQueryNASIS(conn, q, close = TRUE, ...)
```

Arguments

conn	A DBIConnection object, as returned by DBI::dbConnect().
q	A statement to execute using DBI::dbGetQuery; or a (named) vector containing multiple statements to evaluate separately
close	Close connection after query? Default: TRUE
...	Additional arguments to DBI::dbGetQuery

Value

Result of DBI::dbGetQuery

downloadSSURGO *Get SSURGO ZIP files from Web Soil Survey 'Download Soils Data'*

Description

Download ZIP files containing spatial (ESRI shapefile) and tabular (TXT) files with standard SSURGO format; optionally including the corresponding SSURGO Template Database with include_template=TRUE.

Usage

```
downloadSSURGO(
  WHERE = NULL,
  areasymbols = NULL,
  destdir = tempdir(),
  exdir = destdir,
  include_template = FALSE,
  extract = TRUE,
  remove_zip = FALSE,
  overwrite = FALSE,
  quiet = FALSE
)
```

Arguments

WHERE	A SQL WHERE clause expression used to filter records in sacatalog table. Alternately WHERE can be any spatial object supported by SDA_spatialQuery() for defining the target extent.
areasymbols	Character vector of soil survey area symbols e.g. c("CA067", "CA077"). Used in lieu of WHERE arugment.
destdir	Directory to download ZIP files into. Default tempdir().
exdir	Directory to extract ZIP archives into. May be a directory that does not yet exist. Each ZIP file will extract to a folder labeled with areasymbol in this directory. Default: destdir
include_template	Include the (possibly state-specific) MS Access template database? Default: FALSE
extract	Logical. Extract ZIP files to exdir? Default: TRUE
remove_zip	Logical. Remove ZIP files after extracting? Default: FALSE
overwrite	Logical. Overwrite by re-extracting if directory already exists? Default: FALSE
quiet	Logical. Passed to curl::curl_download().

Details

To specify the Soil Survey Areas you would like to obtain data you use a WHERE clause for query of sacatalog table such as areasymbol = 'CA067', "areasymbol IN ('CA628', 'CA067')" or areasymbol LIKE 'CT%'.

Pipe-delimited TXT files are found in */tabular/* folder extracted from a SSURGO ZIP. The files are named for tables in the SSURGO schema. There is no header / the files do not have column names. See the *Soil Data Access Tables and Columns Report*: https://sdmdataaccess.nrcs.usda.gov/documents/TablesAndColumnsReport.pdf for details on tables, column names and metadata including the default sequence of columns used in TXT files. The function returns a try-error if the WHERE/areasymbols arguments result in

Several ESRI shapefiles are found in the */spatial/* folder extracted from a SSURGO ZIP. These have prefix soilmu_ (mapunit), soilsa_ (survey area), soilsf_ (special features). There will also be a TXT file with prefix soilsf_ describing any special features. Shapefile names then have an a_ (polygon), l_ (line), p_ (point) followed by the soil survey area symbol.

Value

Character. Paths to downloaded ZIP files (invisibly). May not exist if remove_zip = TRUE.

`estimateColorMixture` *Estimate color mixtures using weighted average of CIELAB color coordinates*

Description

Estimate color mixtures using weighted average of CIELAB color coordinates

Usage

```
estimateColorMixture(x, wt = "pct", backTransform = FALSE)
```

Arguments

`x`	data.frame, typically from NASIS containing at least CIE LAB ('L', 'A', 'B') and some kind of weight
`wt`	fractional weights, usually area of hz face
`backTransform`	logical, should the mixed sRGB representation of soil color be transformed to closest Munsell chips? This is performed by aqp::rgb2Munsell default: `FALSE`

Value

A data.frame containing estimated color mixture

Note

See `mixMunsell` for a more realistic (but slower) simulation of subtractive mixing of pigments.

Author(s)

D.E. Beaudette

`estimateSTR` *Estimate Soil Temperature Regime*

Description

Estimate soil temperature regime (STR) based on mean annual soil temperature (MAST), mean summer temperature (MSST), mean winter soil temperature (MWST), presence of O horizons, saturated conditions, and presence of permafrost. Several assumptions are made when O horizon or saturation are undefined.

Usage

```
estimateSTR(
  mast,
  mean.summer,
  mean.winter,
  O.hz = NA,
  saturated = NA,
  permafrost = FALSE
)
```

Arguments

`mast`	vector of mean annual soil temperature (deg C)
`mean.summer`	vector of mean summer soil temperature (deg C)
`mean.winter`	vector of mean winter soil temperature (deg C)
`O.hz`	logical vector of O horizon presence / absence
`saturated`	logical vector of seasonal saturation
`permafrost`	logical vector of permafrost presence / absence

Details

Soil Temperature Regime Evaluation Tutorial

Value

Vector of soil temperature regimes.

Author(s)

D.E. Beaudette

References

Soil Survey Staff. 2015. Illustrated guide to soil taxonomy. U.S. Department of Agriculture, Natural Resources Conservation Service, National Soil Survey Center, Lincoln, Nebraska.

See Also

`STRplot`

Examples

```
# simple example
estimateSTR(mast=17, mean.summer = 22, mean.winter = 12)
```

`fetchKSSL` *Get Kellogg Soil Survey Laboratory Data from SoilWeb snapshot*

Description

Download soil characterization and morphologic data via BBOX, MLRA, or soil series name query, from the KSSL database.

Usage

```
fetchKSSL(
  series = NA,
  bbox = NA,
  mlra = NA,
  pedlabsampnum = NA,
  pedon_id = NA,
  pedon_key = NA,
  returnMorphologicData = FALSE,
  returnGeochemicalData = FALSE,
  simplifyColors = FALSE,
  progress = TRUE
)
```

Arguments

`series`	vector of soil series names, case insensitive
`bbox`	a single bounding box in WGS84 geographic coordinates e.g. `c(-120, 37, -122, 38)`
`mlra`	vector of MLRA IDs, e.g. "18" or "22A"
`pedlabsampnum`	vector of KSSL pedon lab sample number
`pedon_id`	vector of user pedon ID
`pedon_key`	vector of KSSL internal pedon ID
`returnMorphologicData`	logical, optionally request basic morphologic data, see details section
`returnGeochemicalData`	logical, optionally request geochemical, optical and XRD/thermal data, see details section
`simplifyColors`	logical, simplify colors (from morphologic data) and join with horizon data
`progress`	logical, optionally give progress when iterating over multiple requests

Details

This is an experimental interface to a subset for the most commonly used data from a snapshot of KSSL (lab characterization) and NASIS (morphologic) data.

Series-queries are case insensitive. Series name is based on the "correlated as" field (from KSSL snapshot) when present. The "sampled as" classification was promoted to "correlated as" if the "correlated as" classification was missing.

When `returnMorphologicData` is TRUE, the resulting object is a list. The standard output from `fetchKSSL` (`SoilProfileCollection` object) is stored in the named element "SPC". The additional elements are basic morphologic data: soil color, rock fragment volume, pores, structure, and redoximorphic features. There is a 1:many relationship between the horizon data in "SPC" and the additional dataframes in `morph`. See examples for ideas on how to "flatten" these tables.

When `returnGeochemicalData` is TRUE, the resulting object is a list. The standard output from `fetchKSSL` (`SoilProfileCollection` object) is stored in the named element "SPC". The additional elements are geochemical and mineralogy analysis tables, specifically: geochemical/elemental analyses "geochem", optical mineralogy "optical", and X-ray diffraction / thermal "xrd_thermal". `returnGeochemicalData` will include additional dataframes `geochem`, `optical`, and `xrd_thermal` in list result.

Setting `simplifyColors=TRUE` will automatically flatten the soil color data and join to horizon level attributes.

Function arguments (`series`, `mlra`, etc.) are fully vectorized except for `bbox`.

Value

a `SoilProfileCollection` object when `returnMorphologicData` is FALSE, otherwise a list.

Note

SoilWeb maintains a snapshot of these KSSL and NASIS data. The SoilWeb snapshot was developed using methods described here: https://github.com/dylanbeaudette/process-kssl-snapshot. Please use the link below for the live data.

Author(s)

D.E. Beaudette and A.G. Brown

References

http://ncsslabdatamart.sc.egov.usda.gov/

See Also

fetchOSD

Examples

```
if(requireNamespace("curl") &
    curl::has_internet()) {

    library(aqp)

    # search by series name
    s <- fetchKSSL(series='auburn')

    # search by bounding-box
    # s <- fetchKSSL(bbox=c(-120, 37, -122, 38))

    # how many pedons
    length(s)

    # plot
    plotSPC(s, name='hzn_desgn', max.depth=150)

    ##
    ## morphologic data
    ##

    # get lab and morphologic data
    s <- fetchKSSL(series='auburn', returnMorphologicData = TRUE)

    # extract SPC
    pedons <- s$SPC

    ## automatically simplify color data
    s <- fetchKSSL(series='auburn', returnMorphologicData = TRUE, simplifyColors=TRUE)

    # check
    par(mar=c(0,0,0,0))
    plot(pedons, color='moist_soil_color', print.id=FALSE)

}
```

`fetchLDM` *Query data from Kellogg Soil Survey Laboratory Data Mart via Soil Data Access or local SQLite snapshot*

Description

LDM model diagram: https://jneme910.github.io/Lab_Data_Mart_Documentation/Documents/SDA_KSSL_Data_model.html

Usage

```
fetchLDM(
  x = NULL,
  what = "pedlabsampnum",
  bycol = "pedon_key",
  tables = c("lab_physical_properties", "lab_chemical_properties",
   "lab_calculations_including_estimates_and_default_values", "lab_rosetta_Key"),
  chunk.size = 1000,
  ntries = 3,
  layer_type = c("horizon", "layer", "reporting layer"),
  prep_code = c("S", ""),
  analyzed_size_frac = c("<2 mm", ""),
  dsn = NULL
)
```

Arguments

x	A vector of values to find in column specified by what, default NULL uses no constraints on what
what	A single column name from tables: lab_combine_nasis_ncss, lab_webmap, lab_site, lab_pedon or lab_area
bycol	A single column name from lab_layer used for processing chunks; default: "pedon_key"
tables	A vector of table names; Default is "lab_physical_properties", "lab_chemical_properties", "lab_calculations_including_estimates_and_default_values", and "lab_rosetta_Key". May also include one or more of: "lab_mir", "lab_mineralogy_glass_count", "lab_major_and_trace_elements_and_oxides", "lab_xray_and_thermal" but it will be necessary to select appropriate prep_code and analyzed_size_frac for your analysis (see *Details*).
chunk.size	Number of pedons per chunk (for queries that may exceed maxJsonLength)
ntries	Number of tries (times to halve chunk.size) before returning NULL; default 3
layer_type	Default: "horizon", "layer", and "reporting layer"
prep_code	Default: "S" and "". May also include one or more of: "F", "HM", "HM_SK" "GP", "M", "N", or "S"
analyzed_size_frac	Default: "<2 mm" and "". May also include one or more of: "<0.002 mm", "0.02-0.05 mm", "0.05-0.1 mm", "0.1-0.25 mm", "0.25-0.5 mm", "0.5-1 mm", "1-2 mm", "0.02-2 mm", "0.05-2 mm"
dsn	Data source name; either a path to a SQLite database, an open DBIConnection or (default) NULL (to use soilDB::SDA_query)

Details

If the chunk.size parameter is set too large and the Soil Data Access request fails, the algorithm will re-try the query with a smaller (halved) chunk.size argument. This will be attempted up to 3 times before returning NULL

Currently the lab_area tables are joined only for the "Soil Survey Area" related records.

When requesting data from "lab_major_and_trace_elements_and_oxides", "lab_mineralogy_glass_count", or "lab_xray_and_thermal" multiple preparation codes (prep_code) or size fractions (analyzed_size_frac) are possible. The default behavior of fetchLDM() is to attempt to return a topologically valid (minimal overlaps) *SoilProfileCollection*. This is achieved by setting prep_code="S" ("sieved") and analyzed_size_frac="<2 mm". You may specify alternate or additional preparation codes or fractions as needed, but note that this may cause "duplication" of some layers where measurements were made with different preparation or on fractionated samples

Value

a SoilProfileCollection for a successful query, a try-error if no site/pedon locations can be found or NULL for an empty lab_layer (within sites/pedons) result

Examples

```
## Not run:

if(requireNamespace("curl") &
   curl::has_internet()) {

  # fetch by ssa_key
  res <- fetchLDM(8297, what = "ssa_key")

  # physical properties correlated as taxonomic subgroup "Typic Argialbolls"
  res <- fetchLDM(x = "Typic Argialbolls",
                  what = "corr_taxsubgrp",
                  tables = "lab_physical_properties")

  # fetch by area_code (SSA only)
  res <- fetchLDM("CA630", what = "area_code")
}


## End(Not run)
```

---

fetchNASIS *Get a pedon or component data SoilProfileCollection from NASIS*

Description

Fetch commonly used site/pedon/horizon data or component from NASIS, returned as a SoilProfileCollection object.

Usage

```
fetchNASIS(
  from = "pedons",
  url = NULL,
```

```
  SS = TRUE,
  rmHzErrors = FALSE,
  nullFragsAreZero = TRUE,
  soilColorState = "moist",
  mixColors = TRUE,
  lab = FALSE,
  fill = FALSE,
  dropAdditional = TRUE,
  dropNonRepresentative = TRUE,
  duplicates = FALSE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_RMF_from_NASIS_db(SS = TRUE, dsn = NULL)

get_concentrations_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_phfmp_from_NASIS_db(SS = TRUE, stringsAsFactors = NULL, dsn = NULL)
```

Arguments

from	determines what objects should fetched? ('pedons' | 'components' | 'pedon_report')
url	string specifying the url for the NASIS pedon_report (default: NULL)
SS	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: TRUE)
rmHzErrors	should pedons with horizon depth errors be removed from the results? (default: FALSE)
nullFragsAreZero	should fragment volumes of NULL be interpreted as 0? (default: TRUE), see details
soilColorState	Used only for from='pedons'; which colors should be used to generate the convenience field soil_color? ('moist' or 'dry')
mixColors	should mixed colors be calculated (Default: TRUE) where multiple colors are populated for the same moisture state in a horizon? FALSE takes the dominant color for each horizon moist/dry state.
lab	should the phlabresults child table be fetched with site/pedon/horizon data (default: FALSE)
fill	include pedon or component records without horizon data in result? (default: FALSE)
dropAdditional	Used only for from='components' with duplicates=TRUE. Prevent "duplication" of mustatus=="additional" mapunits? Default: TRUE

dropNonRepresentative
: Used only for `from='components'` with `duplicates=TRUE`. Prevent "duplication" of non-representative data mapunits? Default: `TRUE`

duplicates
: Used only for `from='components'`. Duplicate components for all instances of use (i.e. one for each legend data mapunit is used on; optionally for additional mapunits, and/or non-representative data mapunits?)

stringsAsFactors
: deprecated

dsn
: Optional: path to local SQLite database containing NASIS table structure; default: `NULL`

Details

This function imports data from NASIS into R as a `SoilProfileCollection` object. It "flattens" NASIS pedon and component tables, including their child tables, into several more easily manageable data frames. Primarily these functions access the local NASIS database using an ODBC connection. However using the `fetchNASIS()` argument `from = "pedon_report"`, data can be read from the NASIS Report 'fetchNASIS', as either a txt file or url. The primary purpose of `fetchNASIS(from = "pedon_report")` is to facilitate importing datasets larger than 8000+ pedons/components.

The value of `nullFragsAreZero` will have a significant impact on the rock fragment fractions returned by fetchNASIS. Set `nullFragsAreZero = FALSE` in those cases where there are many data-gaps and `NULL` rock fragment values should be interpreted as `NULL`. Set `nullFragsAreZero = TRUE` in those cases where `NULL` rock fragment values should be interpreted as 0.

This function attempts to do most of the boilerplate work when extracting site/pedon/horizon or component data from a local NASIS database. Pedons that are missing horizon data, or have errors in their horizonation are excluded from the returned object, however, their IDs are printed on the console. Pedons with combination horizons (e.g. B/C) are erroneously marked as errors due to the way in which they are stored in NASIS as two overlapping horizon records.

Tutorials:

- fetchNASIS Pedons Tutorial
- fetchNASIS Components Tutorial

Value

A SoilProfileCollection object

Author(s)

D. E. Beaudette, J. M. Skovlin, S.M. Roecker, A.G. Brown

See Also

`get_component_data_from_NASIS()`

fetchNASISLabData *Get NCSS Pedon laboratory data from NASIS*

Description

Fetch KSSL laboratory pedon/horizon layer data from a local NASIS database, return as a SoilProfileCollection object.

Usage

```
fetchNASISLabData(SS = TRUE, dsn = NULL)
```

Arguments

SS	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: TRUE)#'
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Details

This function currently works only on Windows, and requires a 'nasis_local' ODBC connection.

Value

a SoilProfileCollection object

Author(s)

J.M. Skovlin and D.E. Beaudette

See Also

get_labpedon_data_from_NASIS_db

fetchNASISWebReport *Get component tables from NASIS Web Reports*

Description

Get component tables from NASIS Web Reports

Usage

```
fetchNASISWebReport(
  projectname,
  rmHzErrors = FALSE,
  fill = FALSE,
  stringsAsFactors = NULL
)

get_component_from_NASISWebReport(projectname, stringsAsFactors = NULL)

get_chorizon_from_NASISWebReport(
  projectname,
  fill = FALSE,
  stringsAsFactors = NULL
)

get_legend_from_NASISWebReport(
  mlraoffice,
  areasymbol,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

get_lmuaoverlap_from_NASISWebReport(
  areasymbol,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

get_mapunit_from_NASISWebReport(
  areasymbol,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

get_projectmapunit_from_NASISWebReport(projectname, stringsAsFactors = NULL)

get_projectmapunit2_from_NASISWebReport(
  mlrassoarea,
  fiscalyear,
  projectname,
  stringsAsFactors = NULL
)

get_project_from_NASISWebReport(mlrassoarea, fiscalyear)

get_progress_from_NASISWebReport(mlrassoarea, fiscalyear, projecttypename)
```

```
get_project_correlation_from_NASISWebReport(
  mlrassoarea,
  fiscalyear,
  projectname
)

get_cosoilmoist_from_NASISWebReport(
  projectname,
  impute = TRUE,
  stringsAsFactors = NULL
)

get_sitesoilmoist_from_NASISWebReport(usiteid)
```

Arguments

projectname	text string vector of project names to be inserted into a SQL WHERE clause (default: NA)
rmHzErrors	should pedons with horizonation errors be removed from the results? (default: FALSE)
fill	should rows with missing component ids be removed (default: FALSE)
stringsAsFactors	deprecated
mlraoffice	text string value identifying the MLRA Regional Soil Survey Office group name inserted into a SQL WHERE clause (default: NA)
areasymbol	text string value identifying the area symbol (e.g. IN001 or IN%) inserted into a SQL WHERE clause (default: NA) NULL (default: TRUE)
droplevels	logical: indicating whether to drop unused levels in classifying factors. This is useful when a class has large number of unused classes, which can waste space in tables and figures.
mlrassoarea	text string value identifying the MLRA Soil Survey Office areasymbol symbol inserted into a SQL WHERE clause (default: NA)
fiscalyear	text string value identifying the fiscal year inserted into a SQL WHERE clause (default: NA)
projecttypename	text string value identifying the project type name inserted into a SQL WHERE clause (default: NA)
impute	replace missing (i.e. NULL) values with "Not_Populated" for categorical data, or the "RV" for numeric data or 201 cm if the "RV" is also NULL (default: TRUE)
usiteid	character: User Site IDs

Value

A data.frame or list with the results.

Author(s)

Stephen Roecker

`fetchOSD` *Get Official Series Descriptions and summaries from SoilWeb API*

Description

This function fetches a variety of data associated with named soil series, extracted from the USDA-NRCS Official Series Description text files and detailed soil survey (SSURGO). These data are periodically updated and made available via SoilWeb.

Usage

```
fetchOSD(soils, colorState = "moist", extended = FALSE)
```

Arguments

`soils`	a character vector of named soil series; case-insensitive
`colorState`	color state for horizon soil color visualization: "moist" or "dry"
`extended`	if `TRUE` additional soil series summary data are returned, see details

Details

- overview of all soil series query functions
- competing soil series
- siblings

The standard set of "site" and "horizon" data are returned as a `SoilProfileCollection` object (`extended=FALSE`. The "extended" suite of summary data can be requested by setting `extended=TRUE`. The resulting object will be a `list` with the following elements:)

SPC `SoilProfileCollection` containing standards "site" and "horizon" data

competing competing soil series from the SC database snapshot

geog_assoc_soils geographically associated soils, extracted from named section in the OSD

geomcomp empirical probabilities for geomorphic component, derived from the current SSURGO snapshot

hillpos empirical probabilities for hillslope position, derived from the current SSURGO snapshot

mtnpos empirical probabilities for mountain slope position, derived from the current SSURGO snapshot

terrace empirical probabilities for river terrace position, derived from the current SSURGO snapshot

flats empirical probabilities for flat landscapes, derived from the current SSURGO snapshot

shape_across empirical probabilities for surface shape (across-slope) from the current SSURGO snapshot

shape_down empirical probabilities for surface shape (down-slope) from the current SSURGO snapshot

pmkind empirical probabilities for parent material kind, derived from the current SSURGO snapshot

pmorigin empirical probabilities for parent material origin, derived from the current SSURGO snapshot

mlra empirical MLRA membership values, derived from the current SSURGO snapshot

climate experimental climate summaries from PRISM stack (CONUS only)

NCCPI select quantiles of NCCPI and Irrigated NCCPI, derived from the current SSURGO snapshot

metadata metadata associated with SoilWeb cached summaries

When using `extended = TRUE`, there are a couple of scenarios in which series morphology contained in `SPC` do not fully match records in the associated series summaries (e.g. `competing`).

1. **A query for soil series that exist entirely outside of CONUS (e.g. PALAU).** - Climate summaries are empty `data.frames` because these summaries are currently generated from PRISM. We are working on a solution that uses DAYMET.
2. **A query for data within CONUS, but OSD morphology missing due to parsing error (e.g. formatting, typos).** - Extended summaries are present but morphology missing from `SPC`. A warning is issued.

These last two cases are problematic for analysis that makes use of morphology and extended data, such as outlined in this tutorial on competing soil series.

Value

a `SoilProfileCollection` object containing basic soil morphology and taxonomic information.

Author(s)

D.E. Beaudette, A.G. Brown

References

USDA-NRCS OSD search tools: https://www.nrcs.usda.gov/wps/portal/nrcs/detailfull/soils/home/?cid=nrcs142p2_053587

See Also

OSDquery, siblings

Examples

```
if(requireNamespace("curl") &
   curl::has_internet()) {

  # soils of interest
  s.list <- c('musick', 'cecil', 'drummer', 'amador', 'pentz',
              'reiff', 'san joaquin', 'montpellier', 'grangeville', 'pollasky', 'ramona')

  # fetch and convert data into an SPC
  s.moist <- fetchOSD(s.list, colorState='moist')
  s.dry <- fetchOSD(s.list, colorState='dry')

  # plot profiles
  # moist soil colors
  if(require("aqp")) {

    par(mar=c(0,0,0,0), mfrow=c(2,1))
    plot(s.moist, name='hzname',
         cex.names=0.85, axis.line.offset=-4)
    plot(s.dry, name='hzname',
         cex.names=0.85, axis.line.offset=-4)

    # extended mode: return a list with SPC + summary tables
    x <- fetchOSD(s.list, extended = TRUE, colorState = 'dry')

    par(mar=c(0,0,1,1))
    plot(x$SPC)
    str(x, 1)


  # use makeChunks() for iteration over larger sequences of soil series
  s.list <- c('musick', 'cecil', 'drummer', 'amador', 'pentz',
              'reiff', 'san joaquin', 'montpellier', 'grangeville', 'pollasky', 'ramona')

  # make a vector of chunk IDs, with 2 series / chunk
  ck <- makeChunks(s.list, size = 2)

  # split original data by chunk IDs
  # iterate over resulting list
  # run fetchOSD() on pieces
  # result is a list of SoilProfileCollection objects
  x <- lapply(split(s.list, ck), fetchOSD)

  # flatten into a single SPC
  x <- combine(x)

  # there should be 11 profiles
  length(x)
  }
```

```
}
```

`fetchPedonPC` *Get a SoilProfileCollection from a PedonPC v.5 database*

Description

Fetch commonly used site/horizon data from a version 5.x PedonPC database, return as a SoilProfileCollection object.

Usage

```
fetchPedonPC(dsn)

getHzErrorsPedonPC(dsn, strict = TRUE)
```

Arguments

`dsn`	The path to a PedonPC version 6.x database
`strict`	Use "strict" horizon error checking? Default: `TRUE`

Value

a SoilProfileCollection class object

Note

This function attempts to do most of the boilerplate work when extracting site/horizon data from a PedonPC or local NASIS database. Pedons that have errors in their horizonation are excluded from the returned object, however, their IDs are printed on the console. See `getHzErrorsPedonPC` for a simple approach to identifying pedons with problematic horizonation. Records from the 'taxhistory' table are selected based on 1) most recent record, or 2) record with the least amount of missing data.

Author(s)

D. E. Beaudette and J. M. Skovlin

See Also

`get_hz_data_from_pedon_db`

`fetchRaCA` *Get Rapid Carbon Assessment (RaCA) data*

Description

Get Rapid Carbon Assessment (RaCA) data by state, geographic bounding-box, RaCA site ID, or soil series query from the SoilWeb API. This interface to the data was an experimental delivery service that does not include the latest soil carbon measurements.

Please use current RaCA distribution if you need measured SOC.

This interface will be updated sometime calendar year 2022 to include the latest soil morphology, taxonomic classification, and measured SOC values. More detailed coordinates for sample sites should also be available.

See https://www.nrcs.usda.gov/Internet/FSE_MANUSCRIPTS/alabama/raca_download/raca_download.zip for direct download of the full dataset.

Usage

```
fetchRaCA(
  series = NULL,
  bbox = NULL,
  state = NULL,
  rcasiteid = NULL,
  get.vnir = FALSE
)
```

Arguments

`series`	a soil series name; case-insensitive
`bbox`	a bounding box in WGS84 geographic coordinates e.g. `c(-120, 37, -122, 38)`, constrained to a 5-degree block
`state`	a two-letter US state abbreviation; case-insensitive
`rcasiteid`	a RaCA site id (e.g. 'C1609C01')
`get.vnir`	logical, should associated VNIR spectra be downloaded? (see details)

Details

The VNIR spectra associated with RaCA data are quite large (each gzip-compressed VNIR spectra record is about 6.6kb), so requests for these data are disabled by default. Note that VNIR spectra can only be queried by soil series or geographic BBOX.

Value

`pedons`: a `SoilProfileCollection` object containing site/pedon/horizon data

`trees`: a `data.frame` object containing tree DBH and height

`veg`: a `data.frame` object containing plant species

`stock`: a `data.frame` object containing carbon quantities (stocks) at standardized depths

`sample`: a `data.frame` object containing sample-level bulk density and soil organic carbon values

`spectra`: a numeric `matrix` containing VNIR reflectance spectra from 350–2500 nm

Author(s)

D.E. Beaudette, USDA-NRCS staff

References

https://www.nrcs.usda.gov/wps/portal/nrcs/detail/soils/survey/?cid=nrcs142p2_054164

See Also

`fetchOSD`

`fetchSCAN` *Get Daily Climate Data from USDA-NRCS SCAN (Soil Climate Analysis Network) Stations*

Description

Query soil/climate data from USDA-NRCS SCAN Stations

Usage

```
fetchSCAN(
  site.code = NULL,
  year = NULL,
  report = "SCAN",
  timeseries = c("Daily", "Hourly"),
  ...
)

SCAN_sensor_metadata(site.code)

SCAN_site_metadata(site.code = NULL)
```

Arguments

`site.code`	a vector of site codes. If `NULL` `SCAN_site_metadata()` returns metadata for all SCAN sites.
`year`	a vector of years
`report`	report name, single value only; default `'SCAN'`, other example options include individual sensor codes, e.g. `'SMS'` for Soil Moisture Storage, `'TEMP'` for temperature

`timeseries`	either `'Daily'` or `'Hourly'`
`...`	additional arguments. May include `intervalType`, `format`, `sitenum`, `interval`, `year`, `month`. Presence of additional arguments bypasses default batching functionality provided in the function and submits a 'raw' request to the API form.

Details

Possible above and below ground sensor types include: 'SMS' (soil moisture), 'STO' (soil temperature), 'SAL' (salinity), 'TAVG' (daily average air temperature), 'TMIN' (daily minimum air temperature), 'TMAX' (daily maximum air temperature), 'PRCP' (daily precipitation), 'PREC' (daily precipitation), 'SNWD' (snow depth), 'WTEQ' (snow water equivalent),'WDIRV' (wind direction), 'WSPDV' (wind speed), 'LRADT' (solar radiation/langley total).

SCAN Sensors:

All Soil Climate Analysis Network (SCAN) sensor measurements are reported hourly.

Element Measured	Sensor Type
Air Temperature	Shielded thermistor
Barometric Pressure	Silicon capacitive pressure sensor
Precipitation	Storage-type gage or tipping bucket
Relative Humidity	Thin film capacitance-type sensor
Snow Depth	Sonic sensor (not on all stations)
Snow Water Content	Snow pillow device and a pressure transducer (not on all stations)
Soil Moisture	Dielectric constant measuring device. Typical measurements are at 2", 4", 8", 20", and 40" where
Soil Temperature	Encapsulated thermistor. Typical measurements are at 2", 4", 8", 20", and 40" where possible.
Solar Radiation	Pyranometer
Wind Speed and Direction	Propellor-type anemometer

SNOTEL Sensors:

All Snow Telemetry (SNOTEL) sensor measurements are reported daily.

Element Measured	Sensor Type
Air Temperature	Shielded thermistor
Barometric Pressure	Silicon capacitive pressure sensor
Precipitation	Storage-type gage or tipping bucket
Relative Humidity	Thin film capacitance-type sensor
Snow Depth	Sonic sensor
Snow Water Content	Snow pillow device and a pressure transducer
Soil Moisture	Dielectric constant measuring device. Typical measurements are at 2", 4", 8", 20", and 40" where
Soil Temperature	Encapsulated thermistor. Typical measurements are at 2", 4", 8", 20", and 40" where possible.
Solar Radiation	Pyranometer
Wind Speed and Direction	Propellor-type anemometer

See the fetchSCAN tutorial for additional usage and visualization examples.

Value

a `list` of `data.frame` objects, where each element name is a sensor type, plus a `metadata` table; different `report` types change the types of sensor data returned. `SCAN_sensor_metadata()` and `SCAN_site_metadata()` return a `data.frame`. `NULL` on bad request.

Author(s)

D.E. Beaudette, A.G. Brown

References

See the National Water and Climate Center home page for more information on the SCAN and SNOTEL programs, information on web services, and interactive maps of snow water equivalent, precipitation and streamflow.

Examples

```
if(requireNamespace("curl") &
    curl::has_internet()) {

    # get data
    x <- try(fetchSCAN(site.code=c(356, 2072), year=c(2015, 2016)))
    str(x)

    # get sensor metadata
    m <- SCAN_sensor_metadata(site.code=c(356, 2072))

    # get site metadata
    m <- SCAN_site_metadata(site.code=c(356, 2072))

    # get hourly data (396315 records)
   # x <- try(fetchSCAN(site.code=c(356, 2072), year=c(2015, 2016), timeseries = "Hourly"))
}
```

`fetchSDA_spatial` *Get Spatial Data from Soil Data Access by* `mukey`, `nationalmusym` *or* `areasymbol`

Description

This method facilitates queries to Soil Data Access (SDA) mapunit and survey area geometry. Queries are generated based on map unit key (`mukey`) and national map unit symbol (`nationalmusym`) for `mupolygon` (SSURGO) or `gsmmupolygon` (STATSGO) geometry OR legend key (`lkey`) and area symbols (`areasymbol`) for `sapolygon` (Soil Survey Area; SSA) geometry).

A Soil Data Access query returns geometry and key identifying information about the map unit or area of interest. Additional columns from the map unit or legend table can be included; see `add.fields` argument.

Usage

```
fetchSDA_spatial(
  x,
  by.col = "mukey",
  method = "feature",
  geom.src = "mupolygon",
  db = "SSURGO",
  add.fields = NULL,
  chunk.size = 10,
  verbose = TRUE,
  as_Spatial = getOption("soilDB.return_Spatial", default = FALSE)
)
```

Arguments

x	A vector of map unit keys (mukey) or national map unit symbols (nmusym) for mupolygon geometry OR legend keys (lkey) or soil survey area symbols (areasymbol) for sapolygon geometry
by.col	Column name containing map unit identifier "mukey", "nmusym"/"nationalmusym" for geom.src mupolygon OR "areasymbol", "areaname", "mlraoffice", "mouagncyresp" for geom.src sapolygon; default is determined by is.numeric(x) TRUE for mukey or lkey and nationalmusym or areasymbol otherwise.
method	geometry result type: "feature" returns polygons, "bbox" returns the bounding box of each polygon (via STEnvelope()), and "point" returns a single point (via STPointOnSurface()) within each polygon.
geom.src	Either mupolygon (map unit polygons) or sapolygon (soil survey area boundary polygons)
db	Default: "SSURGO". When geom.src is mupolygon, use STATSGO polygon geometry instead of SSURGO by setting db = "STATSGO"
add.fields	Column names from mapunit or legend table to add to result. Must specify parent table name as the prefix before column name e.g. mapunit.muname.
chunk.size	Number of values of x to process per query. Necessary for large results. Default: 10
verbose	Print messages?
as_Spatial	Return sp classes? e.g. Spatial*DataFrame. Default: FALSE.

Details

This function automatically "chunks" the input vector (using makeChunks()) of map unit identifiers to minimize the likelihood of exceeding the SDA data request size. The number of chunks varies with the chunk.size setting and the length of your input vector. If you are working with many map units and/or large extents, you may need to decrease this number in order to have more chunks.

Querying regions with complex mapping may require smaller chunk.size. Numerically adjacent IDs in the input vector may share common qualities (say, all from same soil survey area or region) which could cause specific chunks to perform "poorly" (slow or error) no matter what the chunk size is. Shuffling the order of the inputs using sample() may help to eliminate problems related

to this, depending on how you obtained your set of MUKEY/nationalmusym to query. One could feasibly use `muacres` as a heuristic to adjust for total acreage within chunks.

Note that STATSGO data are fetched where `CLIPAREASYMBOL = 'US'` to avoid duplicating state and national subsets of the geometry.

Value

an `sf` data.frame corresponding to SDA spatial data for all symbols requested. If `as_Spatial=TRUE` returns a `Spatial*DataFrame` from the sp package via `sf::as_Spatial()` for backward compatibility. Default result contains geometry with attribute table containing unique feature ID, symbol and area symbol plus additional fields in result specified with `add.fields`.

Author(s)

Andrew G. Brown, Dylan E. Beaudette

Examples

```
if(requireNamespace("curl") &
   curl::has_internet()) {

    # get spatial data for a single mukey
    single.mukey <- try(fetchSDA_spatial(x = "2924882"))

    # demonstrate fetching full extent (multi-mukey) of national musym
    full.extent.nmusym <- try(fetchSDA_spatial(x = "2x815", by = "nmusym"))

    # compare extent of nmusym to single mukey within it
    if (!inherits(single.mukey, 'try-error') &&
        !inherits(full.extent.nmusym, 'try-error')) {

        if (requireNamespace("sf")) {

         plot(sf::st_geometry(full.extent.nmusym), col = "RED", border = 0)
         plot(sf::st_geometry(single.mukey), add = TRUE, col = "BLUE", border = 0)

        }

    }

    # demo adding a field (`muname`) to attribute table of result
    head(try(fetchSDA_spatial(x = "2x815", by="nmusym", add.fields="muname")))
}
```

`fetchSoilGrids` *Get SoilGrids 250m properties information from point locations*

Description

This function obtains SoilGrids properties information (250m raster resolution) given a `data.frame` containing site IDs, latitudes and longitudes. SoilGrids API and maps return values as whole (integer) numbers to minimize the storage space used. These values are converted by to produce conventional units by 'fetchSoilGrids()"

Usage

```
fetchSoilGrids(
  x,
  loc.names = c("id", "lat", "lon"),
  verbose = FALSE,
  progress = FALSE
)
```

Arguments

`x`	A `data.frame` containing 3 columns referring to site ID, latitude and longitude.
`loc.names`	Optional: Column names referring to site ID, latitude and longitude. Default: `c("id","lat","lon")`
`verbose`	Print messages? Default: `FALSE`
`progress`	logical, give progress when iterating over multiple requests; Default: `FALSE`

Details

Properties:

Name	Description	Mapped units	Co
bdod	Bulk density of the fine earth fraction	cg/cm^3	
cec	Cation Exchange Capacity of the soil	mmol(c)/kg	
cfvo	Volumetric fraction of coarse fragments (> 2 mm)	cm^3/dm^3 (vol per mil)	
clay	Proportion of clay particles (< 0.002 mm) in the fine earth fraction	g/kg	
nitrogen	Total nitrogen (N)	cg/kg	
phh2o	Soil pH	pH*10	
sand	Proportion of sand particles (> 0.05 mm) in the fine earth fraction	g/kg	
silt	Proportion of silt particles (= 0.002 mm and = 0.05 mm) in the fine earth fraction	g/kg	
soc	Soil organic carbon content in the fine earth fraction	dg/kg	
ocd	Organic carbon density	hg/m^3	
ocs	Organic carbon stocks	t/ha	

SoilGrids predictions are made for the six standard depth intervals specified in the GlobalSoilMap

IUSS working group and its specifications. The depth intervals returned are: `"0-5cm"`, `"5-15cm"`, `"15-30cm"`, `"30-60cm"`, `"60-100cm"`, `"100-200cm"` and the properties returned are `"bdod"`, `"cec"`, `"cfvo"`, `"clay"`, `"nitrogen"`, `"phh2o"`, `"sand"`, `"silt"`, `"soc"` – each with 5th, 50th, 95th, mean and uncertainty values. The uncertainty values are the ratio between the inter-quantile range (90% prediction interval width) and the median : `(Q0.95-Q0.05)/Q0.50`. Point data requests are made through `properties/query` endpoint of the SoilGrids v2.0 REST API. Please check ISRIC's data policy, disclaimer and citation: https://www.isric.org/about/data-policy.

Find out more information about the SoilGrids and GlobalSoilMap products here:

- https://www.isric.org/explore/soilgrids/faq-soilgrids
- https://www.isric.org/sites/default/files/GlobalSoilMap_specifications_december_2015_2.pdf

Value

A SoilProfileCollection

Author(s)

Andrew G. Brown

References

Poggio, L., de Sousa, L. M., Batjes, N. H., Heuvelink, G. B. M., Kempen, B., Ribeiro, E., and Rossiter, D.: SoilGrids 2.0: producing soil information for the globe with quantified spatial uncertainty, SOIL, 7, 217-240, 2021. doi:10.5194/soil72172021

Examples

```
## Not run:
 if(requireNamespace("curl") &
   curl::has_internet()) {

  library(aqp)

  your.points <- data.frame(id  = c("A", "B"),
                            lat = c(37.9, 38.1),
                            lon = c(-120.3, -121.5),
                            stringsAsFactors = FALSE)

  x <- try(fetchSoilGrids(your.points))

  if (!inherits(x, 'try-error'))
   plotSPC(x, name = NA, color = "socQ50")
 }

## End(Not run)
```

fetchVegdata *Get vegetation plot data from local NASIS database*

Description

Get vegetation plot data from local NASIS database

Usage

```
fetchVegdata(SS = TRUE, stringsAsFactors = NULL, dsn = NULL)

get_vegplot_from_NASIS_db(SS = TRUE, stringsAsFactors = NULL, dsn = NULL)

get_vegplot_location_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_vegplot_trhi_from_NASIS_db(SS = TRUE, stringsAsFactors = NULL, dsn = NULL)

get_vegplot_species_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_vegplot_transect_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_vegplot_transpecies_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_vegplot_tree_si_summary_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_vegplot_tree_si_details_from_NASIS_db(
  SS = TRUE,
```

```
  stringsAsFactors = NULL,
  dsn = NULL
)

get_vegplot_textnote_from_NASIS_db(
  SS = TRUE,
  fixLineEndings = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)
```

Arguments

SS	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: TRUE)
stringsAsFactors	deprecated
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL
fixLineEndings	Replace '\r\n' with '\n'; Default: TRUE

Value

A named list containing: "vegplot", "vegplotlocation", "vegplotrhi", "vegplotspecies", "vegtransect", "vegtransplantsum", 'vegsiteindexsum', "vegsiteindexdet", and "vegplottext" tables

---

filter_geochem *Filter KSSL Geochemical Table*

Description

A function to subset KSSL "geochem" / elemental analysis result table to obtain rows/columns based on: column name, preparation code, major / trace element method.

Usage

```
filter_geochem(
  geochem,
  columns = NULL,
  prep_code = NULL,
  major_element_method = NULL,
  trace_element_method = NULL
)
```

Arguments

geochem	geochemical data, as returned by fetchKSSL
columns	Column name(s) to include in result
prep_code	Character vector of prep code(s) to include in result.
major_element_method	Character vector of major element method(s) to include in result.
trace_element_method	Character vector of trace element method(s) to include in result.

Value

A data.frame, subset according to the constraints specified in arguments.

Author(s)

Andrew G. Brown.

---

format_SQL_in_statement

Format vector of values into a string suitable for an SQL IN *statement.*

Description

Concatenate a vector to SQL IN-compatible syntax: `letters[1:3]` becomes `('a','b','c')`. Values in x are first passed through `unique()`.

Usage

```
format_SQL_in_statement(x)
```

Arguments

x	A character vector.

Value

A character vector (unit length) containing concatenated group syntax for use in SQL IN, with unique value found in x.

Note

Only `character` output is supported.

Examples

```
library(aqp)

# get some mukeys
q <- "select top(2) mukey from mapunit;"
mukeys <- SDA_query(q)

# format for use in an SQL IN statement
mukey.inst <- format_SQL_in_statement(mukeys$mukey)
mukey.inst

# make a more specific query: for component+horizon data, just for those mukeys
q2 <- sprintf("SELECT * FROM mapunit
               INNER JOIN component ON mapunit.mukey = component.mukey
               INNER JOIN chorizon ON component.cokey = chorizon.cokey
               WHERE mapunit.mukey IN %s;", mukey.inst)
# do the query
res <- SDA_query(q2)

# build a SoilProfileCollection from horizon-level records
depths(res) <- cokey ~ hzdept_r + hzdepb_r

# normalize mapunit/component level attributes to site-level for plot
site(res) <- ~ muname + mukey + compname + comppct_r + taxclname

# make a nice label
res$labelname <- sprintf("%s (%s%s)", res$compname, res$comppct_r, "%")

# major components only
res <- subset(res, comppct_r >= 85)

# inspect plot of result
par(mar=c(0,0,0,0))
groupedProfilePlot(res, groups = "mukey", color = "hzname", cex.names=0.8,
                   id.style = "side", label = "labelname")
```

`getHzErrorsNASIS` *Get Logic Errors in NASIS/PedonPC Pedon Horizon*

Description

Get Logic Errors in NASIS/PedonPC Pedon Horizon

Usage

```
getHzErrorsNASIS(strict = TRUE, SS = TRUE, dsn = NULL)
```

Arguments

strict	how strict should horizon boundaries be checked for consistency: TRUE=more | FALSE=less
SS	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: TRUE)
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Value

A data.frame containing problematic records with columns: 'peiid','pedon_id','hzdept','hzdepb','hzname'

---

get_colors_from_NASIS_db

Get Soil Color Data from a local NASIS Database

Description

Get, format, mix, and return color data from a NASIS database.

Usage

```
get_colors_from_NASIS_db(SS = TRUE, mixColors = TRUE, dsn = NULL)
```

Arguments

SS	fetch data from Selected Set in NASIS or from the entire local database (default: TRUE)
mixColors	should mixed colors be calculated (Default: TRUE) where multiple colors are populated for the same moisture state in a horizon? FALSE takes the dominant color based on colorpct or first record based on horizon ID (phiid) sorting for "moist" and "dry" state. Pedon Horizon Color records without a moisture state populated are ignored.
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Value

A data.frame with the results.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

simplifyColorData, get_hz_data_from_NASIS_db, get_site_data_from_NASIS_db

---

`get_colors_from_pedon_db` *Get Soil Color Data from a PedonPC Database*

Description

Get, format, mix, and return color data from a PedonPC database.

Usage

```
get_colors_from_pedon_db(dsn)
```

Arguments

`dsn`	The path to a 'pedon.mdb' database.

Value

A data.frame with the results.

Author(s)

Dylan E. Beaudette and Jay M. Skovlin

See Also

`get_hz_data_from_pedon_db`, `get_site_data_from_pedon_db`

`get_comonth_from_NASIS_db` *Get component month data from a local NASIS Database*

Description

Get component month data from a local NASIS Database.

Usage

```
get_comonth_from_NASIS_db(
  SS = TRUE,
  fill = FALSE,
  stringsAsFactors = NULL,
  dsn = NULL
)
```

Arguments

SS	get data from the currently loaded Selected Set in NASIS or from the entire local database (default: TRUE)
fill	should missing "month" rows in the comonth table be filled with NA (FALSE)
stringsAsFactors	deprecated
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Value

A list with the results.

Author(s)

Stephen Roecker

See Also

fetchNASIS

Examples

```
if(local_NASIS_defined()) {
  # query text note data
  cm <- try(get_comonth_from_NASIS_db())

  # show structure of component month data
  str(cm)
}
```

---

get_component_data_from_NASIS_db

Get component data from a local NASIS Database

Description

Get component data from a local NASIS Database

Usage

```
get_component_data_from_NASIS_db(
  SS = TRUE,
  nullFragsAreZero = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_component_diaghz_from_NASIS_db(SS = TRUE, dsn = NULL)

get_component_restrictions_from_NASIS_db(SS = TRUE, dsn = NULL)

get_component_correlation_data_from_NASIS_db(
  SS = TRUE,
  dropAdditional = TRUE,
  dropNotRepresentative = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_component_cogeomorph_data_from_NASIS_db(SS = TRUE, dsn = NULL)

get_component_cogeomorph_data_from_NASIS_db2(SS = TRUE, dsn = NULL)

get_component_copm_data_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_component_esd_data_from_NASIS_db(
  SS = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_component_otherveg_data_from_NASIS_db(SS = TRUE, dsn = NULL)

get_copedon_from_NASIS_db(SS = TRUE, dsn = NULL)

get_component_horizon_data_from_NASIS_db(
  SS = TRUE,
  fill = FALSE,
  dsn = NULL,
  nullFragsAreZero = TRUE
)
```

Arguments

SS	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: TRUE)
nullFragsAreZero	should surface fragment cover percentages of NULL be interpreted as 0? (default: TRUE)
stringsAsFactors	deprecated
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL
dropAdditional	Remove map units with "additional" status? Default: TRUE
dropNotRepresentative	Remove non-representative data map units? Default: TRUE
fill	Return a single minimal (NA-filled) horizon for components with no horizon records? Default FALSE

Value

a data.frame

Author(s)

Dylan E. Beaudette, Stephen Roecker, and Jay M. Skovlin

See Also

fetchNASIS

Examples

```
if(local_NASIS_defined()) {
 # query text note data
 fc <- try(get_component_data_from_NASIS_db())

 # show structure of component data returned
 str(fc)
}
```

get_component_from_GDB *Get a SoilProfileCollection from a SSURGO file geodatabase*

Description

Functions to load and flatten commonly used tables and from SSURGO file geodatabases, and create soil profile collection objects (SPC).

Usage

```
get_component_from_GDB(
  dsn = "gNATSGO_CONUS.gdb",
  WHERE = NULL,
  childs = FALSE,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

get_legend_from_GDB(
  dsn = "gNATSGO_CONUS.gdb",
  WHERE = NULL,
  droplevels = TRUE,
  stringsAsFactors = NULL,
  stats = FALSE
)

get_mapunit_from_GDB(
  dsn = "gNATSGO_CONUS.gdb",
  WHERE = NULL,
  droplevels = TRUE,
  stringsAsFactors = NULL,
  stats = FALSE
)

fetchGDB(
  dsn = "gNATSGO_CONUS.gdb",
  WHERE = NULL,
  childs = TRUE,
  droplevels = TRUE,
  stringsAsFactors = NULL
)
```

Arguments

dsn	data source name (interpretation varies by driver - for some drivers, dsn is a file name, but may also be a folder, or contain the name and access credentials of

	a database); in case of GeoJSON, dsn may be the character string holding the geojson data. It can also be an open database connection.
WHERE	text string formatted as an SQL WHERE clause (default: FALSE)
childs	logical; if FALSE parent material and geomorphic child tables are not flattened and appended
droplevels	logical: indicating whether to drop unused levels in classifying factors. This is useful when a class has large number of unused classes, which can waste space in tables and figures.
stringsAsFactors	deprecated
stats	Return statistics (number of mapunit keys per legend; number of components, major components per mapunit, total and hydric component percentage)? Default: FALSE

Details

These functions return data from SSURGO file geodatabases with the use of a simple text string that formatted as an SQL WHERE clause (e.g. `WHERE = "areasymbol = 'IN001'"`. Any columns within the target table can be specified (except for fetchGDB() currently, which only targets the legend with the WHERE clause).

Value

A `data.frame` or `SoilProfileCollection` object.

Author(s)

Stephen Roecker

Examples

```
## replace `dsn` with path to your own geodatabase (SSURGO OR gNATSGO)
##
##
##  download CONUS gNATSGO from here:
##   https://www.nrcs.usda.gov/wps/portal/nrcs/detail/soils/survey/geo/?cid=nrcseprd1464625
##
##
# dsn <- "D:/geodata/soils/gNATSGO_CONUS.gdb"

# le <- get_legend_from_GDB(dsn = dsn, WHERE = "areasymbol LIKE '%'")

# mu <- get_mapunit_from_GDB(dsn = dsn, WHERE = "muname LIKE 'Miami%'")

# co <- get_component_from_GDB(dsn, WHERE = "compname = 'Miami'
#                              AND majcompflag = 'Yes'", childs = FALSE)
```

```
# f_in_GDB <- fetchGDB(WHERE = "areasymbol LIKE 'IN%'")
```

`get_component_from_SDA` *Get SSURGO/STATSGO2 Mapunit Data from Soil Data Access*

Description

Functions to download and flatten commonly used tables and from Soil Data Access, and create soil profile collection objects (SPC).

Usage

```
get_component_from_SDA(
  WHERE = NULL,
  duplicates = FALSE,
  childs = TRUE,
  droplevels = TRUE,
  nullFragsAreZero = TRUE,
  stringsAsFactors = NULL
)

get_cointerp_from_SDA(
  WHERE = NULL,
  mrulename = NULL,
  duplicates = FALSE,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

get_legend_from_SDA(WHERE = NULL, droplevels = TRUE, stringsAsFactors = NULL)

get_lmuaoverlap_from_SDA(
  WHERE = NULL,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

get_mapunit_from_SDA(WHERE = NULL, droplevels = TRUE, stringsAsFactors = NULL)

get_chorizon_from_SDA(
  WHERE = NULL,
  duplicates = FALSE,
  childs = TRUE,
```

```
  nullFragsAreZero = TRUE,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

fetchSDA(
  WHERE = NULL,
  duplicates = FALSE,
  childs = TRUE,
  nullFragsAreZero = TRUE,
  rmHzErrors = FALSE,
  droplevels = TRUE,
  stringsAsFactors = NULL
)

get_cosoilmoist_from_SDA(
  WHERE = NULL,
  duplicates = FALSE,
  impute = TRUE,
  stringsAsFactors = NULL
)
```

Arguments

WHERE	text string formatted as an SQL WHERE clause (default: FALSE)
duplicates	logical; if TRUE a record is returned for each unique mukey (may be many per nationalmusym)
childs	logical; if FALSE parent material and geomorphic child tables are not flattened and appended
droplevels	logical: indicating whether to drop unused levels in classifying factors. This is useful when a class has large number of unused classes, which can waste space in tables and figures.
nullFragsAreZero	should fragment volumes of NULL be interpreted as 0? (default: TRUE), see details
stringsAsFactors	deprecated
mrulename	character. Interpretation rule names
rmHzErrors	should pedons with horizonation errors be removed from the results? (default: FALSE)
impute	replace missing (i.e. NULL) values with "Not_Populated" forcategorical data, or the "RV" for numeric data or 201 cm if the "RV" is alsoNULL (default: TRUE)

Details

These functions return data from Soil Data Access with the use of a simple text string that formatted as an SQL WHERE clause (e.g. WHERE = "areasymbol = 'IN001'". All functions are SQL queries that wrap around SDAquery() and format the data for analysis.

Beware SDA includes the data for both SSURGO and STATSGO2. The `areasymbol` for STATSGO2 is `US`. For just SSURGO, include `WHERE = "areareasymbol != 'US'"`.

If the duplicates argument is set to TRUE, duplicate components are returned. This is not necessary with data returned from NASIS, which has one unique national map unit. SDA has duplicate map national map units, one for each legend it exists in.

The value of `nullFragsAreZero` will have a significant impact on the rock fragment fractions returned by `fetchSDA`. Set `nullFragsAreZero = FALSE` in those cases where there are many data-gaps and NULL rock fragment values should be interpreted as NULLs. Set `nullFragsAreZero = TRUE` in those cases where NULL rock fragment values should be interpreted as 0.

Additional examples can be found in the Soil Data Access (SDA) Tutorial

Value

A data.frame or SoilProfileCollection object.

Author(s)

Stephen Roecker

See Also

SDA_query

`get_cosoilmoist_from_NASIS`

Get the Component Soil Moisture Tables

Description

Read and flatten the component soil moisture month tables from a local NASIS Database.

Usage

```
get_cosoilmoist_from_NASIS(
  SS = TRUE,
  impute = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)
```

Arguments

`SS`	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: `TRUE`)
`impute`	replace missing (i.e. `NULL`) values with `"Not_Populated"` for categorical data, or the "RV" for numeric data or `201` cm if the "RV" is also `NULL` (default: `TRUE`)

`stringsAsFactors`
: deprecated

`dsn`
: Optional: path to local SQLite database containing NASIS table structure; default: `NULL`

Details

The component soil moisture tables within NASIS house monthly data on flooding, ponding, and soil moisture status. The soil moisture status is used to specify the water table depth for components (e.g. `status == "Moist"`).

Value

A data.frame.

Author(s)

S.M. Roecker

See Also

fetchNASIS, get_cosoilmoist_from_NASISWebReport, get_cosoilmoist_from_SDA, `get_comonth_from_SDA`

Examples

```
if(local_NASIS_defined()) {
 # load cosoilmoist (e.g. water table data)
 test <- try(get_cosoilmoist_from_NASIS())

 # inspect
 if(!inherits(test, 'try-error')) {
   head(test)
 }
}
```

---

get_EDIT_ecoclass_by_geoUnit

Get Ecological Dynamics Information Tool (EDIT) ecological sites by catalog (ESD/ESG) and MLRA

Description

Data are accessed via Ecological Dynamics Interpretive Tool (EDIT) web services: https://edit.jornada.nmsu.edu/resources/es
`geoUnit` refers to MLRA codes, possibly with a leading zero and trailing "X" for two digit MLRA symbols.

Usage

```
get_EDIT_ecoclass_by_geoUnit(geoUnit, catalog = c("esd", "esg"))
```

Arguments

geoUnit	A character vector of geoUnit codes e.g. c("018X","022A") for MLRAs 18 and 22A.
catalog	Catalog ID. One of: "esd" or "esg"

Value

A data.frame containing: geoUnit, id, legacyId, name. NULL if no result.

Examples

```
if(requireNamespace("curl") &
  curl::has_internet()) {
  get_EDIT_ecoclass_by_geoUnit(c("018X","022A"))
}
```

---

get_extended_data_from_NASIS_db

Get accessory tables and summaries from a local NASIS Database

---

Description

Get accessory tables and summaries from a local NASIS Database

Usage

```
get_extended_data_from_NASIS_db(
  SS = TRUE,
  nullFragsAreZero = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)
```

Arguments

SS	get data from the currently loaded Selected Set in NASIS or from the entire local database (default: TRUE)
nullFragsAreZero	should fragment volumes of NULL be interpreted as 0? (default: TRUE), see details

`stringsAsFactors`
: deprecated

`dsn`
: Optional: path to local SQLite database containing NASIS table structure; default: `NULL`

Value

A list with the results.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

`get_hz_data_from_NASIS_db`, `get_site_data_from_NASIS_db`

Examples

```
if(local_NASIS_defined()) {
 # query extended data
 e <- try(get_extended_data_from_NASIS_db())

 # show contents of extended data
 str(e)
}
```

---

get_extended_data_from_pedon_db

Get accessory tables and summaries from a local pedonPC Database

---

Description

Get accessory tables and summaries from a local pedonPC Database.

Usage

```
get_extended_data_from_pedon_db(dsn)
```

Arguments

`dsn`
: The path to a 'pedon.mdb' database.

Value

A list with the results.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

get_hz_data_from_pedon_db, get_site_data_from_pedon_db

---

get_hz_data_from_NASIS_db

Get Horizon Data from a local NASIS Database

Description

Get horizon-level data from a local NASIS database.

Usage

```
get_hz_data_from_NASIS_db(
  SS = TRUE,
  fill = FALSE,
  stringsAsFactors = NULL,
  dsn = NULL
)
```

Arguments

SS	fetch data from Selected Set in NASIS or from the entire local database (default: TRUE)
fill	include pedons without horizon data in result? default: FALSE
stringsAsFactors	deprecated
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Value

A data.frame.

Note

NULL total rock fragment values are assumed to represent an *absence* of rock fragments, and set to 0.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

get_hz_data_from_NASIS_db, get_site_data_from_NASIS_db

---

get_hz_data_from_pedon_db

Get Horizon Data from a PedonPC Database

Description

Get horizon-level data from a PedonPC database.

Usage

```
get_hz_data_from_pedon_db(dsn)
```

Arguments

dsn	The path to a 'pedon.mdb' database.

Value

A data.frame.

Note

NULL total rock fragment values are assumed to represent an *absence* of rock fragments, and set to 0.

Author(s)

Dylan E. Beaudette and Jay M. Skovlin

See Also

get_colors_from_pedon_db, get_site_data_from_pedon_db

`get_lablayer_data_from_NASIS_db` *Get lab pedon layer data from a local NASIS Database*

Description

Get lab pedon layer-level (horizon-level) data from a local NASIS database.

Usage

```
get_lablayer_data_from_NASIS_db(SS = TRUE, dsn = NULL)
```

Arguments

`SS`	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: `TRUE`)
`dsn`	Optional: path to local SQLite database containing NASIS table structure; default: `NULL`

Value

A data.frame.

Note

This function queries KSSL laboratory site/horizon data from a local NASIS database from the lab layer data table.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

`get_labpedon_data_from_NASIS_db`

`get_labpedon_data_from_NASIS_db` *Get lab pedon data from a local NASIS Database*

Description

Get lab pedon-level data from a local NASIS database.

Usage

```
get_labpedon_data_from_NASIS_db(SS = TRUE, dsn = NULL)
```

Arguments

`SS`	fetch data from the currently loaded selected set in NASIS or from the entire local database (default: TRUE)
`dsn`	Optional: path to local SQLite database containing NASIS table structure; default: `NULL`

Details

This function currently works only on Windows, and requires a 'nasis_local' ODBC connection.

Value

A data.frame.

Note

This function queries KSSL laboratory site/horizon data from a local NASIS database from the lab pedon data table.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

`get_lablayer_data_from_NASIS_db`

get_mapunit_from_NASIS *Get Legend, Mapunit and Legend Mapunit Area Overlap Tables*

Description

Get Legend, Mapunit and Legend Mapunit Area Overlap Tables

Usage

```
get_mapunit_from_NASIS(
  SS = TRUE,
  repdmu = TRUE,
  droplevels = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_legend_from_NASIS(
  SS = TRUE,
  droplevels = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_lmuaoverlap_from_NASIS(
  SS = TRUE,
  droplevels = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)

get_projectmapunit_from_NASIS(SS = TRUE, stringsAsFactors = NULL, dsn = NULL)
```

Arguments

SS	Fetch data from the currently loaded selected set in NASIS or from the entire local database (default: TRUE)
repdmu	Return only "representative" data mapunits? Default: TRUE
droplevels	Drop unused levels from farmlndcl and other factor levels from NASIS domains?
stringsAsFactors	deprecated
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

get_NASIS_metadata *Get NASIS Metadata (Domain, Column and Choice Lists)*

Description

Retrieve a table containing domain and column names with choice list labels/names/sequences/values from the NASIS 7 metadata tables.

Usage

```
get_NASIS_metadata(dsn = NULL)

get_NASIS_column_metadata(x, what = "ColumnPhysicalName", dsn = NULL)
```

Arguments

dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL
x	character vector to match in NASIS metadata
what	Column to match x against. Default "ColumnPhysicalName"; alternate options include "DomainID", "DomainName", "DomainRanked", "DisplayLabel", "ChoiceSequence", "ChoiceValue", "ChoiceName", "ChoiceLabel", "ChoiceObsolete", "ChoiceDescription", "ColumnLogicalName"

Details

These data are derived from the MetadataDomainDetail, MetadataDomainMaster, and MetadataTableColumn tables and help with mapping between values stored in the NASIS database and human-readable values. The human-readable values align with the values returned in public facing interfaces such as SSURGO via Soil Data Access and NASIS Web Reports. The data in these tables can also be used to create *ordered* factors where options for levels of a particular data element follow a logical ChoiceSequence.

If a local NASIS instance is set up, and this is the first time get_NASIS_metadata() has been called, the metadata will be obtained from the NASIS local database. Subsequent runs in the same session will use a copy of the data object NASIS.metadata cached in soilDB.env.

For users without a local NASIS instance, a cached copy of the NASIS metadata are used (data/metadata.rda).

See ?soilDB::metadata for additional details.

Value

a data.frame containing DomainID, DomainName, DomainRanked, DisplayLabel, ChoiceSequence, ChoiceValue, ChoiceName, ChoiceLabel, ChoiceObsolete, ColumnPhysicalName, ColumnLogicalName

a data.frame containing selected NASIS metadata sorted first on DomainID and then on ChoiceSequence

Examples

```
get_NASIS_metadata()
get_NASIS_column_metadata("texcl")
```

`get_NASIS_table_key_by_name` *Get a NASIS table key by type and table name*

Description

Get a NASIS table key by type and table name

Usage

```
get_NASIS_table_key_by_name(
  tables,
  keycol = c("all", "fkey", "pkeyref", "pkey")
)
```

Arguments

`tables`	character vector of table names
`keycol`	One of: "fkey" the foreign key; "pkeyref" the primary key referenced by the foreign key, or "pkey" the primary key.

Value

The key column name for the specified table name

Examples

```
## Not run:
get_NASIS_table_key_by_name(c("site","phorizon_View_1","not_a_table"))

## End(Not run)#'
```

get_NASIS_table_name_by_purpose *Get NASIS 7 Physical Table Names*

Description

Method generalizing concepts of NASIS 7 data model to group tables by "purpose." Most of our more complex queries rely on tables from one or more purposes, so individual higher-level functions might call a function like this to identify the relevant tables from a data source.

Usage

```
get_NASIS_table_name_by_purpose(
 purpose = c("metadata", "lookup", "nasis", "site", "pedon", "transect", "component",
    "vegetation", "project", "techsoilservice", "area", "soilseries", "legend",
    "mapunit", "datamapunit"),
  SS = FALSE
)
```

Arguments

purpose	character. One or more of: "metadata", "lookup", "nasis", "site", "pedon", "transect", "component", "vegetation", "project", "techsoilservice", "area", "soilseries", "legend", "mapunit", "datamapunit"
SS	append "_View_1" on appropriate tables? Default: FALSE

Value

character vector of table names

See Also

createStaticNASIS

Examples

```
## Not run:
# get the "site" table names
get_NASIS_table_name_by_purpose("site")

# get the pedon table names
get_NASIS_table_name_by_purpose("pedon", SS = TRUE)

# metadata and lookup not affected by SS argument, but site and pedon are
get_NASIS_table_name_by_purpose(c("metadata", "lookup",
                                  "site", "pedon"), SS = TRUE)

## End(Not run)
```

get_NOAA_GHCND *Get Global Historical Climatology Network Daily (GHCND) data from NOAA API*

Description

Obtain daily climatic summary data for a set of station IDs, years, and datatypes.

Note that typically results from the NOAA API are limited to 1000 records. However, by "chunking" up data into individual station*year*datatypeid combinations, record results generally do not exceed 365 records for daily summaries.

In order to use this function, you must obtain an API token from this website: https://www.ncdc.noaa.gov/cdo-web/token

Usage

```
get_NOAA_GHCND(stations, years, datatypeids, apitoken)
```

Arguments

stations	Station ID (e.g. GHCND:USC00388786)
years	One or more years (e.g. 2017:2020)
datatypeids	One or more NOAA GHCND data type IDs (e.g c("PRCP","SNOW"))
apitoken	API key token for NOAA NCDC web services (https://www.ncdc.noaa.gov/cdo-web/token)

Value

A data.frame containing the GHCND data requested (limit 1000 records)

Examples

```
#' ## in order to use this function, you must obtain an API token from this website:
##  https://www.ncdc.noaa.gov/cdo-web/token

# get_NOAA_GHCND(c("GHCND:USC00388786", "GHCND:USC00388787"),
#                years = 2017:2020,
#                datatypeids = c("PRCP","SNOW"),
#                apitoken = "yourtokenhere")
```

get_NOAA_stations_nearXY *Get NOAA station data near a given latitude and longitude*

Description

Query the NOAA API to get station data (limit 1000 records) near a point. Default extent is plus or minus 0.5 degrees (bounding box) (with bbox = 1) around the specified point [lat, lng].

In order to use this function, you must obtain an API token from this website: https://www.ncdc.noaa.gov/cdo-web/token

Usage

```
get_NOAA_stations_nearXY(lat, lng, apitoken, bbox = 1, crs = "EPSG:4326")
```

Arguments

lat	Latitude or Y coordinate in crs
lng	Longitude or X coordinate in crs
apitoken	API key token for NOAA NCDC web service
bbox	Optional: Dimension of the bounding box centered at lat, lng.
crs	Coordinate Reference System. Default "EPSG:4326"

Value

data.frame containing station information for all stations within a bounding box around lat, lng.

Examples

```
## in order to use this function, you must obtain an API token from this website:
##  https://www.ncdc.noaa.gov/cdo-web/token

# stations <- get_NOAA_stations_nearXY(lat = 37, lng = -120,
#                                      apitoken = "yourtokenhere")
```

get_OSD *Get Official Series Description Data from JSON, HTML or TXT sources*

Description

Get Official Series Description Data from JSON, HTML or TXT sources

Usage

```
get_OSD(
  series,
  base_url = NULL,
  result = c("json", "html", "txt"),
  fix_ocr_errors = FALSE,
  verbose = FALSE
)

get_OSD_JSON(series, base_url = NULL)
```

Arguments

series	A character vector of Official Series names e.g. "Chewacla"
base_url	Optional: alternate JSON/HTML/TXT repository path. Default: NULL uses "https://github.com/ncss-tech/SoilKnowledgeBase" for result="json"
result	Select "json", "html", or "txt" output
fix_ocr_errors	Default: FALSE; Applies only to result='json'. Convert clear cases of Optical Character Recognition (OCR) errors to likely actual values.
verbose	Print errors and warning messages related to HTTP requests? Default: FALSE

Details

The default base_url for result="json" is to JSON files stored in a GitHub repository that is regularly updated from the official source of Series Descriptions. Using format: https://raw.githubusercontent.com/ncss-t for JSON. And "https://soilseriesdesc.sc.egov.usda.gov/OSD_Docs/{LETTER}/{SERIES}.html is for result="html" (official source).

fix_ocr_errors by default is turned off (FALSE). When TRUE, assume that in color data hue/value/chroma lowercase "L" ("l") is a 1, and a capital "O" is interpreted as zero. Also, in horizon desgignations assume lowercase "L" is a 1, and a string that starts with 0 starts with the capital letter "O".

Value

For JSON result: A data.frame with 1 row per series, and 1 column per "section" in the OSD as defined in National Soil Survey Handbook. For TXT or HTML result a list of character vectors containing OSD text with 1 element per series and one value per line.

Examples

```
if(requireNamespace("curl") &
   curl::has_internet()) {

series <- c("Musick", "Hector", "Chewacla")
get_OSD(series)
}
```

`get_SDA_coecoclass` *Get mapunit ecological sites from Soil Data Access*

Description

Get mapunit ecological sites from Soil Data Access

Usage

```
get_SDA_coecoclass(
  method = "None",
  areasymbols = NULL,
  mukeys = NULL,
  WHERE = NULL,
  query_string = FALSE,
  ecoclasstypename = NULL,
  ecoclassref = "Ecological Site Description Database",
  not_rated_value = "Not assigned",
  miscellaneous_areas = TRUE,
  dsn = NULL
)
```

Arguments

`method`	aggregation method. One of: "Dominant Component", "Dominant Condition", "None". If "None" is selected one row will be returned per component, otherwise one row will be returned per map unit.
`areasymbols`	vector of soil survey area symbols
`mukeys`	vector of map unit keys
`WHERE`	character containing SQL WHERE clause specified in terms of fields in `legend`, `mapunit`, `component` or `coecosite` tables, used in lieu of `mukeys` or `areasymbols`
`query_string`	Default: `FALSE`; if `TRUE` return a character string containing query that would be sent to SDA via `SDA_query`
`ecoclasstypename`	If `NULL` no constraint on `ecoclasstypename` is used in the query.

ecoclassref Default: "Ecological Site Description Database". If NULL no constraint on ecoclassref is used in the query.

not_rated_value
Default: "Not assigned"

miscellaneous_areas
Include miscellaneous areas (non-soil components)?

dsn Path to local SQLite database or a DBIConnection object. If NULL (default) use Soil Data Access API via SDA_query().

Details

When method="Dominant Condition" an additional field ecoclasspct_r is returned in the result with the sum of comppct_r that have the dominant condition ecoclassid. The component with the greatest comppct_r is returned for the component and coecosite level information.

Note that if there are multiple coecoclasskey per ecoclassid there may be more than one record per component.

---

get_SDA_cosurfmorph *Get Geomorphic/Surface Morphometry Data from Soil Data Access*

Description

Get Geomorphic/Surface Morphometry Data from Soil Data Access or a local SSURGO data source and summarize by counts and proportions ("probabilities").

Usage

```
get_SDA_cosurfmorph(
  table = c("cosurfmorphgc", "cosurfmorphhpp", "cosurfmorphss", "cosurfmorphmr"),
  by = "compname",
  areasymbols = NULL,
  mukeys = NULL,
  WHERE = NULL,
  db = c("SSURGO", "STATSGO"),
  dsn = NULL,
  query_string = FALSE
)
```

Arguments

table Target table to summarize. Default: "cosurfmorphgc" (3D Geomorphic Component). Alternate choices include cosurfmorphhpp (2D Hillslope Position), cosurfmorphss (Surface Shape), and cosurfmorphmr (Microrelief).

by Grouping variable. Default: "compname"

areasymbols A vector of soil survey area symbols (e.g. 'CA067')

mukeys	A vector of map unit keys (e.g. 466627)
WHERE	WHERE clause added to SQL query. For example: areasymbol = 'CA067'
db	Either 'SSURGO' (default) or 'STATSGO'. If 'SSURGO' is specified areasymbol = 'US' records are excluded. If 'STATSGO' only areasymbol = 'US' records are included.
dsn	Path to local SSURGO database SQLite database. Default NULL uses Soil Data Access.
query_string	Return query instead of sending to Soil Data Access / local database. Default: FALSE.

Details

Default table="cosurfmorphgc" summarizes columns geomposmntn, geomposhill, geomposflats, and geompostrce. table="cosurfmorphhpp" summarizes "hillslopeprof", table="cosurfmorphss" summarizes shapeacross and shapedown, and table="cosurfmorphmr" summarizes geomicrorelief.

Queries are a generalization of now-deprecated functions from sharpshootR by Dylan Beaudette: geomPosMountainProbability(), geomPosHillProbability(), surfaceShapeProbability(), hillslopeProbability()

Similar summaries of SSURGO component surface morphometry data by series name can be found in fetchOSD(, extended=TRUE) or downloaded from https://github.com/ncss-tech/SoilWeb-data Full component data including surface morphometry summaries at the "site" level can be obtained with fetchSDA().

Value

a data.frame containing the grouping variable (by) and tabular summaries of counts and proportions of geomorphic records.

Author(s)

Dylan E. Beaudette, Andrew G. Brown

See Also

fetchSDA() get_SDA_pmgroupname()

Examples

```
## Not run:
 # Summarize by 3D geomorphic components by component name (default `by='compname'`)
 get_SDA_cosurfmorph(WHERE = "areasymbol = 'CA630'")

 # Whole Soil Survey Area summary (using `by = 'areasymbol'`)
 get_SDA_cosurfmorph(by = 'areasymbol', WHERE = "areasymbol = 'CA630'")

 # 2D Hillslope Position summary(using `table = 'cosurfmorphhpp'`)
 get_SDA_cosurfmorph('cosurfmorphhpp', WHERE = "areasymbol = 'CA630'")

 # Surface Shape summary (using `table = 'cosurfmorphss'`)
```

```
get_SDA_cosurfmorph('cosurfmorphss', WHERE = "areasymbol = 'CA630'")

# Microrelief summary (using `table = 'cosurfmorphmr'`)
get_SDA_cosurfmorph('cosurfmorphmr', WHERE = "areasymbol = 'CA630'")

## End(Not run)
```

`get_SDA_hydric` *Get map unit hydric soils information from Soil Data Access*

Description

Assess the hydric soils composition of a map unit.

Usage

```
get_SDA_hydric(
  areasymbols = NULL,
  mukeys = NULL,
  WHERE = NULL,
  method = "MAPUNIT",
  query_string = FALSE,
  dsn = NULL
)
```

Arguments

`areasymbols`	vector of soil survey area symbols
`mukeys`	vector of map unit keys
`WHERE`	character containing SQL WHERE clause specified in terms of fields in `legend`, `mapunit`, or `component` tables, used in lieu of `mukeys` or `areasymbols`
`method`	One of: `"Mapunit"`, `"Dominant Component"`, `"Dominant Condition"`, `"None"`
`query_string`	Default: `FALSE`; if `TRUE` return a character string containing query that would be sent to SDA via `SDA_query`
`dsn`	Path to local SQLite database or a DBIConnection object. If `NULL` (default) use Soil Data Access API via `SDA_query()`.

Details

The default classes for `method="MAPUNIT"` are as follows:

- `'Nonhydric'` - no hydric components
- `'Hydric'` - all hydric components
- `'Predominantly Hydric'` - hydric component percentage is 50% or more
- `'Partially Hydric'` - one or more of the major components is hydric

- 'Predominantly Nonhydric' - hydric component percentage is less than 50%

The default result will also include the following summaries of component percentages: total_comppct, hydric_majors and hydric_inclusions.

Default method "Mapunit" produces aggregate summaries of all components in the mapunit. Use "Dominant Component" and "Dominant Condition" to get the dominant component (highest percentage) or dominant hydric condition (similar conditions aggregated across components), respectively. Use "None" for no aggregation (one record per component).

Value

a data.frame

Author(s)

Jason Nemecek, Chad Ferguson, Andrew Brown

---

get_SDA_interpretation

Get map unit interpretations from Soil Data Access by rule name

---

Description

Get map unit interpretations from Soil Data Access by rule name

Usage

```
get_SDA_interpretation(
  rulename,
 method = c("Dominant Component", "Dominant Condition", "Weighted Average", "None"),
  areasymbols = NULL,
  mukeys = NULL,
  WHERE = NULL,
  query_string = FALSE,
  not_rated_value = NA_real_,
  dsn = NULL
)
```

Arguments

rulename	character vector of interpretation rule names (matching mrulename in cointerp table)
method	aggregation method. One of: "Dominant Component", "Dominant Condition", "Weighted Average", "None". If "None" is selected one row will be returned per component, otherwise one row will be returned per map unit.
areasymbols	vector of soil survey area symbols
mukeys	vector of map unit keys

WHERE	character containing SQL WHERE clause specified in terms of fields in legend, mapunit, or component tables, used in lieu of mukeys or areasymbols
query_string	Default: FALSE; if TRUE return a character string containing query that would be sent to SDA via SDA_query
not_rated_value	used where rating class is "Not Rated". Default: NA_real
dsn	Path to local SQLite database or a DBIConnection object. If NULL (default) use Soil Data Access API via SDA_query().

Details

Rule Names in `cointerp` **table:**

- AGR-Agronomic Concerns (ND)
- AGR-Available Water Capacity (ND)
- AGR-Natural Fertility (ND)
- AGR-Pesticide and Nutrient Leaching Potential, NIRR (ND)
- AGR-Pesticide and Nutrient Runoff Potential (ND)
- AGR-Physical Limitations (ND)
- AGR-Rooting Depth (ND)
- AGR-Sodicity (ND)
- AGR-Subsurface Salinity (ND)
- AGR-Surface Crusting (ND)
- AGR-Surface Salinity (ND)
- AGR-Water Erosion (ND)
- AGR-Wind Erosion (ND)
- AGR - Air Quality; PM10 (TX)
- AGR - Air Quality; PM2_5 (TX)
- AGR - Avocado Root Rot Hazard (CA)
- AGR - Barley Yield (MT)
- AGR - California Revised Storie Index (CA)
- AGR - Conventional Tillage (TX)
- AGR - Filter Strips (TX)
- AGR - Grape non-irrigated (MO)
- AGR - Hops Site Suitability (WA)
- AGR - Index for alfalfa hay, irrigated (NV)
- AGR - Map Unit Cropland Productivity (MN)
- AGR - Mulch Till (TX)
- AGR - Nitrate Leaching Potential, Irrigated (WA)
- AGR - Nitrate Leaching Potential, Nonirrigated (MA)
- AGR - Nitrate Leaching Potential, Nonirrigated (MT)
- AGR - Nitrate Leaching Potential, Nonirrigated (WA)
- AGR - No Till (TX)
- AGR - No Till (VT)

- AGR - No Till, Tile Drained (TX)
- AGR - Oats Yield (MT)
- AGR - Orchard Groups (TX)
- AGR - Pasture hayland (MO)
- AGR - Pesticide Loss Potential-Leaching
- AGR - Pesticide Loss Potential-Leaching (NE)
- AGR - Pesticide Loss Potential-Soil Surface Runoff
- AGR - Pesticide Loss Potential-Soil Surface Runoff (NE)
- AGR - Plant Growth Index PGI no Climate Adj. (TX)
- AGR - Plant Growth Index PGI with Climate Adj. (TX)
- AGR - Plant Growth Index PGI with Climate Adj. MAP,MAAT (TX)
- AGR - Rangeland Grass/Herbaceous Productivity Index (TX)
- AGR - Ridge Till (TX)
- AGR - Rutting Hazard =< 10,000 Pounds per Wheel (TX)
- AGR - Rutting Hazard > 10,000 Pounds per Wheel (TX)
- AGR - Selenium Leaching Potential (CO)
- AGR - Spring Wheat Yield (MT)
- AGR - Water Erosion Potential (TX)
- AGR - Water Erosion Potential Wide Ratings Array (TX)
- AGR - Wind Erosion Potential (TX)
- AGR - Wind Erosion Potential Wide Ratings Array (TX)
- AGR - Wine Grape Site Suitability (WA)
- AGR - Winter Wheat Yield (MT)
- Alaska Exempt Wetland Potential (AK)
- American Wine Grape Varieties Site Desirability (Long)
- American Wine Grape Varieties Site Desirability (Medium)
- American Wine Grape Varieties Site Desirability (Short)
- American Wine Grape Varieties Site Desirability (Very Long)
- AWM - Animal Mortality Disposal (Catastrophic) (MO)
- AWM - Filter Group (OH)
- AWM - Irrigation Disposal of Wastewater
- AWM - Irrigation Disposal of Wastewater (DE)
- AWM - Irrigation Disposal of Wastewater (MD)
- AWM - Irrigation Disposal of Wastewater (OH)
- AWM - Irrigation Disposal of Wastewater (VT)
- AWM - Land App of Municipal Sewage Sludge (DE)
- AWM - Land App of Municipal Sewage Sludge (MD)
- AWM - Land Application of Dry and Slurry Manure (TX)
- AWM - Land Application of Milk (CT)
- AWM - Land Application of Municipal Biosolids, spring (OR)
- AWM - Land Application of Municipal Biosolids, summer (OR)
- AWM - Land Application of Municipal Biosolids, winter (OR)

- AWM - Land Application of Municipal Sewage Sludge
- AWM - Land Application of Municipal Sewage Sludge (OH)
- AWM - Land Application of Municipal Sewage Sludge (VT)
- AWM - Large Animal Disposal, Pit (MN)
- AWM - Manure and Food Processing Waste
- AWM - Manure and Food Processing Waste (DE)
- AWM - Manure and Food Processing Waste (MD)
- AWM - Manure and Food Processing Waste (OH)
- AWM - Manure and Food Processing Waste (VT)
- AWM - Manure Stacking - Site Evaluation (TX)
- AWM - Overland Flow Process Treatment of Wastewater
- AWM - Overland Flow Process Treatment of Wastewater (VT)
- AWM - Phosphorus Management (TX)
- AWM - Rapid Infil Disposal of Wastewater (DE)
- AWM - Rapid Infil Disposal of Wastewater (MD)
- AWM - Rapid Infiltration Disposal of Wastewater
- AWM - Rapid Infiltration Disposal of Wastewater (VT)
- AWM - Sensitive Soil Features (MN)
- AWM - Slow Rate Process Treatment of Wastewater
- AWM - Slow Rate Process Treatment of Wastewater (VT)
- AWM - Vegetated Treatment Area (PIA)
- AWM - Waste Field Storage Area (VT)
- BLM-Reclamation Suitability (MT)
- BLM - Chaining Suitability
- BLM - Fencing
- BLM - Fire Damage Susceptibility
- BLM - Fugitive Dust Resistance
- BLM - Mechanical Treatment, Rolling Drum
- BLM - Mechanical Treatment, Shredder
- BLM - Medusahead Invasion Susceptibility
- BLM - Pygmy Rabbit Habitat Potential
- BLM - Rangeland Drill
- BLM - Rangeland Seeding, Colorado Plateau Ecoregion
- BLM - Rangeland Seeding, Great Basin Ecoregion
- BLM - Rangeland Tillage
- BLM - Site Degradation Susceptibility
- BLM - Soil Compaction Resistance
- BLM - Soil Restoration Potential
- BLM - Yellow Star-thistle Invasion Susceptibility
- CA Prime Farmland (CA)
- Capping Fill Gravity Septic System (DE)
- CLASS RULE - Depth to any bedrock kind (5 classes) (NPS)

- CLASS RULE - Depth to lithic bedrock (5 classes) (NPS)
- CLASS RULE - Depth to non-lithic bedrock (5 classes) (NPS)
- CLASS RULE - Depth to root limiting layer (5 classes) (NPS)
- CLASS RULE - Soil Inorganic Carbon kg/m2 to 2m (NPS)
- CLASS RULE - Soil Organic Carbon kg/m2 to 2m (NPS)
- CLR-cropland limitation for corn and soybeans (IN)
- CLR-pastureland limitation (IN)
- Commodity Crop Productivity Index (Corn) (WI)
- CPI - Alfalfa Hay, IRR - Eastern Idaho Plateaus (ID)
- CPI - Alfalfa Hay, IRR - Klamath Valley and Basins (OR)
- CPI - Alfalfa Hay, IRR - Snake River Plains (ID)
- CPI - Alfalfa Hay, NIRR- Eastern Idaho Plateaus (ID)
- CPI - Alfalfa Hay, NIRR - Palouse, Northern Rocky Mtns. (ID)
- CPI - Alfalfa Hay, NIRR - Palouse, Northern Rocky Mtns. (WA)
- CPI - Barley, IRR - Eastern Idaho Plateaus (ID)
- CPI - Barley, NIRR - Eastern Idaho Plateaus (ID)
- CPI - Grass Hay, IRR - Eastern Idaho Plateaus (ID)
- CPI - Grass Hay, IRR - Klamath Valleys and Basins (OR)
- CPI - Grass Hay, NIRR - Klamath Valleys and Basins (OR)
- CPI - Grass Hay, NIRR - Palouse, Northern Rocky Mtns. (ID)
- CPI - Grass Hay, NIRR - Palouse, Northern Rocky Mtns. (WA)
- CPI - Potatoes, IRR - Eastern Idaho Plateaus (ID)
- CPI - Potatoes, IRR - Snake River Plains (ID)
- CPI - Small Grains Productivity Index (AK)
- CPI - Small Grains, IRR - Snake River Plains (ID)
- CPI - Small Grains, NIRR - Palouse Prairies (ID)
- CPI - Small Grains, NIRR - Palouse Prairies (OR)
- CPI - Small Grains, NIRR - Palouse Prairies (WA)
- CPI - Small Grains, NIRR - Snake River Plains (ID)
- CPI - Wheat, IRR - Eastern Idaho Plateaus (ID)
- CPI - Wheat, NIRR - Eastern Idaho Plateaus (ID)
- CPI - Wild Hay, NIRR - Eastern Idaho Plateaus (ID)
- CPI - Wild Hay, NIRR - Palouse, Northern Rocky Mtns. (ID)
- CPI - Wild Hay, NIRR - Palouse, Northern Rocky Mtns. (WA)
- Deep Infiltration Systems
- DHS - Catastrophic Event, Large Animal Mortality, Burial
- DHS - Catastrophic Event, Large Animal Mortality, Incinerate
- DHS - Catastrophic Mortality, Large Animal Disposal, Pit
- DHS - Catastrophic Mortality, Large Animal Disposal, Trench
- DHS - Emergency Animal Mortality Disposal by Shallow Burial
- DHS - Emergency Land Disposal of Milk
- DHS - Potential for Radioactive Bioaccumulation

- DHS - Potential for Radioactive Sequestration
- DHS - Rubble and Debris Disposal, Large-Scale Event
- DHS - Site for Composting Facility - Subsurface
- DHS - Site for Composting Facility - Surface
- DHS - Suitability for Clay Liner Material
- DHS - Suitability for Composting Medium and Final Cover
- Elevated Sand Mound Septic System (DE)
- ENG - Animal Disposal by Composting (Catastrophic) (WV)
- ENG - Application of Municipal Sludge (TX)
- ENG - Aquifer Assessment - 7081 (MN)
- ENG - Closed-Loop Horizontal Geothermal Heat Pump (CT)
- ENG - Cohesive Soil Liner (MN)
- ENG - Construction Materials - Gravel Source (MN)
- ENG - Construction Materials - Sand Source (MN)
- ENG - Construction Materials; Gravel Source
- ENG - Construction Materials; Gravel Source (AK)
- ENG - Construction Materials; Gravel Source (CT)
- ENG - Construction Materials; Gravel Source (ID)
- ENG - Construction Materials; Gravel Source (IN)
- ENG - Construction Materials; Gravel Source (MI)
- ENG - Construction Materials; Gravel Source (NE)
- ENG - Construction Materials; Gravel Source (NY)
- ENG - Construction Materials; Gravel Source (OH)
- ENG - Construction Materials; Gravel Source (OR)
- ENG - Construction Materials; Gravel Source (VT)
- ENG - Construction Materials; Gravel Source (WA)
- ENG - Construction Materials; Reclamation
- ENG - Construction Materials; Reclamation (DE)
- ENG - Construction Materials; Reclamation (MD)
- ENG - Construction Materials; Reclamation (MI)
- ENG - Construction Materials; Reclamation (OH)
- ENG - Construction Materials; Roadfill
- ENG - Construction Materials; Roadfill (AK)
- ENG - Construction Materials; Roadfill (GA)
- ENG - Construction Materials; Roadfill (OH)
- ENG - Construction Materials; Sand Source
- ENG - Construction Materials; Sand Source (AK)
- ENG - Construction Materials; Sand Source (CT)
- ENG - Construction Materials; Sand Source (GA)
- ENG - Construction Materials; Sand Source (ID)
- ENG - Construction Materials; Sand Source (IN)
- ENG - Construction Materials; Sand Source (NY)

- ENG - Construction Materials; Sand Source (OH)
- ENG - Construction Materials; Sand Source (OR)
- ENG - Construction Materials; Sand Source (VT)
- ENG - Construction Materials; Sand Source (WA)
- ENG - Construction Materials; Topsoil
- ENG - Construction Materials; Topsoil (AK)
- ENG - Construction Materials; Topsoil (DE)
- ENG - Construction Materials; Topsoil (GA)
- ENG - Construction Materials; Topsoil (ID)
- ENG - Construction Materials; Topsoil (MD)
- ENG - Construction Materials; Topsoil (MI)
- ENG - Construction Materials; Topsoil (OH)
- ENG - Construction Materials; Topsoil (OR)
- ENG - Construction Materials; Topsoil (WA)
- ENG - Daily Cover for Landfill
- ENG - Daily Cover for Landfill (AK)
- ENG - Daily Cover for Landfill (OH)
- ENG - Disposal Field (NJ)
- ENG - Disposal Field Gravity (DE)
- ENG - Disposal Field Suitability Class (NJ)
- ENG - Disposal Field Type Inst (NJ)
- ENG - Dwellings W/O Basements
- ENG - Dwellings W/O Basements (OH)
- ENG - Dwellings With Basements
- ENG - Dwellings with Basements (AK)
- ENG - Dwellings With Basements (OH)
- ENG - Dwellings without Basements (AK)
- ENG - Large Animal Disposal, Pit (CT)
- ENG - Large Animal Disposal, Trench (CT)
- ENG - Lawn and Landscape (OH)
- ENG - Lawn, Landscape, Golf Fairway
- ENG - Lawn, landscape, golf fairway (CT)
- ENG - Lawn, Landscape, Golf Fairway (MI)
- ENG - Lawn, Landscape, Golf Fairway (VT)
- ENG - Local Roads and Streets
- ENG - Local Roads and Streets (AK)
- ENG - Local Roads and Streets (GA)
- ENG - Local Roads and Streets (OH)
- ENG - New Ohio Septic Rating (OH)
- ENG - On-Site Waste Water Absorption Fields (MO)
- ENG - On-Site Waste Water Lagoons (MO)
- ENG - OSHA Soil Types (TX)

- ENG - Pier Beam Building Foundations (TX)
- ENG - Sanitary Landfill (Area)
- ENG - Sanitary Landfill (Area) (AK)
- ENG - Sanitary Landfill (Area) (OH)
- ENG - Sanitary Landfill (Trench)
- ENG - Sanitary Landfill (Trench) (AK)
- ENG - Sanitary Landfill (Trench) (OH)
- ENG - Septage Application - Incorporation or Injection (MN)
- ENG - Septage Application - Surface (MN)
- ENG - Septic System; Disinfection, Surface Application (TX)
- ENG - Septic Tank Absorption Fields
- ENG - Septic Tank Absorption Fields - At-Grade (MN)
- ENG - Septic Tank Absorption Fields - Mound (MN)
- ENG - Septic Tank Absorption Fields - Trench (MN)
- ENG - Septic Tank Absorption Fields (AK)
- ENG - Septic Tank Absorption Fields (DE)
- ENG - Septic Tank Absorption Fields (FL)
- ENG - Septic Tank Absorption Fields (MD)
- ENG - Septic Tank Absorption Fields (NY)
- ENG - Septic Tank Absorption Fields (OH)
- ENG - Septic Tank Absorption Fields (TX)
- ENG - Septic Tank Leaching Chamber (TX)
- ENG - Septic Tank, Gravity Disposal (TX)
- ENG - Septic Tank, Subsurface Drip Irrigation (TX)
- ENG - Sewage Lagoons
- ENG - Sewage Lagoons (AK)
- ENG - Sewage Lagoons (OH)
- ENG - Shallow Excavations
- ENG - Shallow Excavations (AK)
- ENG - Shallow Excavations (MI)
- ENG - Shallow Excavations (OH)
- ENG - Small Commercial Buildings
- ENG - Small Commercial Buildings (OH)
- ENG - Soil Potential of Road Salt Applications (CT)
- ENG - Soil Potential Ratings of SSDS (CT)
- ENG - Source of Caliche (TX)
- ENG - Stormwater Management / Infiltration (NY)
- ENG - Stormwater Management / Ponds (NY)
- ENG - Stormwater Management / Wetlands (NY)
- ENG - Unpaved Local Roads and Streets
- Farm and Garden Composting Facility - Surface
- FOR-Biomass Harvest (WI)

- FOR-Construction Limitations for Haul Roads/Log Landings(ME)
- FOR - Biomass Harvest (MA)
- FOR - Black Walnut Suitability Index (KS)
- FOR - Black Walnut Suitability Index (MO)
- FOR - Compaction Potential (WA)
- FOR - Construction Limitations - Haul Roads/Log Landing (OH)
- FOR - Construction Limitations For Haul Roads (MI)
- FOR - Construction Limitations for Haul Roads/Log Landings
- FOR - Damage by Fire (OH)
- FOR - Displacement Hazard
- FOR - Displacement Potential (WA)
- FOR - General Harvest Season (ME)
- FOR - General Harvest Season (VT)
- FOR - Hand Planting Suitability
- FOR - Hand Planting Suitability (ME)
- FOR - Hand Planting Suitability, MO13 (DE)
- FOR - Hand Planting Suitability, MO13 (MD)
- FOR - Harvest Equipment Operability
- FOR - Harvest Equipment Operability (DE)
- FOR - Harvest Equipment Operability (MD)
- FOR - Harvest Equipment Operability (ME)
- FOR - Harvest Equipment Operability (MI)
- FOR - Harvest Equipment Operability (OH)
- FOR - Harvest Equipment Operability (VT)
- FOR - Log Landing Suitability
- FOR - Log Landing Suitability (ID)
- FOR - Log Landing Suitability (ME)
- FOR - Log Landing Suitability (MI)
- FOR - Log Landing Suitability (OR)
- FOR - Log Landing Suitability (VT)
- FOR - Log Landing Suitability (WA)
- FOR - Mechanical Planting Suitability
- FOR - Mechanical Planting Suitability (CT)
- FOR - Mechanical Planting Suitability (ME)
- FOR - Mechanical Planting Suitability (OH)
- FOR - Mechanical Planting Suitability, MO13 (DE)
- FOR - Mechanical Planting Suitability, MO13 (MD)
- FOR - Mechanical Site Preparation (Deep)
- FOR - Mechanical Site Preparation (Deep) (DE)
- FOR - Mechanical Site Preparation (Deep) (MD)
- FOR - Mechanical Site Preparation (Surface)
- FOR - Mechanical Site Preparation (Surface) (DE)

- FOR - Mechanical Site Preparation (Surface) (MD)
- FOR - Mechanical Site Preparation (Surface) (MI)
- FOR - Mechanical Site Preparation (Surface) (OH)
- FOR - Mechanical Site Preparation; Deep (CT)
- FOR - Mechanical Site Preparation; Surface (ME)
- FOR - Potential Erosion Hazard (Off-Road/Off-Trail)
- FOR - Potential Erosion Hazard (Off-Road/Off-Trail) (MI)
- FOR - Potential Erosion Hazard (Off-Road/Off-Trail) (OH)
- FOR - Potential Erosion Hazard (Road/Trail)
- FOR - Potential Erosion Hazard (Road/Trail) (PIA)
- FOR - Potential Erosion Hazard, Road/Trail, Spring Thaw (AK)
- FOR - Potential Fire Damage Hazard
- FOR - Potential Seedling Mortality
- FOR - Potential Seedling Mortality (FL)
- FOR - Potential Seedling Mortality (MI)
- FOR - Potential Seedling Mortality (OH)
- FOR - Potential Seedling Mortality (PIA)
- FOR - Potential Seedling Mortality (VT)
- FOR - Potential Seedling Mortality(ME)
- FOR - Potential Windthrow Hazard (ME)
- FOR - Potential Windthrow Hazard (MI)
- FOR - Potential Windthrow Hazard (NY)
- FOR - Potential Windthrow Hazard (VT)
- FOR - Puddling Hazard
- FOR - Puddling Potential (WA)
- FOR - Road Suitability (Natural Surface)
- FOR - Road Suitability (Natural Surface) (ID)
- FOR - Road Suitability (Natural Surface) (ME)
- FOR - Road Suitability (Natural Surface) (OH)
- FOR - Road Suitability (Natural Surface) (OR)
- FOR - Road Suitability (Natural Surface) (VT)
- FOR - Road Suitability (Natural Surface) (WA)
- FOR - Rutting Hazard by Month
- FOR - Rutting Hazard by Season
- FOR - Shortleaf pine littleleaf disease susceptibility
- FOR - Soil Compactibility Risk
- FOR - Soil Rutting Hazard
- FOR - Soil Rutting Hazard (ME)
- FOR - Soil Rutting Hazard (OH)
- FOR - Soil Sustainability Forest Biomass Harvesting (CT)
- FOR - White Oak Suitability (MO)
- FOR - Windthrow Hazard

- FOR - Windthrow Hazard (WA)
- FOR (USFS) - Road Construction/Maintenance (Natural Surface)
- FOTG - Indiana Corn Yield Calculation (IN)
- FOTG - Indiana Slippage Potential (IN)
- FOTG - Indiana Soy Bean Yield Calculation (IN)
- FOTG - Indiana Wheat Yield Calculation (IN)
- Fragile Soil Index
- Gravity Full Depth Septic System (DE)
- GRL-FSG-NP-W (MT)
- GRL - Excavations to 24 inches for Plastic Pipelines (TX)
- GRL - Fencing, 24 inch Post Depth (MT)
- GRL - Fencing, Post Depth =<24 inches
- GRL - Fencing, Post Depth =<36 inches
- GRL - Fencing, Post Depth Less Than 24 inches (TX)
- GRL - Fencing, Post Depth Less Than 36 inches (TX)
- GRL - Juniper Encroachment Potential (NM)
- GRL - NV range seeding (Wind C = 10) (NV)
- GRL - NV range seeding (Wind C = 100) (NV)
- GRL - NV range seeding (Wind C = 20) (NV)
- GRL - NV range seeding (Wind C = 30) (NV)
- GRL - NV range seeding (Wind C = 40) (NV)
- GRL - NV range seeding (Wind C = 50) (NV)
- GRL - NV range seeding (Wind C = 60) (NV)
- GRL - NV range seeding (Wind C = 80) (NV)
- GRL - NV range seeding (Wind C >= 160) (NV)
- GRL - Pasture and Hayland SG (OH)
- GRL - Ranch Access Roads (TX)
- GRL - Rangeland Chaining (TX)
- GRL - Rangeland Disking (TX)
- GRL - Rangeland Dozing/Grubbing (TX)
- GRL - Rangeland Planting by Mechanical Seeding (TX)
- GRL - Rangeland Prescribed Burning (TX)
- GRL - Rangeland Roller Chopping (TX)
- GRL - Rangeland Root Plowing (TX)
- GRL - Utah Juniper Encroachment Potential
- GRL - Western Juniper Encroachment Potential (OR)
- Ground-based Solar Arrays, Ballast Anchor Systems
- Ground-based Solar Arrays, Soil-penetrating Anchor Systems
- Ground Penetrating Radar Penetration
- Hybrid Wine Grape Varieties Site Desirability (Long)
- Hybrid Wine Grape Varieties Site Desirability (Medium)
- Hybrid Wine Grape Varieties Site Desirability (Short)

- Inland Wetlands (CT)
- IRR-restrictive features for irrigation (OH)
- ISDH Septic Tank Interpretation (IN)
- Land Application of Municipal Sewage Sludge (PA)
- Lined Retention Systems
- Low Pressure Pipe Septic System (DE)
- MIL - Bivouac Areas (DOD)
- MIL - Excavations Crew-Served Weapon Fighting Position (DOD)
- MIL - Excavations for Individual Fighting Position (DOD)
- MIL - Excavations for Vehicle Fighting Position (DOD)
- MIL - Helicopter Landing Zones (DOD)
- MIL - Trafficability Veh. Type 1 1-pass wet season (DOD)
- MIL - Trafficability Veh. Type 1 50-passes wet season (DOD)
- MIL - Trafficability Veh. Type 1 dry season (DOD)
- MIL - Trafficability Veh. Type 2 1-pass wet season (DOD)
- MIL - Trafficability Veh. Type 2 50-passes wet season (DOD)
- MIL - Trafficability Veh. Type 2 dry season (DOD)
- MIL - Trafficability Veh. Type 3 1-pass wet season (DOD)
- MIL - Trafficability Veh. Type 3 50-passes wet season (DOD)
- MIL - Trafficability Veh. Type 3 dry season (DOD)
- MIL - Trafficability Veh. Type 4 1-pass wet season (DOD)
- MIL - Trafficability Veh. Type 4 50-passes wet season (DOD)
- MIL - Trafficability Veh. Type 4 dry season (DOD)
- MIL - Trafficability Veh. Type 5 1-pass wet season (DOD)
- MIL - Trafficability Veh. Type 5 50-passes wet season (DOD)
- MIL - Trafficability Veh. Type 5 dry season (DOD)
- MIL - Trafficability Veh. Type 6 1-pass wet season (DOD)
- MIL - Trafficability Veh. Type 6 50-passes wet season (DOD)
- MIL - Trafficability Veh. Type 6 dry season (DOD)
- MIL - Trafficability Veh. Type 7 1-pass wet season (DOD)
- MIL - Trafficability Veh. Type 7 50-passes wet season (DOD)
- MIL - Trafficability Veh. Type 7 dry season (DOD)
- MT - Conservation Tree/Shrub Groups (MT)
- Muscadine Wine Grape Site Desirability (Very Long)
- NCCPI - Irrigated National Commodity Crop Productivity Index
- NCCPI - National Commodity Crop Productivity Index (Ver 3.0)
- NCCPI - NCCPI Corn Submodel (I)
- NCCPI - NCCPI Cotton Submodel (II)
- NCCPI - NCCPI Small Grains Submodel (II)
- NCCPI - NCCPI Soybeans Submodel (I)
- Nitrogen Loss Potential (ND)
- Permafrost Sensitivity (AK)

- Pressure Dose Capping Fill Septic System (DE)
- Pressure Dose Full Depth Septic System (DE)
- REC - Camp and Picnic Areas (AK)
- REC - Camp Areas (CT)
- REC - Camp Areas; Primitive (AK)
- REC - Foot and ATV Trails (AK)
- REC - Off-Road Motorcycle Trails (CT)
- REC - Paths and Trails (CT)
- REC - Picnic Areas (CT)
- REC - Playgrounds (AK)
- REC - Playgrounds (CT)
- RSK-risk assessment for manure application (OH)
- Salinity Risk Index, Discharge Model (ND)
- SAS - CMECS Substrate Class
- SAS - CMECS Substrate Origin
- SAS - CMECS Substrate Subclass
- SAS - CMECS Substrate Subclass/Group
- SAS - CMECS Substrate Subclass/Group/Subgroup
- SAS - Eastern Oyster Habitat Restoration Suitability
- SAS - Eelgrass Restoration Suitability
- SAS - Land Utilization of Dredged Materials
- SAS - Mooring Anchor - Deadweight
- SAS - Mooring Anchor - Mushroom
- SAS - Northern Quahog (Hard Clam) Habitat Suitability
- Septic System A/B Soil System (Alternate) (PA)
- Septic System At-Grade Bed (Alternate) (PA)
- Septic System At Grade Shallow Field (alternative) (WV)
- Septic System CO-OP RFS III w/At-Grade Bed (PA)
- Septic System CO-OP RFS III w/Drip Irrigation (PA)
- Septic System CO-OP RFS III w/Spray Irrigation (PA)
- Septic System Drip Irrigation (Alternate) (PA)
- Septic System Drip Irrigation (alternative) (WV)
- Septic System Dual Field Trench (conventional) (WV)
- Septic System Elevated Field (alternative) (WV)
- Septic System Free Access Sand Filter w/At-Grade Bed (PA)
- Septic System Free Access Sand Filter w/Drip Irrigation (PA)
- Septic System Free Access Sand Filterw/Spray Irrigation (PA)
- Septic System In Ground Bed (conventional) (PA)
- Septic System In Ground Trench (conventional) (PA)
- Septic System In Ground Trench (conventional) (WV)
- Septic System Low Pressure Pipe (alternative) (WV)
- Septic System Modified Subsurface Sand Filter (Alt.) (PA)

- Septic System Mound (alternative) (WV)
- Septic System Peat Based Option1 (UV & At-Grade Bed)Alt (PA)
- Septic System Peat Based Option1 w/At-Grade Bed (Alt.) (PA)
- Septic System Peat Based Option2 w/Spray Irrigation (PA)
- Septic System Peat Sys Opt3 w/Subsurface Sand Filter (PA)
- Septic System Sand Mound Bed or Trench (PA)
- Septic System Shallow In Ground Trench (conventional) (WV)
- Septic System Shallow Placement Pressure Dosed (Alt.) (PA)
- Septic System Spray Irrigation (PA)
- Septic System Steep Slope Mound (alternative) (WV)
- Septic System Steep Slope Sand Mound (Alternate) (PA)
- Septic System Subsurface Sand Filter Bed (conventional) (PA)
- Septic System Subsurface Sand Filter Trench (standard) (PA)
- Shallow Infiltration Systems
- SOH - Suitability for Aerobic Soil Organisms
- SOH - Agricultural Organic Soil Subsidence
- SOH - Concentration of Salts- Soil Surface
- SOH - Organic Matter Depletion
- SOH - Soil Surface Sealing
- SOH - Soil Susceptibility to Compaction
- Soil Habitat for Saprophyte Stage of Coccidioides
- SOIL HEALTH ASSESSMENT (NJ)
- Soil Vegetative Groups (CA)
- Surface Runoff Class (CA)
- Unlined Retention Systems
- URB - Commercial Brick Bldg; w/Reinforced Concrete Slab (TX)
- URB - Commercial Brick Buildings w/Concrete Slab (TX)
- URB - Commercial Metal Bldg; w/Concrete Slab (TX)
- URB - Commercial Metal Bldg; w/Reinforced Concrete Slab (TX)
- URB - Commercial Metal Buildings w/o Concrete Slab (TX)
- URB - Concrete Driveways and Sidewalks (TX)
- URB - Dwellings on Concrete Slab (TX)
- URB - Dwellings With Basements (TX)
- URB - Lawns and Ornamental Plantings (TX)
- URB - Reinforced Concrete Slab (TX)
- URB - Rural Residential Development on Concrete Slab (TX)
- URB - Rural Residential Development w/Basement (TX)
- URB - Urban Residential Development on Concrete Slab (TX)
- URB - Urban Residential Development w/Basement (TX)
- URB/REC - Camp Areas
- URB/REC - Camp Areas (GA)
- URB/REC - Camp Areas (HI)

- URB/REC - Camp Areas (MI)
- URB/REC - Camp Areas (OH)
- URB/REC - Golf Fairways (OH)
- URB/REC - Off-Road Motorcycle Trails
- URB/REC - Off-Road Motorcycle Trails (OH)
- URB/REC - Paths and Trails
- URB/REC - Paths and Trails (GA)
- URB/REC - Paths and Trails (MI)
- URB/REC - Paths and Trails (OH)
- URB/REC - Picnic Areas
- URB/REC - Picnic Areas (GA)
- URB/REC - Picnic Areas (MI)
- URB/REC - Picnic Areas (OH)
- URB/REC - Playgrounds
- URB/REC - Playgrounds (GA)
- URB/REC - Playgrounds (MI)
- URB/REC - Playgrounds (OH)
- Vinifera Wine Grape Site Desirability (Long to Medium)
- Vinifera Wine Grape Site Desirability (Long)
- Vinifera Wine Grape Site Desirability (Short to Medium)
- Vinifera Wine Grape Site Desirability (Short)
- WAQ - Soil Pesticide Absorbed Runoff Potential (TX)
- WAQ - Soil Pesticide Leaching Potential (TX)
- WAQ - Soil Pesticide Solution Runoff Potential (TX)
- WLF-Soil Suitability - Karner Blue Butterfly (WI)
- WLF - Burrowing Mammals & Reptiles (TX)
- WLF - Chufa for Turkey Forage (LA)
- WLF - Crawfish Aquaculture (TX)
- WLF - Desert Tortoise (CA)
- WLF - Desertic Herbaceous Plants (TX)
- WLF - Domestic Grasses & Legumes for Food and Cover (TX)
- WLF - Food Plots for Upland Wildlife < 2 Acres (TX)
- WLF - Freshwater Wetland Plants (TX)
- WLF - Gopher Tortoise Burrowing Suitability
- WLF - Grain & Seed Crops for Food and Cover (TX)
- WLF - Irr. Domestic Grasses & Legumes for Food & Cover (TX)
- WLF - Irrigated Freshwater Wetland Plants (TX)
- WLF - Irrigated Grain & Seed Crops for Food & Cover (TX)
- WLF - Irrigated Saline Water Wetland Plants (TX)
- WLF - Riparian Herbaceous Plants (TX)
- WLF - Riparian Shrubs, Vines, & Trees (TX)
- WLF - Saline Water Wetland Plants (TX)

- WLF - Upland Coniferous Trees (TX)
- WLF - Upland Deciduous Trees (TX)
- WLF - Upland Desertic Shrubs & Trees (TX)
- WLF - Upland Mixed Deciduous & Coniferous Trees (TX)
- WLF - Upland Native Herbaceous Plants (TX)
- WLF - Upland Shrubs & Vines (TX)
- WMS-Subsurface Water Management, Installation (ND)
- WMS-Subsurface Water Management, Outflow Quality (ND)
- WMS-Subsurface Water Management, Performance (ND)
- WMS - Constructing Grassed Waterways (OH)
- WMS - Constructing Grassed Waterways (TX)
- WMS - Constructing Terraces & Diversions (TX)
- WMS - Constructing Terraces and Diversions (OH)
- WMS - Drainage - (MI)
- WMS - Drainage (IL)
- WMS - Drainage (OH)
- WMS - Embankments, Dikes, and Levees
- WMS - Embankments, Dikes, and Levees (OH)
- WMS - Embankments, Dikes, and Levees (VT)
- WMS - Excavated Ponds (Aquifer-fed)
- WMS - Excavated Ponds (Aquifer-fed) (OH)
- WMS - Excavated Ponds (Aquifer-fed) (VT)
- WMS - Grape Production with Drip Irrigation (TX)
- WMS - Grassed Waterways - (MI)
- WMS - Irrigation, General
- WMS - Irrigation, Micro (above ground)
- WMS - Irrigation, Micro (above ground) (VT)
- WMS - Irrigation, Micro (subsurface drip)
- WMS - Irrigation, Micro (subsurface drip) (VT)
- WMS - Irrigation, Sprinkler (close spaced outlet drops)
- WMS - Irrigation, Sprinkler (general)
- WMS - Irrigation, Sprinkler (general) (VT)
- WMS - Irrigation, Surface (graded)
- WMS - Irrigation, Surface (level)
- WMS - Pond Reservoir Area
- WMS - Pond Reservoir Area (GA)
- WMS - Pond Reservoir Area (MI)
- WMS - Pond Reservoir Area (OH)
- WMS - Sprinkler Irrigation (MT)
- WMS - Sprinkler Irrigation RDC (IL)
- WMS - Subsurface Drains - Installation (VT)
- WMS - Subsurface Drains - Performance (VT)

- WMS - Subsurface Drains < 3 Feet Deep (TX)
- WMS - Subsurface Drains > 3 Feet Deep (TX)
- WMS - Subsurface Water Management, Outflow Quality
- WMS - Subsurface Water Management, System Installation
- WMS - Subsurface Water Management, System Performance
- WMS - Surface Drains (TX)
- WMS - Surface Irrigation Intake Family (TX)
- WMS - Surface Water Management, System

Value

a data.frame

Author(s)

Jason Nemecek, Chad Ferguson, Andrew Brown

Examples

```
if(requireNamespace("curl") &
    curl::has_internet()) {

 # get two forestry interpretations for CA630
 get_SDA_interpretation(c("FOR - Potential Seedling Mortality",
                          "FOR - Road Suitability (Natural Surface)"),
                        method = "Dominant Condition",
                        areasymbols = "CA630")
}
```

`get_SDA_metrics` *Get Soil Data Access, Lab Data Mart and Web Soil Survey Usage Metrics*

Description

Obtain pre-calculated tabular reports of usage, activities, areas of interest (AOI), exports, ecological sites, ratings and reports for specific areas, times and intervals.

Usage

```
get_SDA_metrics(query_name, query_frequency, query_year, state = NULL)
```

Arguments

query_name	One or more of: 'LDM_Usage', 'SDA_Usage', 'wss_ActivityCounts', 'wss_AOIDefinition', 'wss_AOISizeRange', 'wss_ExportCounts', 'wss_PrintableOutput', 'wss_top100AOIs', 'wss_top100Ecologicalsites', 'wss_top100ratings', 'wss_top100reports'
query_frequency	One or more of: 'M', 'CY', 'FY'
query_year	Integer. One or more years e.g. 2020:2021
state	Optional: State abbreviation; Default: NULL uses "xnational" for all states.

Value

A data.frame containing query results

Author(s)

Jason Nemecek

Examples

```
## Not run:
get_SDA_metrics('SDA_Usage', 'CY', 2019:2021)

## End(Not run)
```

---

get_SDA_muaggatt *Get map unit aggregate attribute information from Soil Data Access*

Description

Get map unit aggregate attribute information from Soil Data Access

Usage

```
get_SDA_muaggatt(
  areasymbols = NULL,
  mukeys = NULL,
  WHERE = NULL,
  query_string = FALSE,
  dsn = NULL
)
```

Arguments

areasymbols	vector of soil survey area symbols
mukeys	vector of map unit keys
WHERE	character containing SQL WHERE clause specified in terms of fields in legend, mapunit, or muaggatt tables, used in lieu of mukeys or areasymbols
query_string	Default: FALSE; if TRUE return a character string containing query that would be sent to SDA via SDA_query
dsn	Path to local SQLite database or a DBIConnection object. If NULL (default) use Soil Data Access API via SDA_query().

Value

a data.frame

Author(s)

Jason Nemecek, Chad Ferguson, Andrew Brown

---

get_SDA_pmgroupname *Get map unit parent material group information from Soil Data Access*

Description

Get map unit parent material group information from Soil Data Access

Usage

```
get_SDA_pmgroupname(
  areasymbols = NULL,
  mukeys = NULL,
  WHERE = NULL,
  method = "DOMINANT COMPONENT",
  simplify = TRUE,
  query_string = FALSE,
  dsn = NULL
)
```

Arguments

areasymbols	vector of soil survey area symbols
mukeys	vector of map unit keys
WHERE	character containing SQL WHERE clause specified in terms of fields in legend, mapunit, component, or copmgrp tables, used in lieu of mukeys or areasymbols
method	One of: "Dominant Component", "Dominant Condition", "None"
simplify	logical; group into generalized parent material groups? Default TRUE

query_string	Default: FALSE; if TRUE return a character string containing query that would be sent to SDA via SDA_query
dsn	Path to local SQLite database or a DBIConnection object. If NULL (default) use Soil Data Access API via SDA_query().

Details

Default method is "Dominant Component" to get the dominant component (highest percentage). Use "Dominant Condition" or dominant parent material condition (similar conditions aggregated across components). Use "None" for no aggregation (one record per component).

Value

a data.frame

Author(s)

Jason Nemecek, Chad Ferguson, Andrew Brown

---

get_SDA_property *Get map unit properties from Soil Data Access*

Description

Get map unit properties from Soil Data Access

Usage

```
get_SDA_property(
  property,
  method = c("Dominant Component (Category)", "Weighted Average", "Min/Max",
    "Dominant Component (Numeric)", "Dominant Condition", "None"),
  areasymbols = NULL,
  mukeys = NULL,
  WHERE = NULL,
  top_depth = 0,
  bottom_depth = 200,
  FUN = NULL,
  include_minors = FALSE,
  miscellaneous_areas = FALSE,
  query_string = FALSE,
  dsn = NULL
)
```

Arguments

property	character vector of labels from property dictionary tables (see details) OR physical column names from component or chorizon table.
method	one of: "Dominant Component (Category)", "Dominant Component (Numeric)", "Weighted Average", "MIN", "MAX", "Dominant Condition", or "None". If "None" is selected, the number of rows returned will depend on whether a component or horizon level property was selected, otherwise the result will be 1:1 with the number of map units.
areasymbols	vector of soil survey area symbols
mukeys	vector of map unit keys
WHERE	character containing SQL WHERE clause specified in terms of fields in legend or mapunit tables, used in lieu of mukeys or areasymbols. With aggregation method "NONE" the WHERE clause may additionally contain logic for columns from the component and chorizon table.
top_depth	Default: 0 (centimeters); a numeric value for upper boundary (top depth) used only for method="Weighted Average", "Dominant Component (Numeric)", and "MIN/MAX"
bottom_depth	Default: 200 (centimeters); a numeric value for lower boundary (bottom depth) used only for method="Weighted Average", "Dominant Component (Numeric)", and "MIN/MAX"
FUN	Optional: character representing SQL aggregation function either "MIN" or "MAX" used only for method="min/max"; this argument is calculated internally if you specify method="MIN" or method="MAX"
include_minors	Include minor components in "Weighted Average" or "MIN/MAX" results? Default: TRUE
miscellaneous_areas	Include miscellaneous areas (non-soil components) in results? Default: FALSE. Now works with all method types)
query_string	Default: FALSE; if TRUE return a character string containing query that would be sent to SDA via SDA_query
dsn	Path to local SQLite database or a DBIConnection object. If NULL (default) use Soil Data Access API via SDA_query().

Details

The property argument refers to one of the property names or columns specified in the tables below. Note that property can be specified as either a character vector of labeled properties, such as "Bulk Density 0.33 bar H2O - Rep Value", OR physical column names such as "dbthirdbar_r". To get "low" and "high" values for a particular property, replace the _r with _l or _h in the physical column name; for example property = c("dbthirdbar_l","dbthirdbar_r","dbthirdbar_h"). You can view exhaustive lists of component and component horizon level properties in the Soil Data Access "Tables and Columns Report".

Selected Component-level Properties:

Property (Component)	Column

Range Production - Favorable Year	rsprod_h
Range Production - Normal Year	rsprod_r
Range Production - Unfavorable Year	rsprod_l
Corrosion of Steel	corsteel
Corrosion of Concrete	corcon
Drainage Class	drainagecl
Hydrologic Group	hydgrp
Taxonomic Class Name	taxclname
Taxonomic Order	taxorder
Taxonomic Suborder	taxsuborder
Taxonomic Temperature Regime	taxtempregime
Wind Erodibility Group	weg
Wind Erodibility Index	wei
t Factor	tfact

Selected Horizon-level Properties:

Property (Horizon)	**Column**
0.1 bar H2O - Rep Value	wtenthbar_r
0.33 bar H2O - Rep Value	wthirdbar_r
15 bar H2O - Rep Value	wfifteenbar_r
Available Water Capacity - Rep Value	awc_r
Bray 1 Phosphate - Rep Value	pbray1_r
Bulk Density 0.1 bar H2O - Rep Value	dbtenthbar_r
Bulk Density 0.33 bar H2O - Rep Value	dbthirdbar_r
Bulk Density 15 bar H2O - Rep Value	dbfifteenbar_r
Bulk Density oven dry - Rep Value	dbovendry_r
CaCO3 Clay - Rep Value	claysizedcarb_r
Calcium Carbonate - Rep Value	caco3_r
Cation Exchange Capacity - Rep Value	cec7_r
Coarse Sand - Rep Value	sandco_r
Coarse Silt - Rep Value	siltco_r
Effective Cation Exchange Capacity - Rep Value	ecec_r
Electrial Conductivity 1:5 by volume - Rep Value	ec15_r
Electrical Conductivity - Rep Value	ec_r
Exchangeable Sodium Percentage - Rep Value	esp_r
Extract Aluminum - Rep Value	extral_r
Extractable Acidity - Rep Value	extracid_r
Fine Sand - Rep Value	sandfine_r
Fine Silt - Rep Value	siltfine_r
Free Iron - Rep Value	freeiron_r
Gypsum - Rep Value	gypsum_r
Kf	kffact
Ki	kifact
Kr	krfact
Kw	kwfact
LEP - Rep Value	lep_r
Liquid Limit - Rep Value	ll_r

Medium Sand - Rep Value	sandmed_r
Organic Matter - Rep Value	om_r
Oxalate Aluminum - Rep Value	aloxalate_r
Oxalate Iron - Rep Value	feoxalate_r
Oxalate Phosphate - Rep Value	poxalate_r
Plasticity Index - Rep Value	pi_r
Rock Fragments 3 - 10 inches - Rep Value	frag3to10_r
Rock Fragments > 10 inches - Rep Value	fraggt10_r
Rubbed Fiber % - Rep Value	fiberrubbedpct_r
Satiated H2O - Rep Value	wsatiated_r
Saturated Hydraulic Conductivity - Rep Value	ksat_r
Sodium Adsorption Ratio - Rep Value	sar_r
Sum of Bases - Rep Value	sumbases_r
Total Clay - Rep Value	claytotal_r
Total Phosphate - Rep Value	ptotal_r
Total Sand - Rep Value	sandtotal_r
Total Silt - Rep Value	silttotal_r
Unrubbed Fiber % - Rep Value	fiberunrubbedpct_r
Very Coarse Sand - Rep Value	sandvc_r
Very Fine Sand - Rep Value	sandvf_r
Water Soluble Phosphate - Rep Value	ph2osoluble_r
no. 10 sieve - Rep Value	sieveno10_r
no. 200 sieve - Rep Value	sieveno200_r
no. 4 sieve - Rep Value	sieveno4_r
no. 40 sieve - Rep Value	sieveno40_r
pH .01M CaCl2 - Rep Value	ph01mcacl2_r
pH 1:1 water - Rep Value	ph1to1h2o_r
pH Oxidized - Rep Value	phoxidized_r

Value

a data.frame with result

Author(s)

Jason Nemecek, Chad Ferguson, Andrew Brown

Examples

```
if(requireNamespace("curl") &
    curl::has_internet()) {

 # get 1/3 bar bulk density [0,25] centimeter depth weighted average from dominant component
 get_SDA_property(property = c("dbthirdbar_l","dbthirdbar_r","dbthirdbar_h"),
                  method = "Dominant Component (Numeric)",
                  areasymbols = "CA630",
                  top_depth = 0,
```

```
                    bottom_depth = 25)
}
```

`get_SDV_legend_elements` *Get Soil Data Viewer Attribute Information*

Description

Get Soil Data Viewer Attribute Information

Usage

```
get_SDV_legend_elements(
  WHERE,
  alpha = 255,
  notratedcolor = rgb(1, 1, 1, 0),
  simplify = TRUE
)
```

Arguments

`WHERE`	WHERE clause for query of Soil Data Access `sdvattribute` table
`alpha`	transparency value applied in calculation of hexadecimal color. Default: `255` (opaque).
`notratedcolor`	Used to add 'Not rated' color entries where applicable. Default: `"#FFFFFF00"` (transparent white).
`simplify`	Return a data.frame when `WHERE` is length 1? Return a list with 1 element per legend when `WHERE` is length > `1`? Default: `TRUE`

Value

A list with a data.frame element for each element of `where` containing `"attributekey"`, `"attributename"`, `"attributetype"`, `"attributetablename"`, `"attributecolumnname"`, `"attributedescription"`, `"nasisrulename"`, `"label"`, `"order"`, `"value"`, `"lower_value"`, `"upper_value"`,`"red"`, `"green"`, `"blue"` and `"hex"` columns.

`get_site_data_from_NASIS_db` *Get Site Data from a local NASIS Database*

Description

Get site-level data from a local NASIS database.

Usage

```
get_site_data_from_NASIS_db(
  SS = TRUE,
  nullFragsAreZero = TRUE,
  stringsAsFactors = NULL,
  dsn = NULL
)
```

Arguments

`SS`	fetch data from Selected Set in NASIS or from the entire local database (default: `TRUE`)
`nullFragsAreZero`	should surface fragment cover percentages of NULL be interpreted as 0? (default: TRUE)
`stringsAsFactors`	deprecated
`dsn`	Optional: path to local SQLite database containing NASIS table structure; default: `NULL`

Details

When multiple "site bedrock" entries are present, only the shallowest is returned by this function.

Value

A data.frame

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

`get_hz_data_from_NASIS_db`

get_site_data_from_pedon_db

Get Site Data from a PedonPC Database

Description

Get site-level data from a PedonPC database.

Usage

```
get_site_data_from_pedon_db(dsn)
```

Arguments

dsn	The path to a 'pedon.mdb' database.

Value

A data.frame.

Author(s)

Dylan E. Beaudette and Jay M. Skovlin

See Also

get_hz_data_from_pedon_db, get_veg_from_AK_Site,

get_soilseries_from_NASIS

Get records from the Series Classification (SC) database

Description

These functions return records from the Series Classification (SC) database, either from the local NASIS database (all series) or via web report (named series only).
get_competing_soilseries_from_NASIS(): Get Soil Series from NASIS Matching Taxonomic Class Name

Usage

```
get_soilseries_from_NASIS(
  stringsAsFactors = NULL,
  dsn = NULL,
  delimiter = " over "
)

get_soilseries_from_NASISWebReport(soils, stringsAsFactors = NULL)

get_competing_soilseries_from_NASIS(x, what = "taxclname", dsn = NULL)
```

Arguments

stringsAsFactors	deprecated
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL
delimiter	*character*. Used to collapse taxminalogy records where multiple values are used to describe strongly contrasting control sections. Default " over " creates combination mineralogy classes as they would be used in the family name.
soils	A vector of soil series names
x	Taxonomic Class Name (or other field specified by what) to match, use % for wildcard
what	Column name to match x against, default: 'taxclname'

Value

A data.frame

Author(s)

Stephen Roecker

---

get_text_notes_from_NASIS_db

Get text note data from a local NASIS Database

Description

Get text note data from a local NASIS Database

Usage

```
get_text_notes_from_NASIS_db(SS = TRUE, fixLineEndings = TRUE, dsn = NULL)

get_mutext_from_NASIS_db(SS = TRUE, fixLineEndings = TRUE, dsn = NULL)

get_cotext_from_NASIS_db(SS = TRUE, fixLineEndings = TRUE, dsn = NULL)
```

Arguments

SS	get data from the currently loaded Selected Set in NASIS or from the entire local database (default: TRUE)
fixLineEndings	convert line endings from \r\n to \n
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Value

A list with the results.

Author(s)

Dylan E. Beaudette and Jay M. Skovlin

See Also

get_hz_data_from_pedon_db, get_site_data_from_pedon_db

Examples

```
if(local_NASIS_defined()) {
 # query text note data
 t <- try(get_text_notes_from_NASIS_db())

 # show contents text note data, includes: siteobs, site, pedon, horizon level text notes data.
 str(t)

 # view text categories for site text notes
 if(!inherits(t, 'try-error')) {
  table(t$site_text$textcat)
 }
}
```

---

get_veg_data_from_NASIS_db

Get vegetation data from a local NASIS Database

Description

Get veg data from a local NASIS Database.

Usage

```
get_veg_data_from_NASIS_db(SS = TRUE, dsn = NULL)
```

Arguments

SS	get data from the currently loaded Selected Set in NASIS or from the entire local database (default: TRUE)
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Value

A list with the results.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

Examples

```
if(local_NASIS_defined()) {
 # query text note data
 v <- try(get_veg_from_NASIS_db())

 # show contents veg data returned
 str(v)
}
```

---

get_veg_from_AK_Site *Get Vegetation Data from an AK Site Database*

Description

Get Vegetation Data from an AK Site Database

Usage

```
get_veg_from_AK_Site(dsn)
```

Arguments

dsn	file path the the AK Site access database

Value

A data.frame with vegetation data in long format, linked to site ID.

Author(s)

Dylan E. Beaudette

See Also

get_hz_data_from_pedon_db, get_site_data_from_pedon_db

---

get_veg_from_MT_veg_db

Get Site and Plot-level Data from a Montana RangeDB database

Description

Get Site and Plot-level data from a Montana RangeDB database.

Usage

```
get_veg_from_MT_veg_db(dsn)
```

Arguments

dsn	The name of the Montana RangeDB front-end database connection (see details).

Value

A data.frame.

Author(s)

Jay M. Skovlin

See Also

get_veg_species_from_MT_veg_db, get_veg_other_from_MT_veg_db

get_veg_from_NPS_PLOTS_db

Get Vegetation Data from an NPS PLOTS Database

Description

Used to extract species, stratum, and cover vegetation data from a backend NPS PLOTS Database. Currently works for any Microsoft Access database with an .mdb file format.

Usage

```
get_veg_from_NPS_PLOTS_db(dsn)
```

Arguments

dsn	file path to the NPS PLOTS access database on your system.

Value

A data.frame with vegetation data in a long format with linkage to NRCS soil pedon data via the site_id key field.

Note

This function currently only works on Windows.

Author(s)

Jay M. Skovlin

get_veg_other_from_MT_veg_db

Get cover composition data from a Montana RangeDB database

Description

Get cover composition data from a Montana RangeDB database.

Usage

```
get_veg_other_from_MT_veg_db(dsn)
```

Arguments

dsn	The name of the Montana RangeDB front-end database connection (see details).

Value

A data.frame.

Author(s)

Jay M. Skovlin

See Also

get_veg_from_MT_veg_db, get_veg_species_from_MT_veg_db

`get_veg_species_from_MT_veg_db` *Get species-level Data from a Montana RangeDB database*

Description

Get species-level data from a Montana RangeDB database.

Usage

```
get_veg_species_from_MT_veg_db(dsn)
```

Arguments

dsn	The name of the Montana RangeDB front-end database connection (see details).

Value

A data.frame.

Author(s)

Jay M. Skovlin

See Also

get_veg_from_MT_veg_db, get_veg_other_from_MT_veg_db

`ISSR800.wcs` *Get 800m gridded soil properties from SoilWeb ISSR-800 Web Coverage Service (WCS)*

Description

Intermediate-scale gridded (800m) soil property and interpretation maps from aggregated SSURGO and STATSGO data. These maps were developed by USDA-NRCS-SPSD staff in collaboration with UCD-LAWR. Originally for educational use and interactive thematic maps, these data are a suitable alternative to gridded STATSGO-derived thematic soil maps. The full size grids can be downloaded here.

Usage

```
ISSR800.wcs(aoi, var, res = 800, quiet = FALSE)
```

Arguments

`aoi`	area of interest (AOI) defined using a `Spatial*`, `RasterLayer`, `sf`, `sfc` or `bbox` object, OR a `list`, see details
`var`	ISSR-800 grid name (case insensitive), see details
`res`	grid resolution, units of meters. The native resolution of ISSR-800 grids (this WCS) is 800m.
`quiet`	logical, passed to `curl::curl_download` to enable / suppress URL and progress bar for download.

Details

`aoi` should be specified as a `SpatRaster`, `Spatial*`, `RasterLayer`, `SpatRaster/SpatVector`, `sf`, `sfc`, or `bbox` object or a `list` containing:

`aoi` bounding-box specified as (xmin, ymin, xmax, ymax) e.g. c(-114.16, 47.65, -114.08, 47.68)

`crs` coordinate reference system of BBOX, e.g. 'OGC:CRS84' (EPSG:4326, WGS84 Longitude/Latitude)

The WCS query is parameterized using a rectangular extent derived from the above AOI specification, after conversion to the native CRS (EPSG:5070) of the ISSR-800 grids.

Variables available from this WCS can be queried using `WCS_details(wcs = 'ISSR800')`.

Value

A SpatRaster (or RasterLayer) object containing indexed map unit keys and associated raster attribute table or a try-error if request fails. By default, spatial classes from the `terra` package are returned. If the input object class is from the `raster` or `sp` packages a RasterLayer is returned.

Note

There are still some issues to be resolved related to the encoding of NA Variables with a natural zero (e.g. SAR) have 0 set to NA.

Author(s)

D.E. Beaudette and A.G. Brown

Examples

```
## Not run:
library(terra)

# see WCS_details() for variable options
WCS_details(wcs = 'ISSR800')

# get wind erodibility group
res <- ISSR800.wcs(list(aoi = c(-116, 35, -115.5, 35.5), crs = "EPSG:4326"),
                   var = 'weg', res = 800)
plot(res)

## End(Not run)
```

KSSL_VG_model *Develop a Water Retention Curve from KSSL Data*

Description

Water retention curve modeling via van Genuchten model and KSSL data.

Usage

```
KSSL_VG_model(VG_params, phi_min = 10^-6, phi_max = 10^8, pts = 100)
```

Arguments

VG_params	data.frame or list object with the parameters of the van Genuchten model, see details
phi_min	lower limit for water potential in kPa
phi_max	upper limit for water potential in kPa
pts	number of points to include in estimated water retention curve

Details

This function was developed to work with measured or estimated parameters of the van Genuchten model, as generated by the Rosetta model. As such, `VG_params` should have the following format and conventions:

theta_r saturated water content, values should be in the range of {0, 1}

theta_s residual water content, values should be in the range of {0, 1}

alpha related to the inverse of the air entry suction, function expects log10-transformed values with units of 1/cm

npar index of pore size distribution, function expects log10-transformed values (dimensionless)

Value

A list with the following components:

VG_curve estimated water retention curve: paired estimates of water potential (phi) and water content (theta)

VG_function spline function for converting water potential (phi, units of kPa) to estimated volumetric water content (theta, units of percent, range: {0, 1})

VG_inverse_function spline function for converting volumetric water content (theta, units of percent, range: {0, 1}) to estimated water potential (phi, units of kPa)

Note

A practical example is given in the fetchSCAN tutorial.

Author(s)

D.E. Beaudette

References

water retention curve estimation

Examples

```
# basic example
d <- data.frame(
  theta_r = 0.0337216,
  theta_s = 0.4864061,
  alpha = -1.581517,
  npar = 0.1227247
)

vg <- KSSL_VG_model(d)

str(vg)
```

loafercreek *Example* SoilProfilecollection *Objects Returned by* fetchNASIS.

Description

Several examples of soil profile collections returned by `fetchNASIS(from='pedons')` as `SoilProfileCollection` objects.

Examples

```
if(require("aqp")) {
# load example dataset
  data("gopheridge")

  # what kind of object is this?
  class(gopheridge)

  # how many profiles?
  length(gopheridge)

  # there are 60 profiles, this calls for a split plot
  par(mar=c(0,0,0,0), mfrow=c(2,1))

  # plot soil colors
  plot(gopheridge[1:30, ], name='hzname', color='soil_color')
  plot(gopheridge[31:60, ], name='hzname', color='soil_color')

  # need a larger top margin for legend
  par(mar=c(0,0,4,0), mfrow=c(2,1))
  # generate colors based on clay content
  plot(gopheridge[1:30, ], name='hzname', color='clay')
  plot(gopheridge[31:60, ], name='hzname', color='clay')

  # single row and no labels
  par(mar=c(0,0,0,0), mfrow=c(1,1))
  # plot soils sorted by depth to contact
  plot(gopheridge, name='', print.id=FALSE, plot.order=order(gopheridge$bedrckdepth))

  # plot first 10 profiles
 plot(gopheridge[1:10, ], name='hzname', color='soil_color', label='pedon_id', id.style='side')

  # add rock fragment data to plot:
  addVolumeFraction(gopheridge[1:10, ], colname='total_frags_pct')

  # add diagnostic horizons
 addDiagnosticBracket(gopheridge[1:10, ], kind='argillic horizon', col='red', offset=-0.4)

  ## loafercreek
```

```
 data("loafercreek")
 # plot first 10 profiles
plot(loafercreek[1:10, ], name='hzname', color='soil_color', label='pedon_id', id.style='side')

 # add rock fragment data to plot:
 addVolumeFraction(loafercreek[1:10, ], colname='total_frags_pct')

 # add diagnostic horizons
addDiagnosticBracket(loafercreek[1:10, ], kind='argillic horizon', col='red', offset=-0.4)
}
```

---

local_NASIS_defined *Check for presence of* nasis_local *ODBC data source*

Description

Check for presence of a NASIS data source. This function *always* returns FALSE when the odbc package is not available (regardless of whether you have an ODBC data source properly set up).

Usage

```
local_NASIS_defined(dsn = NULL)
```

Arguments

dsn	Optional: path to local SQLite database, or a DBIConnection, containing NASIS table structure; default: NULL

Details

If dsn is specified as a character vector it is assumed to refer to a SQLite data source. The result will be TRUE or FALSE depending on the result of RSQLite::dbCanConnect().

If dsn is specified as a DBIConnection the function returns the value of DBI::dbExistsTable("MetadataDomainMaster")

Value

logical

Examples

```
if(local_NASIS_defined()) {
  # use fetchNASIS or some other lower-level fetch function
} else {
  message('could not find `nasis_local` ODBC data source')
}
```

makeChunks *Generate chunk labels for splitting data*

Description

Generate chunk labels for splitting data

Usage

```
makeChunks(ids, size = 100)
```

Arguments

ids	vector of IDs
size	chunk (group) size

Value

A numeric vector

Examples

```
# split the lowercase alphabet into 2 chunks

aggregate(letters,
          by = list(makeChunks(letters, size=13)),
          FUN = paste0, collapse=",")
```

make_EDIT_service_URL *Make Ecological Dynamics Interpretive Tool (EDIT) web services URL*

Description

Construct a URL for Ecological Dynamics Interpretive Tool (EDIT) web services (https://edit.jornada.nmsu.edu/serv
to return PDF, TXT or JSON results.

Usage

```
make_EDIT_service_URL(
  src = c("descriptions", "downloads", "plant-community-tables", "models", "keys"),
  catalog = c("esd", "esg"),
  geoUnit = NULL,
  ecoclass = NULL,
  landuse = NULL,
  state = NULL,
  community = NULL,
  key = NULL,
  endpoint = NULL,
  querystring = NULL
)
```

Arguments

src	One of: descriptions, downloads, plant-community-tables, models, keys
catalog	Catalog ID. One of: esd or esg
geoUnit	Geographic unit ID. For example: 022A
ecoclass	Ecological class ID. For example: F022AX101CA
landuse	Optional: Used only for src = "plant-community-tables"
state	Optional: Used only for src = "plant-community-tables"
community	Optional: Used only for src = "plant-community-tables"
key	Optional: Key number. All keys will be returned if not specified.
endpoint	Optional: Specific endpoint e.g. overview.json, class-list.json, soil-features.json
querystring	Optional: Additional request parameters specified as a query string ?param1=value&param2=value.

Details

See the following official EDIT developer resources to see which endpoints are available for Ecological Site Description (ESD) or Ecological Site Group (ESG) catalogs:

- https://edit.jornada.nmsu.edu/resources/esd
- https://edit.jornada.nmsu.edu/resources/esg

Value

A character vector containing URLs with specified parameters. This function is vectorized.

See Also

get_EDIT_ecoclass_by_geoUnit

Examples

```
# url for all geoUnit keys as PDF
make_EDIT_service_URL(src = "descriptions",
                      catalog = "esd",
                      geoUnit = "039X")

# url for a single key within geoUnit as PDF
make_EDIT_service_URL(src = "descriptions",
                      catalog = "esd",
                      geoUnit = "039X",
                      key = "1")

# query for "full" description in JSON
desc <-  make_EDIT_service_URL(src = "descriptions",
                               catalog = "esd",
                               geoUnit = "039X",
                               endpoint = "R039XA109AZ.json")

# query for "overview"
desc_ov <- make_EDIT_service_URL(src = "descriptions",
                                 catalog = "esd",
                                 geoUnit = "039X",
                                 ecoclass = "R039XA109AZ",
                                 endpoint = "overview.json")

# query for specific section, e.g. "water features"
desc_wf <- make_EDIT_service_URL(src = "descriptions",
                                 catalog = "esd",
                                 geoUnit = "039X",
                                 ecoclass = "R039XA109AZ",
                                 endpoint = "water-features.json")

# construct the URLs -- that is a query essentially
# then download the result with read_json

#full <- jsonlite::read_json(desc)
#overview <- jsonlite::read_json(desc_ov)
#waterfeature <- jsonlite::read_json(desc_wf)
```

`metadata` *NASIS 7 Metadata*

Description

NASIS 7 Metadata from MetadataDomainDetail, MetadataDomainMaster, and MetadataTableColumn tables

Format

A data.frame with the following columns:

- DomainID - Integer. ID that uniquely identifies a domain in a data model, not just within a database.
- DomainName - Character. Domain Name.
- DomainRanked - Integer. Is domain ranked? 0 = No; 1 = Yes
- DisplayLabel - Character. Domain Display Label.
- ChoiceSequence - Integer. Order or sequence of Choices.
- ChoiceValue - Integer. Value of choice level.
- ChoiceName - Character. Name of choice level.
- ChoiceLabel - Character. Label of choice level.
- ChoiceObsolete - Integer. Is choice level obsolete? 0 = No; 1 = Yes
- ColumnPhysicalName - Character. Physical column name.
- ColumnLogicalName - Character. Logical column name.

---

mukey.wcs *Get gNATSGO / gSSURGO Map Unit Key (*mukey*) grid from SoilWeb Web Coverage Service (WCS)*

Description

Download chunks of the gNATSGO or gSSURGO map unit key grid via bounding-box from the SoilWeb WCS.

Usage

```
mukey.wcs(aoi, db = c("gNATSGO", "gSSURGO", "RSS"), res = 30, quiet = FALSE)
```

Arguments

aoi	area of interest (AOI) defined using either a Spatial*, RasterLayer, sf, sfc or bbox object, or a list, see details
db	name of the gridded map unit key grid to access, should be either 'gNATSGO' or 'gSSURGO' (case insensitive)
res	grid resolution, units of meters. The native resolution of gNATSGO and gSSURGO (this WCS) is 30m; and Raster Soil Surveys (RSS) are at 10m resolution. If res is not specified the native resolution of the source is used.
quiet	logical, passed to curl::curl_download to enable / suppress URL and progress bar for download.

Details

`aoi` should be specified as one of: `SpatRaster`, `Spatial*`, `RasterLayer`, `sf`, `sfc`, `bbox` object, OR a `list` containing:

`aoi` bounding-box specified as (xmin, ymin, xmax, ymax) e.g. c(-114.16, 47.65, -114.08, 47.68)

`crs` coordinate reference system of BBOX, e.g. 'OGC:CRS84' (EPSG:4326, WGS84 Longitude/Latitude)

The WCS query is parameterized using a rectangular extent derived from the above AOI specification, after conversion to the native CRS (EPSG:5070) of the gNATSGO / gSSURGO grid.

Databases available from this WCS can be queried using `WCS_details(wcs = 'mukey')`.

Value

A SpatRaster (or RasterLayer) object containing indexed map unit keys and associated raster attribute table or a try-error if request fails. By default, spatial classes from the `terra` package are returned. If the input object class is from the `raster` or `sp` packages a RasterLayer is returned.

Note

The gNATSGO grid includes raster soil survey map unit keys which are not in SDA.

Author(s)

D.E. Beaudette and A.G. Brown

Examples

```
## Not run:
library(terra)

res <- mukey.wcs(list(aoi = c(-116.7400, 35.2904, -116.7072, 35.3026), crs = "EPSG:4326"),
                 db = 'gNATSGO', res = 30)

MUKEY <- unique(values(res))

prp <- setNames(
  get_SDA_property(
    c("ph1to1h2o_r", "claytotal_r"),
    "weighted average",
    mukeys = MUKEY,
    top_depth = 0,
    bottom_depth = 25,
    include_minors = TRUE,
    miscellaneous_areas = FALSE
  )[, c("mukey", "ph1to1h2o_r", "claytotal_r")],
  c("ID",    "pH1to1_0to25", "clay_0to25")
)

levels(res) <- prp
res2 <- catalyze(res)
```

```
res2

plot(res2[['pH1to1_0to25']])

## End(Not run)
```

NASISChoiceList *Work with NASIS Choice Lists*

Description

Create (ordered) factors and interchange between choice names, values and labels for lists of input vectors.

Usage

```
NASISChoiceList(
  x,
  colnames = names(x),
  what = "ColumnPhysicalName",
  choice = c("ChoiceName", "ChoiceValue", "ChoiceLabel"),
  obsolete = FALSE,
  factor = TRUE,
  droplevels = FALSE,
  ordered = TRUE,
  simplify = TRUE,
  dsn = NULL
)
```

Arguments

x	A named list of vectors to use as input for NASIS Choice List lookup
colnames	vector of values of the column specified by what. E.g. colnames="texcl" for what="ColumnPhysicalName". Default: names(x) (if x is named)
what	passed to get_NASIS_column_metadata(); Column to match x against. Default "ColumnPhysicalName"; alternate options include "DomainID", "DomainName", "DomainRanked", "DisplayLabel", "ChoiceSequence", "ChoiceValue", "ChoiceName", "ChoiceLabel", "ChoiceObsolete", "ChoiceDescription", "ColumnLogicalName"
choice	one of: "ChoiceName", "ChoiceValue", or "ChoiceLabel"
obsolete	Include "obsolete" choices? Default: FALSE
factor	Convert result to factor? Default: TRUE
droplevels	Drop unused factor levels? Default: TRUE (used only when factor=TRUE)
ordered	Should the result be an ordered factor? Default: TRUE (use *only* if DomainRanked is true for all choices)
simplify	Should list result with length 1 be reduced to a single vector? Default: TRUE
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Value

A list of "choices" based on the input `x` that have been converted to a consistent target set of levels (specified by `choice`) via NASIS 7 metadata.

When `factor=TRUE` the result is a factor, possibly ordered when `ordered=TRUE` and the target domain is a "ranked" domain (i.e. `ChoiceSequence` has logical meaning).

When `factor=FALSE` the result is a character or numeric vector. Numeric vectors are always returned when `choice` is `"ChoiceValue"`.

Examples

```
NASISChoiceList(1:3, "texcl")

NASISChoiceList(1:3, "pondfreqcl")

NASISChoiceList("Clay loam", "texcl", choice = "ChoiceValue")

NASISChoiceList("Silty clay loam", "texcl", choice = "ChoiceName")
```

NASISDomainsAsFactor *Get/Set Options for Encoding NASIS Domains as Factors*

Description

Set package option `soilDB.NASIS.DomainsAsFactor` for returning coded NASIS domains as factors.

Usage

```
NASISDomainsAsFactor(x = NULL)
```

Arguments

`x`	logical; default `FALSE`

Value

logical, result of `getOption("soilDB.NASIS.DomainsAsFactor")`

Examples

```
## Not run:
NASISDomansAsFactor(TRUE)

## End(Not run)
```

NASIS_table_column_keys *NASIS 7 Tables, Columns and Foreign Keys*

Description

This dataset contains NASIS 7 Tables, Columns and Foreign Keys

OSDquery *Search full text of Official Series Description on SoilWeb*

Description

This is the R interface to OSD search by Section and OSD Search APIs provided by SoilWeb.

OSD records are searched with the PostgreSQL fulltext indexing and query system (syntax details). Each search field (except for the "brief narrative" and MLRA) corresponds with a section header in an OSD. The results may not include every OSD due to formatting errors and typos. Results are scored based on the number of times search terms match words in associated sections.

Usage

```
OSDquery(
  everything = NULL,
  mlra = "",
  taxonomic_class = "",
  typical_pedon = "",
  brief_narrative = "",
  ric = "",
  use_and_veg = "",
  competing_series = "",
  geog_location = "",
  geog_assoc_soils = ""
)
```

Arguments

everything	search entire OSD text (default is NULL), mlra may also be specified, all other arguments are ignored
mlra	a comma-delimited string of MLRA to search ('17,18,22A')
taxonomic_class	search family level classification
typical_pedon	search typical pedon section
brief_narrative	search brief narrative

`ric`	search range in characteristics section
`use_and_veg`	search use and vegetation section
`competing_series`	search competing series section
`geog_location`	search geographic setting section
`geog_assoc_soils`	search geographically associated soils section

Details

See this webpage for more information.

- family level taxa are derived from SC database, not parsed OSD records
- MLRA are derived via spatial intersection (SSURGO x MLRA polygons)
- MLRA-filtering is only possible for series used in the current SSURGO snapshot (component name)
- logical AND: &
- logical OR: |
- wildcard, e.g. rhy-something rhy:*
- search terms with spaces need doubled single quotes: ”san joaquin”
- combine search terms into a single expression: (grano:* | granite)

Related documentation can be found in the following tutorials

- overview of all soil series query functions
- competing soil series
- siblings

Value

a `data.frame` object containing soil series names that match patterns supplied as arguments.

Note

SoilWeb maintains a snapshot of the Official Series Description data.

Author(s)

D.E. Beaudette

References

https://www.nrcs.usda.gov/wps/portal/nrcs/detailfull/soils/home/?cid=nrcs142p2_053587

See Also

`fetchOSD`, `siblings`, `fetchOSD`

Examples

```
if(requireNamespace("curl") &
   curl::has_internet() &
   require(aqp)) {

  # find all series that list Pardee as a geographically associated soil.
  s <- OSDquery(geog_assoc_soils = 'pardee')

  # get data for these series
  x <- fetchOSD(s$series, extended = TRUE, colorState = 'dry')

  # simple figure
  par(mar=c(0,0,1,1))
  plot(x$SPC)
}
```

`parseWebReport` *Parse contents of a web report, based on supplied arguments.*

Description

Parse contents of a web report, based on supplied arguments.

Usage

```
parseWebReport(url, args, index = 1)
```

Arguments

`url`	Base URL to a LIMS/NASIS web report.
`args`	List of named arguments to send to report, see details.
`index`	Integer index specifying the table to return, or, NULL for a list of tables

Details

Report argument names can be inferred by inspection of the HTML source associated with any given web report.

Value

A `data.frame` object in the case of a single integer passed to `index`, a `list` object in the case of an integer vector or NULL passed to `index`.

Note

Most web reports are for internal use only.

Author(s)

D.E. Beaudette and S.M. Roecker

Examples

```
# pending
```

`processSDA_WKT` *Post-process Well-Known Text from Soil Data Access*

Description

This is a helper function commonly used with `SDA_query` to extract WKT (well-known text) representation of geometry to an `sf` or `sp` object.

Usage

```
processSDA_WKT(d, g = "geom", crs = 4326, p4s = NULL, as_sf = TRUE)
```

Arguments

`d`	`data.frame` returned by `SDA_query`, containing WKT representation of geometry
`g`	name of column in `d` containing WKT geometry
`crs`	CRS definition (e.g. an EPSG code). Default `4326` for WGS84 Geographic Coordinate System
`p4s`	Deprecated: PROJ4 CRS definition
`as_sf`	Return an `sf` `data.frame`? If `FALSE` return a `Spatial*` object.

Details

The SDA website can be found at https://sdmdataaccess.nrcs.usda.gov. See the SDA Tutorial for detailed examples.

The SDA website can be found at https://sdmdataaccess.nrcs.usda.gov. See the SDA Tutorial for detailed examples.

Value

An `sf` object or if `as_sf` is `FALSE` a `Spatial*` object.

Note

This function requires the `sf` package.

Author(s)

D.E. Beaudette, A.G. Brown

`ROSETTA` *Query USDA-ARS ROSETTA Model API*

Description

A simple interface to the ROSETTA model for predicting hydraulic parameters from soil properties. The ROSETTA API was developed by Dr. Todd Skaggs (USDA-ARS) and links to the work of Zhang and Schaap, (2017). See the related tutorial for additional examples.

Usage

```
ROSETTA(
  x,
  vars,
  v = c("1", "2", "3"),
  include.sd = FALSE,
  chunkSize = 10000,
  conf = NULL
)
```

Arguments

`x`	a `data.frame` of required soil properties, may contain other columns, see details
`vars`	character vector of column names in `x` containing relevant soil property values, see details
`v`	ROSETTA model version number: '1', '2', or '3', see details and references.
`include.sd`	logical, include bootstrap standard deviation for estimated parameters
`chunkSize`	number of records per API call
`conf`	configuration passed to `httr::POST()` such as `verbose()`.

Details

Soil properties supplied in `x` must be described, in order, via `vars` argument. The API does not use the names but column ordering must follow: sand, silt, clay, bulk density, volumetric water content at 33kPa (1/3 bar), and volumetric water content at 1500kPa (15 bar).

The ROSETTA model relies on a minimum of 3 soil properties, with increasing (expected) accuracy as additional properties are included:

- required, sand, silt, clay: USDA soil texture separates (percentages) that sum to 100\

- optional, bulk density (any moisture basis): mass per volume after accounting for >2mm fragments, units of gm/cm3
- optional, volumetric water content at 33 kPa: roughly "field capacity" for most soils, units of cm^3/cm^3
- optional, volumetric water content at 1500 kPa: roughly "permanent wilting point" for most plants, units of cm^3/cm^3

The Rosetta pedotransfer function predicts five parameters for the van Genuchten model of unsaturated soil hydraulic properties

- theta_r : residual volumetric water content
- theta_s : saturated volumetric water content
- log10(alpha) : retention shape parameter `[log10(1/cm)]`
- log10(npar) : retention shape parameter
- log10(ksat) : saturated hydraulic conductivity `[log10(cm/d)]`

Column names not specified in `vars` are retained in the output.

Three versions of the ROSETTA model are available, selected using `v = 1`, `v = 2`, or `v = 3`.

- version 1 - Schaap, M.G., F.J. Leij, and M.Th. van Genuchten. 2001. ROSETTA: a computer program for estimating soil hydraulic parameters with hierarchical pedotransfer functions. Journal of Hydrology 251(3-4): 163-176. doi: doi:10.1016/S00221694(01)004668.
- version 2 - Schaap, M.G., A. Nemes, and M.T. van Genuchten. 2004. Comparison of Models for Indirect Estimation of Water Retention and Available Water in Surface Soils. Vadose Zone Journal 3(4): 1455-1463. doi: doi:10.2136/vzj2004.1455.
- version 3 - Zhang, Y., and M.G. Schaap. 2017. Weighted recalibration of the Rosetta pedotransfer model with improved estimates of hydraulic parameter distributions and summary statistics (Rosetta3). Journal of Hydrology 547: 39-53. doi: doi:10.1016/j.jhydrol.2017.01.004.

Author(s)

D.E. Beaudette, Todd Skaggs (ARS), Richard Reid

References

Consider using the interactive version, with copy/paste functionality at: `https://www.handbook60.org/rosetta`.

Rosetta Model Home Page: `https://www.ars.usda.gov/pacific-west-area/riverside-ca/agricultural-water-efficiency-and-salinity-research-unit/docs/model/rosetta-model/`.

Python ROSETTA model: `http://www.u.arizona.edu/~ygzhang/download.html`.

Yonggen Zhang, Marcel G. Schaap. 2017. Weighted recalibration of the Rosetta pedotransfer model with improved estimates of hydraulic parameter distributions and summary statistics (Rosetta3). Journal of Hydrology. 547: 39-53. doi:10.1016/j.jhydrol.2017.01.004.

Kosugi, K. 1999. General model for unsaturated hydraulic conductivity for soils with lognormal pore-size distribution. Soil Sci. Soc. Am. J. 63:270-277.

Mualem, Y. 1976. A new model predicting the hydraulic conductivity of unsaturated porous media. Water Resour. Res. 12:513-522.

Schaap, M.G. and W. Bouten. 1996. Modeling water retention curves of sandy soils using neural networks. Water Resour. Res. 32:3033-3040.

Schaap, M.G., Leij F.J. and van Genuchten M.Th. 1998. Neural network analysis for hierarchical prediction of soil water retention and saturated hydraulic conductivity. Soil Sci. Soc. Am. J. 62:847-855.

Schaap, M.G., and F.J. Leij, 1998. Database Related Accuracy and Uncertainty of Pedotransfer Functions, Soil Science 163:765-779.

Schaap, M.G., F.J. Leij and M. Th. van Genuchten. 1999. A bootstrap-neural network approach to predict soil hydraulic parameters. In: van Genuchten, M.Th., F.J. Leij, and L. Wu (eds), Proc. Int. Workshop, Characterization and Measurements of the Hydraulic Properties of Unsaturated Porous Media, pp 1237-1250, University of California, Riverside, CA.

Schaap, M.G., F.J. Leij, 1999, Improved prediction of unsaturated hydraulic conductivity with the Mualem-van Genuchten, Submitted to Soil Sci. Soc. Am. J.

van Genuchten, M.Th. 1980. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. Soil Sci. Am. J. 44:892-898.

Schaap, M.G., F.J. Leij, and M.Th. van Genuchten. 2001. ROSETTA: a computer program for estimating soil hydraulic parameters with hierarchical pedotransfer functions. Journal of Hydrology 251(3-4): 163-176. doi: doi:10.1016/S00221694(01)004668.

Schaap, M.G., A. Nemes, and M.T. van Genuchten. 2004. Comparison of Models for Indirect Estimation of Water Retention and Available Water in Surface Soils. Vadose Zone Journal 3(4): 1455-1463. doi: doi:10.2136/vzj2004.1455.

Zhang, Y., and M.G. Schaap. 2017. Weighted recalibration of the Rosetta pedotransfer model with improved estimates of hydraulic parameter distributions and summary statistics (Rosetta3). Journal of Hydrology 547: 39-53. doi: doi:10.1016/j.jhydrol.2017.01.004.

`SCAN_SNOTEL_metadata` *Get SCAN and SNOTEL Station Metadata*

Description

These data have been compiled from several sources and represent a progressive effort to organize SCAN/SNOTEL station metadata. Therefore, some records may be missing or incorrect.

Format

A data frame with 1092 observations on the following 12 variables.

list("Name") station name

list("Site") station ID

list("State") state

list("Network") sensor network: SCAN / SNOTEL

list("County") county

list("Elevation_ft") station elevation in feet

list("Latitude") latitude of station

list("Longitude") longitude of station

list("HUC") associated watershed

list("climstanm") climate station name (TODO: remove this column)

list("upedonid") associated user pedon ID

list("pedlabsampnum") associated lab sample ID

`SDA_query` *Query Soil Data Access*

Description

Submit a query to the Soil Data Access (SDA) REST/JSON web-service and return the results as a data.frame. There is a 100,000 record limit and 32Mb JSON serializer limit, per query. Queries should contain a WHERE statement or JOIN condition to limit the number of rows affected / returned. Consider wrapping calls to `SDA_query` in a function that can iterate over logical chunks (e.g. areasymbol, mukey, cokey, etc.). The function `makeChunks` can help with such iteration.

Usage

```
SDA_query(q)
```

Arguments

`q`	A valid T-SQL query surrounded by double quotes

Details

The SDA website can be found at https://sdmdataaccess.nrcs.usda.gov and query examples can be found at https://sdmdataaccess.nrcs.usda.gov/QueryHelp.aspx. A library of query examples can be found at https://nasis.sc.egov.usda.gov/NasisReportsWebSite/limsreport.aspx?report_name=SDA-SQL_Library_Home.

SSURGO (detailed soil survey) and STATSGO (generalized soil survey) data are stored together within SDA. This means that queries that don't specify an area symbol may result in a mixture of SSURGO and STATSGO records. See the examples below and the SDA Tutorial for details.

Value

a data.frame result (`NULL` if empty, try-error on error)

Note

This function requires the `httr`, `jsonlite`, and `xml2` packages

Author(s)

D.E. Beaudette

See Also

SDA_spatialQuery

Examples

```
if(requireNamespace("curl") & requireNamespace("wk") &
   curl::has_internet()) {

  ## get SSURGO export date for all soil survey areas in California
  # there is no need to filter STATSGO
  # because we are filtering on SSURGO area symbols
  q <- "SELECT areasymbol, saverest FROM sacatalog WHERE areasymbol LIKE 'CA%';"
  x <- SDA_query(q)
  head(x)


  ## get SSURGO component data associated with the
  ## Amador series / major component only
  # this query must explicitly filter out STATSGO data
  q <- "SELECT cokey, compname, comppct_r FROM legend
    INNER JOIN mapunit mu ON mu.lkey = legend.lkey
    INNER JOIN component co ON mu.mukey = co.mukey
    WHERE legend.areasymbol != 'US' AND compname = 'Amador';"

  res <- SDA_query(q)
  str(res)


  ## get component-level data for a specific soil survey area (Yolo county, CA)
  # there is no need to filter STATSGO because the query contains
  # an implicit selection of SSURGO data by areasymbol
  q <- "SELECT
    component.mukey, cokey, comppct_r, compname, taxclname,
    taxorder, taxsuborder, taxgrtgroup, taxsubgrp
    FROM legend
    INNER JOIN mapunit ON mapunit.lkey = legend.lkey
    LEFT OUTER JOIN component ON component.mukey = mapunit.mukey
    WHERE legend.areasymbol = 'CA113' ;"

  res <- SDA_query(q)
  str(res)

  ## get tabular data based on result from spatial query
  # there is no need to filter STATSGO because
  # SDA_Get_Mukey_from_intersection_with_WktWgs84() implies SSURGO
  p <- wk::as_wkt(wk::rct(-120.9, 37.7, -120.8, 37.8))
  q <- paste0("SELECT mukey, cokey, compname, comppct_r FROM component
```

```
    WHERE mukey IN (SELECT DISTINCT mukey FROM
    SDA_Get_Mukey_from_intersection_with_WktWgs84('", p,
     "')) ORDER BY mukey, cokey, comppct_r DESC")

  x <- SDA_query(q)
  str(x)
}
```

SDA_spatialQuery *Query Soil Data Access by spatial intersection with supplied geometry*

Description

Query SDA (SSURGO / STATSGO) records via spatial intersection with supplied geometries. Input can be SpatialPoints, SpatialLines, or SpatialPolygons objects with a valid CRS. Map unit keys, overlapping polygons, or the spatial intersection of geom + SSURGO / STATSGO polygons can be returned. See details.

Usage

```
SDA_spatialQuery(
  geom,
  what = "mukey",
  geomIntersection = FALSE,
  db = c("SSURGO", "STATSGO", "SAPOLYGON"),
  byFeature = FALSE,
  idcol = "gid",
  query_string = FALSE,
  as_Spatial = getOption("soilDB.return_Spatial", default = FALSE)
)
```

Arguments

geom	an sf or Spatial* object, with valid CRS. May contain multiple features.
what	a character vector specifying what to return. 'mukey': data.frame with intersecting map unit keys and names, 'mupolygon' overlapping or intersecting map unit polygons from selected database, 'areasymbol': data.frame with intersecting soil survey areas, 'sapolygon': overlapping or intersecting soil survey area polygons (SSURGO only)
geomIntersection	logical; FALSE: overlapping map unit polygons returned, TRUE: intersection of geom + map unit polygons is returned.
db	a character vector identifying the Soil Geographic Databases ('SSURGO' or 'STATSGO') to query. Option *STATSGO* works with what = "mukey" and what = "mupolygon".
byFeature	Iterate over features, returning a combined data.frame where each feature is uniquely identified by value in idcol. Default FALSE.

idcol	Unique IDs used for individual features when byFeature = TRUE; Default "gid"
query_string	Default: FALSE; if TRUE return a character string containing query that would be sent to SDA via SDA_query
as_Spatial	Return sp classes? e.g. Spatial*DataFrame. Default: FALSE.

Details

Queries for map unit keys are always more efficient vs. queries for overlapping or intersecting (i.e. least efficient) features. geom is converted to GCS / WGS84 as needed. Map unit keys are always returned when using what = "mupolygon".

SSURGO (detailed soil survey, typically 1:24,000 scale) and STATSGO (generalized soil survey, 1:250,000 scale) data are stored together within SDA. This means that queries that don't specify an area symbol may result in a mixture of SSURGO and STATSGO records. See the examples below and the SDA Tutorial for details.

Value

A data.frame if what = 'mukey', otherwise an sf object. A try-error in the event the request cannot be made or if there is an error in the query.

Note

Row-order is not preserved across features in geom and returned object. Use byFeature argument to iterate over features and return results that are 1:1 with the inputs. Polygon area in acres is computed server-side when what = 'mupolygon' and geomIntersection = TRUE.

Author(s)

D.E. Beaudette, A.G. Brown, D.R. Schlaepfer

See Also

SDA_query

Examples

```
## Not run:
  if (requireNamespace("aqp") && requireNamespace("sf")) {

    library(aqp)
    library(sf)

    ## query at a point

    # example point
    p <- sf::st_as_sf(data.frame(x = -119.72330,
                                 y = 36.92204),
                      coords = c('x', 'y'),
                      crs = 4326)
```

```
# query map unit records at this point
res <- SDA_spatialQuery(p, what = 'mukey')

# convert results into an SQL "IN" statement
# useful when there are multiple intersecting records
mu.is <- format_SQL_in_statement(res$mukey)

# composite SQL WHERE clause
sql <- sprintf("mukey IN %s", mu.is)

# get commonly used map unit / component / chorizon records
# as a SoilProfileCollection object
# request that results contain `mukey` with `duplicates = TRUE`
x <- fetchSDA(sql, duplicates = TRUE)

# safely set texture class factor levels
# by making a copy of this column
# this will save in lieu of textures in the original
# `texture` column
horizons(x)$texture.class <- factor(x$texture, levels = SoilTextureLevels())

# graphical depiction of the result
plotSPC(x,
        color = 'texture.class',
        label = 'compname',
        name = 'hzname',
        cex.names = 1,
        width = 0.25,
        plot.depth.axis = FALSE,
        hz.depths = TRUE,
        name.style = 'center-center')

## query mukey + geometry that intersect with a bounding box

# define a bounding box: xmin, xmax, ymin, ymax
#
#         +-------------------(ymax, xmax)
#         |                         |
#         |                         |
#     (ymin, xmin) -----------------+
b <- c(-119.747629, -119.67935, 36.912019, 36.944987)

# convert bounding box to WKT
bbox.sp <- sf::st_as_sf(wk::rct(
  xmin = b[1], xmax = b[2], ymin = b[3], ymax = b[4],
  crs = sf::st_crs(4326)
))

# results contain associated map unit keys (mukey)
# return SSURGO polygons, after intersection with provided BBOX
ssurgo.geom <- SDA_spatialQuery(
  bbox.sp,
  what = 'mupolygon',
```

```
  db = 'SSURGO',
  geomIntersection = TRUE
)

# return STATSGO polygons, after intersection with provided BBOX
statsgo.geom <- SDA_spatialQuery(
  bbox.sp,
  what = 'mupolygon',
  db = 'STATSGO',
  geomIntersection = TRUE
)

# inspect results
par(mar = c(0,0,3,1))
plot(sf::st_geometry(ssurgo.geom), border = 'royalblue')
plot(sf::st_geometry(statsgo.geom), lwd = 2, border = 'firebrick', add = TRUE)
plot(sf::st_geometry(bbox.sp), lwd = 3, add = TRUE)
legend(
  x = 'topright',
  legend = c('BBOX', 'STATSGO', 'SSURGO'),
  lwd = c(3, 2, 1),
  col = c('black', 'firebrick', 'royalblue'),
)

# quick reminder that STATSGO map units often contain many components
# format an SQL IN statement using the first STATSGO mukey
mu.is <- format_SQL_in_statement(statsgo.geom$mukey[1])

# composite SQL WHERE clause
sql <- sprintf("mukey IN %s", mu.is)

# get commonly used map unit / component / chorizon records
# as a SoilProfileCollection object
x <- fetchSDA(sql)

# tighter figure margins
par(mar = c(0,0,3,1))

# organize component sketches by national map unit symbol
# color horizons via awc
# adjust legend title
# add alternate label (vertical text) containing component percent
# move horizon names into the profile sketches
# make profiles wider
aqp::groupedProfilePlot(x,
                        groups = 'nationalmusym',
                        label = 'compname',
                        color = 'awc_r',
                        col.label = 'Available Water Holding Capacity (cm / cm)',
                        alt.label = 'comppct_r',
                        name.style = 'center-center',
                        width = 0.3
)
```

```
    mtext(
      'STATSGO (1:250,000) map units contain a lot of components!',
      side = 1,
      adj = 0,
      line = -1.5,
      at = 0.25,
      font = 4
    )
  }

## End(Not run)
```

`seriesExtent` *Retrieve Soil Series Extent Maps from SoilWeb*

Description

This function downloads a generalized representations of a soil series extent from SoilWeb, derived from the current SSURGO snapshot. Data can be returned as vector outlines (`sf` object) or gridded representation of area proportion falling within 800m cells (`SpatRaster` object). Gridded series extent data are only available in CONUS. Vector representations are returned with a GCS/WGS84 coordinate reference system and raster representations are returned with an Albers Equal Area / NAD83 coordinate reference system (`EPSG:5070`).

Usage

```
seriesExtent(
  s,
  type = c("vector", "raster"),
  timeout = 60,
  as_Spatial = getOption("soilDB.return_Spatial", default = FALSE)
)
```

Arguments

`s`	a soil series name, case-insensitive
`type`	series extent representation, `'vector'`: results in an `sf` object and `'raster'` results in a `SpatRaster` object
`timeout`	time that we are willing to wait for a response, in seconds
`as_Spatial`	Return sp (`SpatialPolygonsDataFrame`) / raster (`RasterLayer`) classes? Default: `FALSE`.

Value

An R spatial object, class depending on `type` and `as_Spatial` arguments

Author(s)

D.E. Beaudette

References

https://casoilresource.lawr.ucdavis.edu/see/

Examples

```
if(requireNamespace("curl") &
   requireNamespace("sf") &
   requireNamespace("terra") &
   curl::has_internet()) {

  # required packages
  library(sf)
  library(terra)

  # specify a soil series name
  s <- 'magnor'

  # return an sf object
  x <- seriesExtent(s, type = 'vector')

  # return a terra SpatRasters
  y <- seriesExtent(s, type = 'raster')

  if (!is.null(x) && !is.null(y)) {
    # note that CRS are different
    sf::st_crs(x)
    terra::crs(y)

    # transform vector representation to CRS of raster
    x <- sf::st_transform(x, terra::crs(y))

    # graphical comparison
    par(mar = c(1, 1 , 1, 3))
    plot(y, axes = FALSE)

    # no fill color
    plot(x['series'], add = TRUE, col = NA)
  }

}
```

`siblings` *Get "siblings" and "cousins" for a given soil series*

Description

Look up siblings and cousins for a given soil series from the current fiscal year SSURGO snapshot via SoilWeb.

The siblings of any given soil series are defined as those soil components (major and minor) that share a parent map unit with the named series (as a major component). Component names are filtered using a snapshot of the Soil Classification database to ensure that only valid soil series names are included. Cousins are siblings of siblings. Data are sourced from SoilWeb which maintains a copy of the current SSURGO snapshot. Visualizations of soil "siblings"-related concepts can be found in the "Sibling Summary" tab of Soil Data Explorer app: https://casoilresource.lawr.ucdavis.edu/sde/.

Additional resources:

- Soil Series Query Functions
- Soil "Siblings" Tutorial
- SSSA 2019 Presentation - Mapping Soilscapes Using Soil Co-Occurrence Networks

Usage

```
siblings(s, only.major = FALSE, component.data = FALSE, cousins = FALSE)
```

Arguments

`s`	character vector, the name of a single soil series, case-insensitive.
`only.major`	logical, should only return siblings that are major components
`component.data`	logical, should component data for siblings (and optionally cousins) be returned?
`cousins`	logical, should siblings-of-siblings (cousins) be returned?

Value

A `list` containing:

- sib: `data.frame` containing siblings, major component flag, and number of co-occurrences
- sib.data: `data.frame` containing sibling component data (only when `component.data = TRUE`)
- cousins: `data.frame` containing cousins, major component flag, and number of co-occurrences (only when `cousins = TRUE`)
- cousin.data: `data.frame` containing cousin component data (only when `cousins = TRUE, component.data = TRUE`)

Author(s)

D.E. Beaudette

References

O'Geen, A., Walkinshaw, M. and Beaudette, D. (2017), SoilWeb: A Multifaceted Interface to Soil Survey Information. Soil Science Society of America Journal, 81: 853-862. doi:10.2136/sssaj2016.11.0386n

See Also

OSDquery, siblings, fetchOSD

Examples

```
if(requireNamespace("curl") &
    curl::has_internet()) {

    # basic usage
    x <- siblings('zook')
    x$sib

    # restrict to siblings that are major components
    # e.g. the most likely siblings
    x <- siblings('zook', only.major = TRUE)
    x$sib
}
```

`simplifyColorData` *Simplify Color Data by ID*

Description

Simplify multiple Munsell color observations associated with each horizon.

Usage

```
simplifyColorData(d, id.var = "phiid", wt = "colorpct", bt = FALSE)
```

Arguments

`d`	a `data.frame` object, typically returned from NASIS, see details
`id.var`	character vector with the name of the column containing an ID that is unique among all horizons in `d`
`wt`	a character vector with the name of the column containing color weights for mixing
`bt`	logical, should the mixed sRGB representation of soil color be transformed to closest Munsell chips? This is performed by `aqp::rgb2Munsell` `aqp::rgb2Munsell`

Details

This function is mainly intended for the processing of NASIS pedon/horizon data which may or may not contain multiple colors per horizon/moisture status combination. `simplifyColorData` will "mix" multiple colors associated with horizons in `d`, according to IDs specified by `id.var`, using "weights" (area percentages) specified by the `wt` argument to `mix_and_clean_colors`.

Note that this function doesn't actually simulate the mixture of pigments on a surface, rather, "mixing" is approximated via weighted average in the CIELAB colorspace.

The `simplifyColorData` function can be applied to data sources other than NASIS by careful use of the `id.var` and `wt` arguments. However, `d` must contain Munsell colors split into columns named "colorhue", "colorvalue", and "colorchroma". In addition, the moisture state ("Dry" or "Moist") must be specified in a column named "colormoiststst".

The `mix_and_clean_colors` function can be applied to arbitrary data sources as long as `x` contains sRGB coordinates in columns named "r", "g", and "b". This function should be applied to chunks of rows within which color mixtures make sense.

Examples:

- KSSL data
- soil color mixing tutorial

Author(s)

D.E. Beaudette

`simplifyFragmentData` *Simplify Coarse Fraction Data*

Description

Simplify multiple coarse fraction (>2mm) records by horizon.

Usage

```
simplifyFragmentData(
  rf,
  id.var,
  vol.var = "fragvol",
  prefix = "frag",
  nullFragsAreZero = TRUE,
  msg = "rock fragment volume",
  ...
)

simplifyArtifactData(
  art,
  id.var,
```

```
  vol.var = "huartvol",
  nullFragsAreZero = nullFragsAreZero,
  ...
)
```

Arguments

rf	a data.frame object, typically returned from NASIS, see details
id.var	character vector with the name of the column containing an ID that is unique among all horizons in rf
vol.var	character vector with the name of the column containing the coarse fragment volume. Default "fragvol" or "huartvol".
prefix	a character vector prefix for input
nullFragsAreZero	should fragment volumes of NULL be interpreted as 0? (default: TRUE), see details
msg	Identifier of data being summarized. Default is "rock fragment volume" but this routine is also used for "surface fragment cover"
...	Additional arguments passed to sieving function (e.g. sieves a named numeric containing sieve size thresholds with class name)
art	a data.frame object, typically returned from NASIS, see details

Details

This function is mainly intended for processing of NASIS pedon/component data which contains multiple coarse fragment descriptions per horizon. simplifyFragmentData will "sieve out" coarse fragments into the USDA classes, split into hard and para- fragments. Likewise, simplifyArtifactData will sieve out human artifacts, and split total volume into "cohesive", "penetrable", "innocuous", and "persistent".

These functions can be applied to data sources other than NASIS by careful use of the id.var and vol.var arguments.

- rf must contain rock or other fragment volumes in the column "fragvol" (or be specified with vol.var), fragment size (mm) in columns "fragsize_l", "fragsize_r", "fragsize_h", fragment cementation class in "fraghard" and flat/non-flat in "fragshp".
- art must contain artifact volumes in the column "huartvol" (or be specified with vol.var), fragment size (mm) in columns "huartsize_l", "huartsize_r", "huartsize_h", artifact cementation class in "huarthard" and flat/non-flat in "huartshp".

Examples:

- KSSL data

Author(s)

D.E. Beaudette, A.G Brown

`SoilWeb_spatial_query` *Get SSURGO Data via Spatial Query*

Description

Get SSURGO Data via Spatial Query to SoilWeb

Usage

```
SoilWeb_spatial_query(
  bbox = NULL,
  coords = NULL,
  what = "mapunit",
  source = "soilweb"
)
```

Arguments

`bbox`	a bounding box in WGS84 geographic coordinates, see examples
`coords`	a coordinate pair in WGS84 geographic coordinates, see examples
`what`	data to query, currently ignored
`source`	the data source, currently ignored

Details

Data are currently available from SoilWeb. These data are a snapshot of the "official" data. The snapshot date is encoded in the "soilweb_last_update" column in the function return value. Planned updates to this function will include a switch to determine the data source: "official" data via USDA-NRCS servers, or a "snapshot" via SoilWeb.

Value

The data returned from this function will depend on the query style. See examples below.

Note

SDA now supports spatial queries, consider using `SDA_spatialQuery` instead.

Author(s)

D.E. Beaudette

Examples

```
if(requireNamespace("curl") &
    curl::has_internet()) {

    # query by bbox
    SoilWeb_spatial_query(bbox=c(-122.05, 37, -122, 37.05))

    # query by coordinate pair
    SoilWeb_spatial_query(coords=c(-121, 38))
}
```

`STRplot` *Graphical Description of US Soil Taxonomy Soil Temperature Regimes*

Description

Graphical Description of US Soil Taxonomy Soil Temperature Regimes

Usage

```
STRplot(mast, msst, mwst, permafrost = FALSE, pt.cex = 2.75, leg.cex = 0.85)
```

Arguments

`mast`	single value or vector of mean annual soil temperature (deg C)
`msst`	single value or vector of mean summer soil temperature (deg C)
`mwst`	single value of mean winter soil temperature (deg C)
`permafrost`	logical: permafrost presence / absence
`pt.cex`	symbol size
`leg.cex`	legend size

Details

Soil Temperature Regime Evaluation Tutorial

Author(s)

D.E. Beaudette

References

Soil Survey Staff. 2015. Illustrated guide to soil taxonomy. U.S. Department of Agriculture, Natural Resources Conservation Service, National Soil Survey Center, Lincoln, Nebraska.

See Also

estimateSTR

Examples

```
par(mar=c(4,1,0,1))
STRplot(mast = 0:25, msst = 10, mwst = 1)
```

`summarizeSoilTemperature` *Get data from Henry Mount Soil Temperature and Water Database*

Description

This function is a front-end to the REST query functionality of the Henry Mount Soil Temperature and Water Database.

Usage

```
summarizeSoilTemperature(soiltemp.data)

month2season(x)

fetchHenry(
  what = "all",
  usersiteid = NULL,
  project = NULL,
  sso = NULL,
  gran = "day",
  start.date = NULL,
  stop.date = NULL,
  pad.missing.days = TRUE,
  soiltemp.summaries = TRUE,
  tz = ""
)
```

Arguments

`soiltemp.data`	A `data.frame` containing soil temperature data
`x`	character vector containing month abbreviation e.g. `c('Jun', 'Dec', 'Sep')`
`what`	type of data to return: 'sensors': sensor metadata only | 'soiltemp': sensor metadata + soil temperature data | 'soilVWC': sensor metadata + soil moisture data | 'airtemp': sensor metadata + air temperature data | 'waterlevel': sensor metadata + water level data l'all': sensor metadata + all sensor data
`usersiteid`	(optional) filter results using a NASIS user site ID

project	(optional) filter results using a project ID
sso	(optional) filter results using a soil survey office code
gran	data granularity: "hour" (if available), "day", "week", "month", "year"; returned data are averages
start.date	(optional) starting date filter
stop.date	(optional) ending date filter
pad.missing.days	should missing data ("day" granularity) be filled with NA? see details
soiltemp.summaries	should soil temperature ("day" granularity only) be summarized? see details
tz	Used for custom timezone. Default "" is current locale

Details

Filling missing days with NA is useful for computing and index of how complete the data are, and for estimating (mostly) unbiased MAST and seasonal mean soil temperatures. Summaries are computed by first averaging over Julian day, then averaging over all days of the year (MAST) or just those days that occur within "summer" or "winter". This approach makes it possible to estimate summaries in the presence of missing data. The quality of summaries should be weighted by the number of "functional years" (number of years with non-missing data after combining data by Julian day) and "complete years" (number of years of data with >= 365 days of non-missing data).

See:

- Henry Mount Soil Climate Database
- fetchHenry Tutorial

Value

a list containing:

sensors	a sf data.frame object containing site-level information
soiltemp	a data.frame object containing soil temperature timeseries data
soilVWC	a data.frame object containing soil moisture timeseries data
airtemp	a data.frame object containing air temperature timeseries data
waterlevel	a data.frame object containing water level timeseries data

Note

This function and the back-end database are very much a work in progress.

Author(s)

D.E. Beaudette

See Also

fetchSCAN

taxaExtent *Get SoilWeb 800m Major Component Soil Taxonomy Grids*

Description

This function downloads a generalized representation of the geographic extent of any single taxon from the top 4 levels of Soil Taxonomy, or taxa matching a given formative element used in Great Group or subgroup taxa. Data are provided by SoilWeb, ultimately sourced from the current SSURGO snapshot. Data are returned as `raster` objects representing area proportion falling within 800m cells. Currently area proportions are based on major components only. Data are only available in CONUS and returned using an Albers Equal Area / NAD83(2011) coordinate reference system (EPSG: 5070).

Usage

```
taxaExtent(
  x,
  level = c("order", "suborder", "greatgroup", "subgroup"),
  formativeElement = FALSE,
  timeout = 60,
  as_Spatial = getOption("soilDB.return_Spatial", default = FALSE)
)
```

Arguments

`x`	single taxon label (e.g. `haploxeralfs`) or formative element (e.g. `pale`), case-insensitive
`level`	the taxonomic level within the top 4 tiers of Soil Taxonomy, one of `'order'`, `'suborder'`, `'greatgroup'`, `'subgroup'`
`formativeElement`	logical, search using formative elements instead of taxon label
`timeout`	time that we are willing to wait for a response, in seconds
`as_Spatial`	Return raster (`RasterLayer`) classes? Default: `FALSE`.

Details

See the Geographic Extent of Soil Taxa tutorial for more detailed examples.

Taxon Queries:

Taxon labels can be conveniently extracted from the `"ST_unique_list"` sample data, provided by the SoilTaxonomy package.

Formative Element Queries:

Greatgroup::

The following labels are used to access taxa containing the following formative elements (in parentheses)

- acr: (acro/acr) extreme weathering
- alb: (alb) presence of an albic horizon
- anhy: (anhy) very dry
- anthra: (anthra) presence of an anthropic epipedon
- aqu: (aqui/aqu) wetness
- argi: (argi) presence of an argillic horizon
- calci: (calci) presence of a calcic horizon
- cry: (cryo/cry) cryic STR
- dur: (duri/dur) presence of a duripan
- dystr: (dystro/dystr) low base saturation
- endo: (endo) ground water table
- epi: (epi) perched water table
- eutr: (eutro/eutr) high base saturation
- ferr: (ferr) presence of Fe
- fibr: (fibr) least decomposed stage
- fluv: (fluv) flood plain
- fol: (fol) mass of leaves
- fragi: (fragi) presence of a fragipan
- fragloss: (fragloss) presence of a fragipan and glossic horizon
- frasi: (frasi) not salty
- fulv: (fulvi/fulv) dark brown with organic carbon
- glac: (glac) presence of ice lenses
- gloss: (glosso/gloss) presence of a glossic horizon
- gypsi: (gypsi) presence of a gypsic horizon
- hal: (hal) salty
- hemi: (hemi) intermediate decomposition
- hist: (histo/hist) organic soil material
- hum: (humi/hum) presence of organic carbon
- hydr: (hydro/hydr) presence of water
- kandi: (kandi) presence of a kandic horizon
- kanhap: (kanhaplo/kanhap) thin kandic horizon
- luvi: (luvi) illuvial organic material
- melan: (melano/melan) presence of a melanic epipedon
- moll: (molli/moll) presence of a mollic epipedon
- natr: (natri/natr) presence of a natric horizon
- pale: (pale) excessive development
- petr: (petro/petr) petrocalcic horizon
- plac: (plac) presence of a thin pan
- plagg: (plagg) presence of a plaggen epipedon
- plinth: (plinth) presence of plinthite
- psamm: (psammo/psamm) sandy texture
- quartzi: (quartzi) high quartz content

- rhod: (rhodo/rhod) dark red colors
- sal: (sali/sal) presence of a salic horizon
- sapr: (sapr) most decomposed stage
- sombri: (sombri) presence of a sombric horizon
- sphagno: (sphagno) presence of sphagnum moss
- sulf: (sulfo/sulfi/sulf) presence of sulfides or their oxidation products
- torri: (torri) torric/aridic SMR
- ud: (udi/ud) udic SMR
- umbr: (umbri/umbr) presence of an umbric epipedon
- ust: (usti/ust) ustic SMR
- verm: (verm) wormy, or mixed by animals
- vitr: (vitri/vitr) presence of glass
- xer: (xero/xer) xeric SMR

Subgroup::
The following labels are used to access taxa containing the following formative elements (in parenthesis).

- abruptic: (abruptic) abrupt textural change
- acric: (acric) low apparent CEC
- aeric: (aeric) more aeration than typic subgroup
- albaquic: (albaquic) presence of albic minerals, wetter than typic subgroup
- albic: (albic) presence of albic minerals
- alfic: (alfic) presence of an argillic or kandic horizon
- alic: (alic) high extractable Al content
- anionic: (anionic) low CEC or positively charged
- anthraquic: (anthraquic) human controlled flooding as in paddy rice culture
- anthropic: (anthropic) an anthropic epipedon
- aquic: (aquic) wetter than typic subgroup
- arenic: (arenic) 50-100cm sandy textured surface
- argic: (argic) argillic horizon
- aridic: (aridic) more aridic than typic subgroup
- calcic: (calcic) presence of a calcic horizon
- chromic: (chromic) high chroma colors
- cumulic: (cumulic) thickened epipedon
- duric: (duric) presence of a duripan
- durinodic: (durinodic) presence of durinodes
- dystric: (dystric) lower base saturation percentage
- entic: (entic) minimal surface/subsurface development
- eutric: (eutric) higher base saturation percentage
- fibric: (fibric) >25cm of fibric material
- fluvaquentic: (fluvaquentic) wetter than typic subgroup, evidence of stratification
- fragiaquic: (fragiaquic) presence of fragic properties, wetter than typic subgroup
- fragic: (fragic) presence of fragic properties

- glacic: (glacic) presence of ice lenses or wedges
- glossaquic: (glossaquic) interfingered horizon boundaries, wetter than typic subgroup
- glossic: (glossic) interfingered horizon boundaries
- grossarenic: (grossarenic) >100cm sandy textured surface
- gypsic: (gypsic) presence of gypsic horizon
- halic: (halic) salty
- haplic: (haplic) central theme of subgroup concept
- hemic: (hemic) >25cm of hemic organic material
- humic: (humic) higher organic matter content
- hydric: (hydric) presence of water
- kandic: (kandic) low activity clay present
- lamellic: (lamellic) presence of lamellae
- leptic: (leptic) thinner than typic subgroup
- limnic: (limnic) presence of a limnic layer
- lithic: (lithic) shallow lithic contact present
- natric: (natric) presence of sodium
- nitric: (nitric) presence of nitrate salts
- ombroaquic: (ombroaquic) surface wetness
- oxyaquic: (oxyaquic) water saturated but not reduced
- pachic: (pachic) epipedon thicker than typic subgroup
- petrocalcic: (petrocalcic) presence of a petrocalcic horizon
- petroferric: (petroferric) presence of petroferric contact
- petrogypsic: (petrogypsic) presence of a petrogypsic horizon
- petronodic: (petronodic) presence of concretions and/or nodules
- placic: (placic) presence of a placic horizon
- plinthic: (plinthic) presence of plinthite
- rhodic: (rhodic) darker red colors than typic subgroup
- ruptic: (ruptic) intermittent horizon
- salic: (salic) presence of a salic horizon
- sapric: (sapric) >25cm of sapric organic material
- sodic: (sodic) high exchangeable Na content
- sombric: (sombric) presence of a sombric horizon
- sphagnic: (sphagnic) sphagnum organic material
- sulfic: (sulfic) presence of sulfides
- terric: (terric) mineral substratum within 1 meter
- thapto: (thaptic/thapto) presence of a buried soil horizon
- turbic: (turbic) evidence of cryoturbation
- udic: (udic) more humid than typic subgroup
- umbric: (umbric) presence of an umbric epipedon
- ustic: (ustic) more ustic than typic subgroup
- vermic: (vermic) animal mixed material
- vitric: (vitric) presence of glassy material
- xanthic: (xanthic) more yellow than typic subgroup
- xeric: (xeric) more xeric than typic subgroup

Value

a SpatRaster object (or RasterLayer when as_Spatial=TRUE)

Author(s)

D.E. Beaudette and A.G. Brown

Examples

```
## Not run:
\donttest{

if(requireNamespace("curl") &
   requireNamespace("terra") &
   curl::has_internet()) {

  library(terra)

  # soil order
  taxa <- 'vertisols'
  x <- taxaExtent(taxa, level = 'order')

  # suborder
  taxa <- 'ustalfs'
  x <- taxaExtent(taxa, level = 'suborder')

  # greatgroup
  taxa <- 'haplohumults'
  x <- taxaExtent(taxa, level = 'greatgroup')

  # subgroup
  taxa <- 'Typic Haploxerepts'
  x <- taxaExtent(taxa, level = 'subgroup')

  # greatgroup formative element
  taxa <- 'psamm'
  x <- taxaExtent(taxa, level = 'greatgroup', formativeElement = TRUE)

  # subgroup formative element
  taxa <- 'abruptic'
  x <- taxaExtent(taxa, level = 'subgroup', formativeElement = TRUE)

  # coarsen for faster plotting
  a <- terra::aggregate(x, fact = 5, na.rm = TRUE)

  # quick evaluation of the result
  terra::plot(a, axes = FALSE)
}
}


## End(Not run)
```

uncode *Convert coded values returned from NASIS and SDA queries into human-readable values*

Description

These functions convert the coded values returned from NASIS or SDA to factors (e.g. 1 = Alfisols) using the metadata tables from NASIS. For SDA the metadata is pulled from a static snapshot in the soilDB package (/data/metadata.rda).

Usage

```
uncode(
  df,
  invert = FALSE,
  db = "NASIS",
  droplevels = FALSE,
  stringsAsFactors = NULL,
  dsn = NULL
)

code(df, db = NULL, droplevels = FALSE, stringsAsFactors = NULL, dsn = NULL)
```

Arguments

df	data.frame
invert	converts the code labels back to their coded values (FALSE)
db	label specifying the soil database the data is coming from, which indicates whether or not to query metadata from local NASIS database ("NASIS") or use soilDB-local snapshot ("LIMS" or "SDA")
droplevels	logical: indicating whether to drop unused levels in classifying factors. This is useful when a class has large number of unused classes, which can waste space in tables and figures.
stringsAsFactors	deprecated
dsn	Optional: path to local SQLite database containing NASIS table structure; default: NULL

Details

These functions convert the coded values returned from NASIS into their plain text representation. It duplicates the functionality of the CODELABEL function found in NASIS. This function is primarily intended to be used internally by other soilDB R functions, in order to minimize the need to manually convert values.

The function works by iterating through the column names in a data frame and looking up whether they match any of the ColumnPhysicalNames found in the metadata domain tables. If matches are

found then the columns coded values are converted to their corresponding factor levels. Therefore it is not advisable to reuse column names from NASIS unless the contents match the range of values and format found in NASIS. Otherwise uncode() will convert their values to NA.

When data is being imported from NASIS, the metadata tables are sourced directly from NASIS. When data is being imported from SDA or the NASIS Web Reports, the metadata is pulled from a static snapshot in the soilDB package.

Set `options(soilDB.NASIS.skip_uncode = TRUE)` to bypass decoding logic; for instance when using soilDB NASIS functions with custom NASIS snapshots that have already been decoded.

Value

A `data.frame` with the results.

Author(s)

Stephen Roecker

Examples

```
# convert column name `fraghard` (fragment hardness) codes to labels
uncode(data.frame(fraghard = 1:10))

# convert column name `fragshp` (fragment shape) labels to codes
code(data.frame(fragshp = c("flat", "nonflat")))
```

---

us_ss_timeline *Timeline of US Published Soil Surveys*

Description

This dataset contains the years of each US Soil Survey was published.

Format

A data.frame with 5209 observations on the following 5 variables.

- `"ssa"`: Soil Survey name, a character vector
- `"year"`: Year of publication, a numeric vector
- `"pdf"`: Does a manuscript PDF document exist? a logical vector
- `"state"`: State abbreviation, a character vector

Details

This data was web scraped from the NRCS Soils Website. The scraping procedure and a example plot are included in the examples section below.

Source

https://www.nrcs.usda.gov/wps/portal/nrcs/soilsurvey/soils/survey/state/

`waterDayYear` *Compute Water Day and Year*

Description

Compute "water" day and year, based on the end of the typical or legal dry season. This is September 30 in California.

Usage

```
waterDayYear(d, end = "09-30", format = "%Y-%m-%d", tz = "")
```

Arguments

`d`	anything the can be safely converted to `POSIXlt`
`end`	"MM-DD" notation for end of water year
`format`	Used in POSIXlt conversion. Default `"%Y-%m-%d"`
`tz`	Used in POSIXlt conversion for custom timezone. Default `""` is current locale

Details

This function doesn't know about leap-years. Probably worth checking.

Value

A `data.frame` object with the following

`wy`	the "water year"
`wd`	the "water day"

Author(s)

D.E. Beaudette

References

Ideas borrowed from: https://github.com/USGS-R/dataRetrieval/issues/246 and https://stackoverflow.com/questions/48123049/create-day-index-based-on-water-year

Examples

```
# try it
waterDayYear('2019-01-01')
```

`WCS_details` *Web Coverage Services Details*

Description

List variables or databases provided by soilDB web coverage service (WCS) abstraction. These lists will be expanded in future versions.

Usage

```
WCS_details(wcs = c("mukey", "ISSR800"))
```

Arguments

`wcs`	a WCS label ('mukey' or 'ISSR800')

Value

a `data.frame`

Examples

```
WCS_details(wcs = 'ISSR800')
```

Index