Package ‘taxotools’

April 4, 2024

Type Package
Title Taxonomic List Processing
Version 0.0.139
Date 2024-04-03
Maintainer Vijay Barve <vijay.barve@gmail.com>
Description Taxonomic lists matching and merging, casting and melting scientific names, managing taxonomic lists from Global Biodiversity Information Facility 'GBIF' or Integrated Taxonomic Information System 'ITIS', harvesting names from Wikipedia and fuzzy matching.
License CC0
Imports taxize, wikitaxa, plyr, sqldf, stringr, stringdist, rmarkdown, stringi
Encoding UTF-8
RoxygenNote 7.3.1
BugReports https://github.com/vijaybarve/taxotools/issues
NeedsCompilation no
Author Vijay Barve [aut, cre] (<https://orcid.org/0000-0002-4852-2567>)
Repository CRAN
Date/Publication 2024-04-04 07:40:02 UTC

R topics documented:

build_gen_syn ........................................ 2
cast_canonical ....................................... 3
cast_cs_field .......................................... 4
cast_scientificname .................................. 5
check_scientific ..................................... 7
compact_ids .......................................... 7
DwC2taxo ............................................... 9
expand_name ......................................... 10
get_accepted_names .................................. 11
**build_gen_syn**

Build genus level synonyms

**Description**

Build a genus level synonym list from master list.

**Usage**

```r
build_gen_syn(dat)
```

**Arguments**

- `dat` master list

**Details**

This genus level synonym list is generated for passing on to get_accepted_names function as a parameter

**Value**

data frame with genus level synonyms with two columns viz. Valid_genus and Original_Genus
**cast_canonical**

Construct canonical names

**Description**

Construct canonical names using Genus, Species and Subspecies fields. At times due to spaces or NAs in the data fields, it makes it tricky to generate canonical names.

**Usage**

```r
cast_canonical(
  dat,
  canonical = "canonical",
  genus = "",
  species = "",
  subspecies = "",
  verbose = FALSE
)
```
cast_cs_field

Description

Builds a character (comma) separated list within a field given a data frame with primary field repeating values and secondary field with values to be character separated in the same field (secondary)

Usage

```r
cast_cs_field(
  data,
  pri,
  sec,
  duplicate = FALSE,
  sepchar = "",
  verbose = FALSE
)
```

Arguments

- `dat` data frame containing taxonomic list
- `canonical` field name for canonical names. Default 'canonical'
- `genus` field name for Genus field
- `species` field name for Species field
- `subspecies` field name for Subspecies field
- `verbose` verbose output, Default: FALSE

Value

a data frame containing Canonical names field added or repopulated using filed names for Genus, Species and Subspecies specified in parameters

See Also

Other Name functions: `build_gen_syn()`, `cast_scientificname()`, `check_scientific()`, `expand_name()`, `get_accepted_names()`, `guess_taxo_rank()`, `list_higher_taxo()`, `melt_canonical()`, `melt_scientificname()`, `resolve_names()`, `taxo_fuzzy_match()`

Examples

```r
mylist <- data.frame("genus" = c("Acodon", "Akodon", "Abrothrix", "Abeomelomys"),
                      "species" = c("jelskii","longipilis","longipilis", "sevia"),
                      "subspecies" = c("pyrrhotis","castaneus","", NA))
cast_canonical(mylist,"canonical","genus","species","subspecies")
```
cast_scientificname

Cast scientific name using taxonomic fields

Description

Combine scientific names using Genus, Species, Subspecies, Author etc.

Usage

cast_scientificname(
  dat = NULL,
  sciname = "scientificname",
  genus = ",",
  subgenus = ",",
  species = ",",
  subspecies = ",",
  author = ",",
  verbose = FALSE
)
Arguments

dat  data frame containing taxonomic data
sciname  column name for scientific names, Default: 'scientificname'
genus  column name for genus, Default: 'genus'
subgenus  column name for subgenus, Default: '
Species  column name for species, Default: 'species'
subspecies  column name for subspecies, Default: 'subspecies'
author  column name for author, Default: 'author'
verbose  verbose output, Default: FALSE

Details

Helpful function to break down Scientific names into Genus, Subgenus, species, Subspecies, Author so that the names can be constructed into canonical names for matching

Value

data frame with additional columns for taxonomic fields

See Also

Other Name functions: build_gen_syn(), cast_canonical(), check_scientific(), expand_name(), get_accepted_names(), guess_taxo_rank(), list_higher_taxo(), melt_canonical(), melt_scientificname(), resolve_names(), taxo_fuzzy_match()

Examples

mylist <- data.frame("id" = c(11,12,13,14,15,16,17,18,19),
"genus" = c("Hypochlorosis","Hypochlorosis","Hypochlorosis","Myrina","Hypochlorosis","Hypochlorosis","Hypochlorosis","Seuku","Sithon"),
"subgenus" = c("","","","","","","","",""),
"species" = c("ancharia","ancharia","ancharia","lorquinii","ancharia","ancharia","lorquinii","emlongi","lorquinii"),
"subspecies" = c("","ancharia","humboldtii","","tenebrosa","tenebrosa","",""),
"author" = c("(Hewitson, 1869)","(Hewitson, 1869)","Druce, 1894", "C. & R. Felder, 1865","Rothschild, 1915", "Rothschild, 1915","Fruhstorfer, 1908", "(Domning et al., 1986)",""),
stringsAsFactors = FALSE)
cast_scientificname(mylist, genus = "genus", subgenus = "subgenus",
species = "species", subspecies = "subspecies",
author = "author")
check_scientific

Parse and resolve a scientific name string

Description
Parse the name using Global Names Resolver 'GNR' and Global Biodiversity Information Facility 'GBIF' parse API to make sure the name is scientific name

Usage
check_scientific(name)

Arguments
name scientific name string to be checked

Value
Resolved canonical name (NULL if not matched)

See Also
Other Name functions: build_gen_syn(), cast_canonical(), cast_scientificname(), expand_name(), get_accepted_names(), guess_tayo_rank(), list_higher_taxo(), melt_canonical(), melt_scientificname(), resolve_names(), taxo_fuzzy_match()

Examples
check_scientific("Akodon longipilis (Waterhouse, 1837)")
check_scientific("Mus longipilis Waterhouse, 1837")
check_scientific("Akodon hershkovitzi Patterson, Gallardo, and Freas, 1984")

compact_ids

compact id numbers

Description
Compacting and converting the id values to numeric if required to make sure dependent functions work well

Usage
compact_ids(dat, id = "id", accid = "accid", startid = 1, verbose = TRUE)
compact_ids

Arguments

dat        taxonomic list in a data frame with id and accid columns
id         column name for 'id'. Default 'id'
accid      column name for 'accid'. Default 'accid'
startid    starting id number for the list. Default 1
verbose    verbose output on the console

Details

Helper function to make sure values for ids are in right format and are compact

Value

returns data frame

See Also

Other List functions: DwC2taxo(), cast_cs_field(), get_synonyms(), match_lists(), melt_cs_field(),
merge_lists(), syn2taxo(), synonymize_subspecies(), taxo2DwC(), taxo2doc(), taxo2syn(),
wiki2taxo()

Examples

mylist <- data.frame("id" = c("1", "2", "3", "4", "5"),
"canonical" = c("Hypochlorosis ancharia",
"Pseudonotis humboldti",
"Myrina ancharia",
"Hypochlorosis ancharia obiana",
"Hypochlorosis lorquinii"),
"family" = c("Lycaenidae", "Lycaenidae",
"Lycaenidae", "Lycaenidae",
"Lycaenidae"),
"accid" = c("0", "1", "1", "0", "0"),
"source" = c("itis", "wiki", "wiki", "itis",
"itis"),
stringsAsFactors = FALSE)

mylist_c <- compact_ids(mylist)

mylist_c <- compact_ids(mylist,startid=1001)

mylist <- data.frame("id" = c(11,12,13,14,15),
"canonical" = c("Hypochlorosis ancharia",
"Pseudonotis humboldti",
"Myrina ancharia",
"Hypochlorosis ancharia obiana",
"Hypochlorosis lorquinii"),
"family" = c("Lycaenidae", "Lycaenidae",
"Lycaenidae", "Lycaenidae",
"Lycaenidae"),
DwC2taxo

Darwin Core to Taxolist format

Description

Converts a Darwin Core name list to taxolist format

Usage

DwC2taxo(namelist, statuslist = NA, source = NA)

Arguments

namelist names list in Darwin Core format

statuslist vector listing taxonomicStatus to be considered in the namelist. If Default value is NA, automatically uses list of

- Accepted
- Synonym
- Valid
- heterotypic Synonym
- homotypic Synonym
- doubtful,
- proparte synonym

source source of the namelist i.e. Global Biodiversity Information Facility ’GBIF’ or Integrated Taxonomic Information System ’ITIS’. Default NA

Details

The name lists downloaded from ’GBIF’ or ’ITIS’ website in Darwin Core (DwC) format has all the required fields for taxolist. The list just needs to be converted to taxolist by renaming column names and and quality checked in terms of missing synonym to accepted name linkages at times.

Value

names list is taxolist format
See Also

Other List functions: `cast_cs_field()`, `compact_ids()`, `get_synonyms()`, `match_lists()`, `melt_cs_field()`, `merge_lists()`, `syn2taxo()`, `synonymize_subspecies()`, `taxo2DwC()`, `taxo2doc()`, `taxo2syn()`, `wiki2taxo()`

Examples

dwclist <- data.frame("taxonKey" = c("5129025","6224429","1896957"),
  "scientificName" = c("Charaxes solon Fabricius, 1793",
    "Papilio jason Linnaeus, 1767",
    "Charaxes jasius (Linnaeus, 1767)"),
  "acceptedTaxonKey" = c("5129025","1896957","1896957"),
  "acceptedScientificName" = c("Charaxes solon Fabricius, 1793",
    "Charaxes jasius (Linnaeus, 1767)"),
  "taxonRank" = c("SPECIES","SPECIES","SPECIES"),
  "taxonomicStatus" = c("ACCEPTED","SYNONYM","ACCEPTED"),
  "family" = c("Nymphalidae","Nymphalidae","Nymphalidae"),
  "order" = c("Lepidoptera","Lepidoptera","Lepidoptera"),
  stringsAsFactors = FALSE)

mytaxo <- DwC2taxo(dwclist)

---

### Description

At times the genus is specified with first character and ‘.’ rather then repeating genus names every time. These are either synonyms or species of the same genus listed one bellow another. To convert these names to canonical names, we need to expand the genus name (typically) using previous entry in the list.

### Usage

`expand_name(fullname, shortname)`

### Arguments

- `fullname` full scientific name
- `shortname` scientific name with short form genus name to expand the Genus

### Value

scientific name with Genus expanded using reference name provided as parameter
get_accepted_names

See Also

Other Name functions: build_gen_syn(), castcanonical(), cast_scientificname(), check_scientific(), get_accepted_names(), guess_tayo_rank(), list_higher_taxo(), melt_canonical(), melt_scientificname(), resolve_names(), taxo_fuzzy_match()

Examples

expand_name("Addax gibbosa", "A. mytilopes")
expand_name("Oryx addax", "O. nasomaculatus")

Description

Match namelist with master and fetch the accepted names using the linkages provided within the data

Usage

get_accepted_names(
  namelist,
  master,
  gen_syn = NA,
  namelookup = NA,
  mastersource = NA,
  match_higher = FALSE,
  fuzzymatch = TRUE,
  fuzzydist = 2,
  canonical = NA,
  genus = NA,
  species = NA,
  subspecies = NA,
  prefix = "",
  verbose = TRUE
)

Arguments

namelist  data frame of the list of names to be resolved. Must contain either column canonical containing binomial or trinomial name without spp. and var. etc. or may contain columns for genus, species and subspecies (any sub-specific unit) and the names of the columns are passed as subsequent parameters.
**get_accepted_names**

**master**
data frame with required columns id, canonical and accid. Other columns like order, family are optional. Column id is typically running ids for each record and accid will contain 0 if the name is currently accepted name and id number of accepted name in case the name is a synonym. Column canonical contains binomial or trinomial without spp. var. etc.

**gen_syn**
data frame with columns Original_Genus and Valid_Genus where Original_genus is synonym and valid_genus is one present in the master. Default: NA when gen_syn is not used.

**namelookup**
Lookup data frame for names where some names might need manual lookup. The columns required are binomial and validname where binomial is new name and validname is present in the master. Default: NA when namelookup is not used.

**mastersource**
vector of sources to be used for assignment with priority

**match_higher**
match genus and family names present in canonical field

**fuzzymatch**
attempt fuzzy matching or not. Default: TRUE

**fuzzydist**
fuzzy distance while matching. Default: 2

**canonical**
column containing names to be resolved to accepted names, Default: NA when columns for genus and species are specified.

**genus**
column containing genus names to be resolved to accepted names and typically accompanied by species and subspecies columns, Default: NA when canonical parameter is supplied.

**species**
column containing species names to be resolved to accepted names and is accompanied by genus, Default: NA

**subspecies**
column containing species names to be resolved to accepted names and is accompanied by genus and species, Default: NA

**prefix**
to be added to all the return fields

**verbose**
display process messages, Default: TRUE

**Details**

Name resolution methods:

- **direct** - was a direct match with name or a synonym
- **direct2** - was a direct match with name or a synonym in non mastersource
- **fuzzy** - used fuzzy matching
- **gensyn** - genus substitution with known genus level synonyms
- **lookup** - Manual lookup in earlier processing
- **sppdrop** - subspecies was dropped
- **sub2sp** - subspecies elevated to species
- **genus** - genus was matched
- **family** - family was matched
- **NA** - could not be resolved

Note: Make sure all the data frames have same character encoding to prevent errors.
get_accepted_names

Value
data frame containing all the original columns with following additional columns:

- **accepted_name** - Accepted name present in the master. NA is not resolved
- **method** - method used to resolve the name. See details for explanation of each method

See Also
Other Name functions: `build_gen_syn()`, `cast_canonical()`, `cast_scientificname()`, `check_scientific()`, `expand_name()`, `guess_taxo_rank()`, `list_higher_taxo()`, `melt_canonical()`, `melt_scientificname()`, `resolve_names()`, `taxo_fuzzy_match()`

Examples

```r
master <- data.frame("id" = c(1,2,3,4,5,6,7),
                         "canonical" = c("Hypochlorosis ancharia",
                                        "Hypochlorosis tenebrosa",
                                        "Pseudonotis humboldti",
                                        "Myrina ancharia",
                                        "Hypochlorosis ancharia tenebrosa",
                                        "Hypochlorosis ancharia obiana",
                                        "Hypochlorosis lorquinii"),
                         "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae",
                                      "Lycaenidae", "Lycaenidae", "Lycaenidae",
                                      "Lycaenidae"),
                         "accid" = c(0,1,1,1,0,0,0),
                         "source" = c("itis","itis","wiki","wiki","itis",
                                      "itis","itis"),
                         stringsAsFactors = FALSE)

mylist <- data.frame("id" = c(11,12,13,14,15,16,17,18,19),
                       "scname" = c("Hypochlorosis ancharia",
                                     "Hypochlorosis ancharii",
                                     "Hypochlorosis tenebrosa",
                                     "Pseudonotis humboldti",
                                     "Abrothrix longipilis",
                                     "Myrinoa anchariana",
                                     "Hypochlorosis ancharia ancharia",
                                     "Myrina lorquinii",
                                     "Sithon lorquinii"),
                       stringsAsFactors = FALSE)

res <- get_accepted_names(namelist = mylist,
                          master=master,
                          canonical = "scname")

gen_syn_list <- data.frame("Original_Genus"=c("Pseudonotis",
                                             "Myrina"),
                          "Valid_Genus"=c("Hypochlorosis",
                                          "Hypochlorosis"),
                          stringsAsFactors = FALSE)
```
res <- get_accepted_names(namelist = mylist,
  master=master,
  gen_syn = gen_syn_list,
  canonical = "scname")

lookup_list <- data.frame("binomial"=c("Sithon lorquinii",
  "Hypochlorosis humboldtii"),
  "validname"=c("Hypochlorosis lorquinii",
  "Hypochlorosis lorquinii"),
  stringsAsFactors = FALSE)

res <- get_accepted_names(namelist = mylist,
  master=master,
  gen_syn = gen_syn_list,
  namelookup = lookup_list,
  canonical = "scname")

mylist_s <- melt_canonical(mylist,canonical = "scname",
  genus = "genus",
  species = "species",
  subspecies = "subspecies")

res <- get_accepted_names(namelist = mylist_s,
  master=master,
  gen_syn = gen_syn_list,
  namelookup = lookup_list,
  genus = "genus",
  species = "species",
  subspecies = "subspecies")

res <- get_accepted_names(namelist = mylist_s,
  master=master,
  gen_syn = gen_syn_list,
  namelookup = lookup_list,
  mastersource = c("itis"),
  genus = "genus",
  species = "species",
  subspecies = "subspecies")

mylist <- data.frame("id"= c(11,12,13,14,15,16,17,18),
  "scname" = c("Hypochlorosis ancharia",
  "Hypochlorosis ancharii",
  "Hypochlorosis",
  "Pseudonotis",
  "Lycaenidae",
  "Pseudonotis humboldtii",
  "Abrothrix longipilis",
  "Myrinana anchariana"),
  stringsAsFactors = FALSE)

res <- get_accepted_names(namelist = mylist,
  master=master,
**get_itis_syn**

Get Integrated Taxonomic Information System 'ITIS' Synonyms for a Scientific Name

**Description**
Fetch Synonyms using Integrated Taxonomic Information System ‘ITIS’ web service

**Usage**
```
get_itis_syn(scname)
```

**Arguments**
- **scname**: Scientific Name

**Value**
a list containing synonyms

**See Also**
Other ITIS functions: `list_itis_syn()`

**Examples**
```
get_itis_syn("Abrothrix longipilis")
get_itis_syn("Abditomys latidens")
```

---

**get_synonyms**

get synonyms

**Description**
get all the synonyms from the master list for the names in the checklist

**Usage**
```
get_synonyms(master = NULL, checklist = NULL, commasep = FALSE, verbose = TRUE)
```
Arguments

master      master list of names (taxolist)
checklist   list of names to be processed (taxolist)
commasep   return list should be comma separated list or each synonym on its own row.
            Default false
verbose    verbose output on the console

Value

Data frame with names from the checklist and their synonyms present in the master list

See Also

Other List functions: DwC2taxo(), cast_cs_field(), compact_ids(), match_lists(), melt_cs_field(),
merge_lists(), syn2taxo(), synonymize_subspecies(), taxo2DwC(), taxo2doc(), taxo2syn(),
wiki2taxo()

Examples

```r
master <- data.frame("id" = c(1,2,3,4,5),
  "canonical" = c("Hypochlorosis ancharia",
  "Pseudonotis humboldti",
  "Myrina ancharia",
  "Hypochlorosis ancharia obiana",
  "Hypochlorosis lorquinii"),
  "family" = c("Lycaenidae", "Lycaenidae",
  "Lycaenidae", "Lycaenidae",
  "Lycaenidae"),
  "accid" = c(0,1,1,0,0),
  "source" = c("itis","wiki","wiki","itis",
  "itis"),
stringsAsFactors = FALSE)
checklist <- data.frame("id" = c(1,2,3),
  "canonical" = c("Hypochlorosis ancharia",
  "Hypochlorosis tenebrosa",
  "Hypochlorosis ancharia tenebrosa"),
  "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae"),
  "accid" = c(0,1,0),
  "source" = c("itis","itis","itis"),
stringsAsFactors = FALSE)
get_synonyms(master,checklist,commasep=FALSE)
get_synonyms(master,checklist,commasep=TRUE)
```
guess_taxo_rank

----

guess_taxo_rank  Guess the taxonomic rank of Scientific Name

Description

Guesses the taxonomic rank i.e. Genus, Species or Subspecies based on number of words

Usage

guess_taxo_rank(name)

Arguments

name  scientific name string to be checked

Value

"Genus or above"  = single word
"Species"  = two words
"Subspecies"  = three words
"Unknown"  = zero or more than three words

See Also

Other Name functions: build_gen_syn(), cast_canonical(), cast_scientificname(), check_scientific(), expand_name(), get_accepted_names(), list_higher_taxo(), melt_canonical(), melt_scientificname(), resolve_names(), taxo_fuzzy_match()

Examples

guess_taxo_rank(""")
guess_taxo_rank("Akodon longipilis")
guess_taxo_rank("Akodon")
guess_taxo_rank("Abrocoma cinerea shistacea")
guess_taxo_rank("Abrocoma cinerea shistacea shistacea")
list_higher_taxo  

Get higher taxonomy data for list of names

Description

Retrieve higher taxonomy information (like Family and Order) for each record from the "Encyclopedia of Life" web API.

Usage

```
list_higher_taxo(
  indf,
  canonical,
  genus = FALSE,
  verbose = FALSE,
  progress = TRUE
)
```

Arguments

- `indf` input data frame containing taxonomic list
- `canonical` field name containing scientific names
- `genus` If TRUE, use only genus level data to get taxonomy
- `verbose` If TRUE, displays each name string for which the higher taxonomy is sought
- `progress` If TRUE prints progress bar and messages on the console.

Details

This function makes use of certain functions in the `taxize` package. It scans and retrieves the taxonomic hierarchy for each scientific name (or just genus name) in the data set. When new data are retrieved, they are stored in a local sqlite database, taxo.db, for faster further access.

Value

- data frame with added / updated columns
  - "Kingdom" Kingdom of the Scientific name
  - "Phylum" Phylum of the Scientific name
  - "Order_" Order of the Scientific name
  - "Family" Family of the Scientific name
  - "Genus" Genus of the Scientific name

and also saves a local copy of taxonomy downloaded for future use in 'taxo.db' sqlite file
**list_itis_syn**

Get ITIS Synonyms for list of names

**Description**

Fetch Synonyms from Integrated Taxonomic Information System 'ITIS'

**Usage**

```r
list_itis_syn(namelist)
```

**Arguments**

- `namelist` list of scientific names

**Value**

a data frame containing canonical names (passed) and synonyms

**See Also**

Other ITIS functions: `get_itis_syn()`

**Examples**

```r
list_itis_syn("Abrothrix longipilis")
list_itis_syn(c("Abditomys latidens", "Abeomelomys sevia", "Abrothrix jelskii"))
```
**list_wiki_syn**  
Get Wikipedia Synonyms for list of names

**Description**
Fetch Synonyms from Wikipedia and clean them for use

**Usage**
```r
list_wiki_syn(namelist, verbose = TRUE)
```

**Arguments**
- `namelist`: list of scientific names
- `verbose`: status output. Default TRUE

**Value**
a data frame containing names, synonyms and Canonical synonyms matched with is scientific name backbone taxonomy

- **Name**: Scientific name
- **WikiName**: Wikipedia page name
- **OrigSyn**: Original synonym returned by Wikipedia
- **Syn**: Synonym in canonical form, matched with GBIF

**Examples**
```r
list_wiki_syn("Abrothrix illutea")
list_wiki_syn(c("Abditomys latidens", "Abeomelomys sevia",
    "Abrocoma schistacea"))
```

**match_lists**  
match two taxonomic lists

**Description**
match two taxonomic lists using canonical names

**Usage**
```r
match_lists(master, checklist, masterfld, checklistfld)
```
**melt_canonical**

Deconstruct canonical names

**Description**

Deconstruct canonical names into Genus, Species and Subspecies fields

**Arguments**

- **master**: master taxonomic list
- **checklist**: match taxonomic list
- **masterfld**: field name for canonical name in master list
- **checklistfld**: field name for canonical name in match list

**Value**

A list with data frames containing matched records, records only in master and checklist and statistics about the records including Jaccard index.

**See Also**

Other List functions: `DwC2taxo()`, `cast_cs_field()`, `compact_ids()`, `get_synonyms()`, `melt_cs_field()`, `merge_lists()`, `syn2taxo()`, `synonymize_subspecies()`, `taxo2DwC()`, `taxo2doc()`, `taxo2syn()`, `wiki2taxo()`

**Examples**

```r
master <- data.frame("canonical" = c("Abrothrix longipilis",
    "Acodon hirtus",
    "Akodon longipilis apta",
    "Akodon longipilis castaneus",
    "Chroeomys jelskii",
    "Acodon jelskii pyrrhotis"),
   stringsAsFactors = FALSE)
checklist <- data.frame("canonical" = c("Abrothrix longipilis",
    "Akodon longipilis apta",
    "Akodon longipilis castaneus",
    "Abrothrix jelskii",
    "Acodon jelskii pyrrhotis"),
   stringsAsFactors = FALSE)
match_lists(master, checklist, "canonical", "canonical")
```
Usage

melt_canonical(
  dat,
  canonical = "",
  genus = "",
  species = "",
  subspecies = "",
  verbose = FALSE
)

Arguments

dat data frame containing taxonomic list
canonical field name for canonical names
genus field name for Genus
species field name for Species
subspecies field name for Subspecies
verbose verbose output, Default: FALSE

Value

a data frame containing Genus, Species and Subspecies fields added or repopulated using data in canonical name field. If unable to parse the name Genus, Species and Subspecies fields will have NA.

See Also

Other Name functions: build_gen_syn(), cast_canonical(), cast_scientificname(), check_scientific(), expand_name(), get_accepted_names(), guess_taxo_rank(), list_higher_taxo(), melt_scientificname(), resolve_names(), taxo_fuzzy_match()

Examples

mylist <- data.frame("canonical" = c("Abrothrix longipilis",
  "Acodon hirtus",
  "Akodon longipilis apta",
  "AKODON LONGIPILIS CASTANEUS",
  "Chroeomys jelskii",
  "Acodon jelskii pyrrhotis"),
  stringsAsFactors = FALSE)
melt_canonical(mylist, "canonical", "genus", "species", "subspecies")
Generate a list melting character (comma) separated field values into multiple records

Description

Builds a list, melting character (comma) separated field values given a data frame with a field with repeating values.

Usage

melt_cs_field(data, melt, sepchar = ",", verbose = FALSE)

Arguments

data: data frame containing a data columns with character(comma) separated values
melt: Field name with character(comma) separated values
sepchar: Character separator between the data items. Default is comma
verbose: verbose output, Default: FALSE

Value

a data frame with separate records for each value in field specified

See Also

Other List functions: DwC2taxo(), cast_cs_field(), compact_ids(), get_synonyms(), match_lists(), merge_lists(), syn2taxo(), synonymize_subspecies(), taxo2DwC(), taxo2doc(), taxo2syn(), wiki2taxo()

Examples

scnames <- c("Abrothrix longipilis", "Abrothrix jelskii")
syn_list <- list_itis_syn(scnames)
cs_syn_list <- cast_cs_field(syn_list ,"Name","Syn")
syn_list_new <- melt_cs_field(cs_syn_list,"Syn")
melt_scientificname  Melt scientific name into fields

Description
Parse scientific names into Genus, Species, Subspecies, Author etc.

Usage
melt_scientificname(
  dat,
  sciname = "",
  genus = "genus",
  subgenus = "subgenus",
  species = "species",
  subspecies = "subspecies",
  author = "author",
  verbose = FALSE
)

Arguments
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dat</td>
<td>data frame containing scientific names</td>
</tr>
<tr>
<td>sciname</td>
<td>column name for scientific names, Default: &quot;&quot;</td>
</tr>
<tr>
<td>genus</td>
<td>column name for genus, Default: 'genus'</td>
</tr>
<tr>
<td>subgenus</td>
<td>column name for subgenus, Default: 'subgenus'</td>
</tr>
<tr>
<td>species</td>
<td>column name for species, Default: 'species'</td>
</tr>
<tr>
<td>subspecies</td>
<td>column name for subspecies, Default: 'subspecies'</td>
</tr>
<tr>
<td>author</td>
<td>column name for author, Default: 'author'</td>
</tr>
<tr>
<td>verbose</td>
<td>verbose output, Default: FALSE</td>
</tr>
</tbody>
</table>

Details
Helpful function to break down Scientific names into Genus, Subgenus, species, Subspecies, Author so that the names can be constructed into canonical names for matching

Value
data frame with additional columns for taxonomic fields

See Also
Other Name functions: build_gen_syn(), cast_canonical(), cast_scientificname(), check_scientific(), expand_name(), get_accepted_names(), guess_taxo_rank(), list_higher_taxo(), melt_canonical(), resolve_names(), taxo_fuzzy_match()
**Examples**

```r
mylist <- data.frame("id" = c(11,12,13,14,15,16,17,18,19),
  "scname" = c("Hypochlorosis ancharia (Hewitson, 1869)",
    "Hypochlorosis ancharia ssp. ancharia (Hewitson, 1869)",
    "Hypochlorosis ancharia ssp. humboldti Druce, 1894",
    "Myrina lorquinii C. & R. Felder, 1865",
    "Hypochlorosis ancharia tenebrosa Rothschild, 1915",
    "Hypochlorosis ancharia tenebrosa Rothschild, 1915",
    "Hypochlorosis (Pseudonotis) metilia Fruhstorfer, 1908",
    "Seuku emlongi (Domning et al., 1986)",
    "Sithon lorquinii"),
  stringsAsFactors = FALSE)

melt_scientificname(mylist, sciname="scname", genus="genus",
    subgenus="subgenus", species="species",subspecies="subspecies",
    author="author")
```

---

**merge_lists**

*merge two lists of names*

**Description**

Useful in generating a master list of names from multiple sources

**Usage**

```r
merge_lists(master = NULL, checklist = NULL, output = "all", verbose = TRUE)
```

**Arguments**

- `master`: master list of names
- `checklist`: list to be merged
- `output`: data returned by the function, one of the five options all, onlyadd, add, merged, new or multi. Default all
- `verbose`: verbose output on the console

**Details**

Matches names is checklist with names on master and returns following data:

- `all` = orig + add + new + multi: all the data
- `onlyadd` = add : returns records from checklist that match with master
- `add` = orig + add : returns all records from master + matched records from checklist
- `merged` = orig + add + new : returns all records from master + matched records from checklist + new taxon from checklist
merge_lists

**new** = returns only new taxon entities that did not match with master

**multi** = taxon from checklist for which two synonyms matched with two different accepted names in master

**Value**

Data frame with addition column `merge_tag`. The `merge_tag` contains four possible values.

- **orig** - names in the master
- **add** - checklist names that matched using synonym linkages including direct matches
- **new** - checklist names that did NOT match with master. Potentially new taxa
- **multi** - taxon from checklist for which two synonyms matched with two different accepted names in master

**See Also**

Other List functions: `DwC2taxo()`, `cast_cs_field()`, `compact_ids()`, `get_synonyms()`, `match_lists()`, `melt_cs_field()`, `syn2taxo()`, `synonymize_subspecies()`, `taxo2DwC()`, `taxo2doc()`, `taxo2syn()`, `wiki2taxo()`

**Examples**

```r
master <- data.frame("id" = c(1,2,3),
  "canonical" = c("Hypochlorosis ancharia",
  "Hypochlorosis tenebrosa",
  "Hypochlorosis ancharia tenebrosa"),
  "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae"),
  "accid" = c(0,1,0),
  "source" = c("itis","itis","itis"),
stringsAsFactors = FALSE)

checklist <- data.frame("id" = c(1,2,3,4,5),
  "canonical" = c("Hypochlorosis ancharia",
  "Pseudonotis humboldti",
  "Myrina ancharia",
  "Hypochlorosis ancharia obiana",
  "Hypochlorosis lorquinii"),
  "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae", "Lycaenidae"),
  "accid" = c(0,1,1,0,0),
  "source" = c("itis","wiki","wiki","itis","itis"),
stringsAsFactors = FALSE)

merged_all <- merge_lists(master,checklist,output="all")
new_taxa <- merge_lists(master,checklist,output="new")
merged_with_new <- merge_lists(master,checklist,output="merged")
merged_add <- merge_lists(master,checklist,output="add")
multi_linked <- merge_lists(master,checklist,output="multi")
```
**resolve_names**

Resolve canonical names against GNA

**Description**

Resolve names against Global Names Architecture (GNA) to make sure the name exists

**Usage**

```r
resolve_names(
  taxolist,
  sciname = "canonical",
  score_threshold = 0.98,
  best_match_only = TRUE,
  add_fields = NA,
  verbose = TRUE
)
```

**Arguments**

- **taxolist** *(data frame)*: taxonomic list
- **sciname** (): column name for scientific names
- **score_threshold** *(numeric)*: to make sure names match as desired. Default (0.98) Higher value indicates best match, lower values would return matches at genus level
- **best_match_only** *(logical)*: If TRUE, best match only returned else return all records returned by GNA. Default: TRUE
- **add_fields** *(character)*: One of NA (default), minimal or all. NA adds a logical column 'resolved', Minimal gives back just four fields, whereas all gives all fields back.
- **verbose** *(logical)*: verbose output, Default: FALSE

**Value**

*(data frame)* names list resolves

**See Also**

Other Name functions: `build_gen_syn()`, `cast_canonical()`, `cast_scientificname()`, `check_scientific()`, `expand_name()`, `get_accepted_names()`, `guess_taxo_rank()`, `list_higher_taxo()`, `melt_canonical()`, `melt_scientificname()`, `taxo_fuzzy_match()`
Examples

```r
mylist <- data.frame("canonical" = c("Abrothrix longipilis", "Acodon hirtus", "Akodon longipilis apta", "AKODON LONGIPILIS CASTANEUS", "Chroeomys jelskii", "Acodon jelskii pyrrhotis"), stringsAsFactors = FALSE)
test <- resolve_names(mylist)
test1 <- resolve_names(mylist, add_fields = "minimal")
test2 <- resolve_names(mylist, best_match_only = FALSE, add_fields = "minimal")
test3 <- resolve_names(mylist, best_match_only = FALSE, add_fields = "all")
```

---

**syn2taxo**

*Synonym list to taxolist*

**Description**

Converts a Synonym list with Accepted Names and Synonym columns to taxolist format

**Usage**

```r
syn2taxo(synlist, canonical = "canonical", synonym = "synonym")
```

**Arguments**

- `synlist`: Synonym list with Accepted name (canonical) and Synonym columns
- `canonical`: Accepted names column name, Default: 'canonical'
- `synonym`: Synonym column name , Default: 'synonym'

**Details**

Converts a synonyms list to taxolist format. If order and family fields are present, then they are carried forward else NAs are populated. Duplicate synonyms with same source are removed but with different sources are retained.

**Value**

returns a data frame in taxolist format with all the names in canonical column and accepted names linked to synonyms using id and accid fields. Order, family and (guessed) taxonlevel are added if missing. Genus, species and subspecies fields are added by melting the canonical names.

**See Also**

Other List functions: `DwC2taxo()`, `cast_cs_field()`, `compact_ids()`, `get_synonyms()`, `match_lists()`, `melt_cs_field()`, `merge_lists()`, `synonymize_subspecies()`, `taxo2DwC()`, `taxo2doc()`, `taxo2syn()`, `wiki2taxo()`
Examples

```r
synlist <- data.frame("id" = c(1,2,3),
  "canonical" = c("Hypochlorosis ancharia",
               "Hypochlorosis ancharia",
               "Hypochlorosis ancharia"),
  "synonym" = c("Hypochlorosis tenebrosa",
               "Pseudonotis humboldti",
               "Myrina ancharia"),
  "family" = c("Lycaenidae", "Lycaenidae",
               "Lycaenidae"),
  "source" = c("itis", "wiki", "wiki"),
  stringsAsFactors = FALSE)
mytaxo <- syn2taxo(synlist)
```

**synonymize_subspecies**  
*Convert all subspecies into synonyms of the species*

**Description**

used in generating master lists

**Usage**

```r
synonymize_subspecies(master, return_unmatched = FALSE, verbose = TRUE)
```

**Arguments**

- **master**  
  List of names with a field named canonical
- **return_unmatched**  
  If the return values should be unmatched (orphan) subspecies records. Default: FALSE
- **verbose**  
  display process messages, Default: TRUE

**Details**

While dealing with taxonomic names only at species level, to take advantage of sub-specific names already available in the lists are sometimes treated as synonyms of the names at species rank. To convert all the subspecies names as synonyms this function is very handy. This function will add id, accid and taxonrank columns to return data if missing from original data.

**Value**

Same list of names with id and accid fields added (or data updated the fields exists) with all subspecies linked to the species names as synonyms
See Also

Other List functions: DwC2taxo(), cast_cs_field(), compact_ids(), get_synonyms(), match_lists(), melt_cs_field(), merge_lists(), syn2taxo(), taxo2DwC(), taxo2doc(), taxo2syn(), wiki2taxo()

Examples

master <- data.frame("id" = c(1,2,3,4,5,6,7),
  "canonical" = c("Hypochlorosis ancharia",
    "Hypochlorosis tenebrosa",
    "Pseudonotis humboldti",
    "Myrina ancharia ancharia",
    "Hypochlorosis ancharia tenebrosa",
    "Hypochlorosis ancharia obiana",
    "Hypochlorosis lorquinii"),
  "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae",
    "Lycaenidae", "Lycaenidae", "Lycaenidae",
    "Lycaenidae"),
  "accid" = c(0,1,1,0,0,0,0),
  "source" = c("itis","itis","wiki","wiki","itis",
    "itis","itis"),
  stringsAsFactors = FALSE)
synonymize_subspecies(master)
synonymize_subspecies(master,return_unmatched = TRUE)

taxo2doc

Taxolist to document

Description

Converts a taxolist to a formatted document in html, pdf or word document

Usage

taxo2doc(
  taxolist = NULL,
  genus = NA,
  family = NA,
  title = ",
  addsource = TRUE,
  mastersource = ",
  duplicatesyn = TRUE,
  sourcecol = c("black", "blue"),
  outformat = "html_document",
  outdir = tempdir(),
  outfile = "taxolist.html"
)
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>taxolist</td>
<td>taxolist</td>
</tr>
<tr>
<td>genus</td>
<td>only process for specific genus. Default (&quot;&quot;) implying process all</td>
</tr>
<tr>
<td>family</td>
<td>only process for specific family. Default (&quot;&quot;) implying process all</td>
</tr>
<tr>
<td>title</td>
<td>List title you want to print in output header</td>
</tr>
<tr>
<td>addsource</td>
<td>boolean If the source tag should be displayed. Default TRUE</td>
</tr>
<tr>
<td>mastersource</td>
<td>source string for the master list</td>
</tr>
<tr>
<td>duplicatesyn</td>
<td>boolean if synonyms should to displayed in their alphabetical sorted position too. Default TRUE</td>
</tr>
<tr>
<td>sourcecol</td>
<td>vector of text color values for each source value</td>
</tr>
<tr>
<td>outformat</td>
<td>output format one of &quot;html_document&quot;, &quot;word_document&quot;, &quot;odt_document&quot;, &quot;rtf_document&quot;, &quot;pdf_document&quot;. Default (&quot;html_document&quot;)</td>
</tr>
<tr>
<td>outdir</td>
<td>output directory for the document. Default temporary directory.</td>
</tr>
<tr>
<td>outfile</td>
<td>output file name. Default (&quot;taxolist.html&quot;)</td>
</tr>
</tbody>
</table>

Details

Converts a taxolist to a formatted document in html, pdf or word document making it easy for taxonomist to read through the data

Value

NULL Saves a document file

See Also

Other List functions: DwC2taxo(), cast_cs_field(), compact_ids(), get_synonyms(), match_lists(), melt_cs_field(), merge_lists(), syn2taxo(), synonymize_subspecies(), taxo2DwC(), taxo2syn(), wiki2taxo()

Examples

```r
taxo <- data.frame("id" = c(1,2,3,4,5,6,7),  
  "canonical" = c("Hypochlorosis ancharia",  
  "Hypochlorosis tenebrosa",  
  "Pseudonotis humboldti",  
  "Myrina ancharia",  
  "Hypochlorosis ancharia tenebrosa",  
  "Hypochlorosis ancharia obiana",  
  "Hypochlorosis lorquinii"),  
  "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae",  
  "Lycaenidae", "Lycaenidae", "Lycaenidae",  
  "Lycaenidae"),  
  "accid" = c(0,1,1,0,0,0,0),  
  "source" = c("itis","itis","wiki","wiki","itis",  
  "itis","itis"),
```
taxo2DwC

Taxolist to Darwin Core (DwC)

Description
Converts a taxolist to Darwin Core format

Usage

```r
taxo2DwC(taxolist, verbose = TRUE)
```

Arguments
- `taxolist` taxolist
- `verbose` verbose output, Default: TRUE

Details
Converts a taxolist to Darwin Core format

Value
returns a taxonomic list in DwC format

See Also
Other List functions: `DwC2taxo()`, `cast_cs_field()`, `compact_ids()`, `get_synonyms()`, `match_lists()`, `melt_cs_field()`, `merge_lists()`, `syn2taxo()`, `syonymize_subspecies()`, `taxo2doc()`, `taxo2syn()`, `wiki2taxo()`

Examples

```r
mytaxo <- data.frame("id" = c(1,2,3,4,5,6,7),
  "canonical" = c("Hypochlorosis ancharia",
    "Hypochlorosis tenebrosa",
    "Pseudonotis humboldti",
    "Myrina ancharia",
    "Hypochlorosis ancharia tenebrosa",
    "Hypochlorosis ancharia obiana",
    "Hypochlorosis lorquinii"),
  "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae",
    "Lycaenidae", "Lycaenidae", "Lycaenidae", "Lycaenidae"),
```
taxo2syn

"accid" = c(0,1,1,0,0,0),
"source" = c("itis","itis","wiki","wiki","itis",
        "itis","itis"),
stringsAsFactors = FALSE)

mysynlst <- taxo2DwC(mytaxo)

taxo2syn  Taxolist to Synonym list

Description

Converts a taxolist to Synonym list with Accepted Names and Synonym columns format

Usage

taxo2syn(
  taxolist,
  canonical = "canonical",
  synonym = "synonym",
  duplicate = FALSE,
  sepchar = ","    
)

Arguments

taxolist  taxolist
canonical  names column name, Default: 'canonical'
synonym   Synonym column name to be created, Default: 'synonym'
duplicate If true, duplicate entries are allowed in secondary field
sepchar   Character separator between the data items. Default is comma

Details

Converts a taxolist to synonyms list

Value

returns a synonym list all the names in same column and accepted names linked to synonyms with id and accid fields

See Also

Other List functions: DwC2taxo(), cast_cs_field(), compact_ids(), get_synonyms(), match_lists(), melt_cs_field(), merge_lists(), syn2taxo(), synonymize_subspecies(), taxo2DwC(), taxo2doc(), wiki2taxo()
Examples

mytaxo <- data.frame("id" = c(1,2,3,4,5,6,7),
                   "canonical" = c("Hypochlorosis anchoria",
                                   "Hypochlorosis tenebrosa",
                                   "Pseudonotis humboldti",
                                   "Myrina anchoria",
                                   "Hypochlorosis anchoria tenebrosa",
                                   "Hypochlorosis anchoria obiana",
                                   "Hypochlorosis lorquinii"),
                   "family" = c("Lycaenidae", "Lycaenidae", "Lycaenidae",
                                 "Lycaenidae", "Lycaenidae", "Lycaenidae",
                                 "Lycaenidae"),
                   "accid" = c(0,1,1,1,0,0,0),
                   "source" = c("itis","itis","wiki","wiki","itis",
                                 "itis","itis"),
                   stringsAsFactors = FALSE)

mysynlst <- taxo2syn(mytaxo)

taxo_fuzzy_match

Description

Fuzzy matching with names

Usage

   taxo_fuzzy_match(name, master, dist = 2)

Arguments

   name   Name to search
   master List of names
   dist   Distance tolerance, Default: 2

Details

Fuzzy matching with names in the master list and return best match.

Value

Matched name, string distance and original name. Null if not found.
See Also

Other Name functions: build_gen_syn(), cast_canonical(), cast_scientificname(), check_scientific(), expand_name(), get_accepted_names(), guess_taxo_rank(), list_higher_taxo(), melt_canonical(), melt_scientificname(), resolve_names()

Examples

```r
taxo_fuzzy_match("Acodon hirta", master)
```

---

### Description

Converts the output of `list_wiki_syn` function to taxolist format of `taxotools` package

### Usage

`wiki2taxo(wikisyn)`

### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wikisyn</td>
<td>Wikipedia synonyms list</td>
</tr>
</tbody>
</table>

### Details

Output of `list_wiki_syn` function has different format than taxolist. This function converts it making sure to add additional fields and maintain the synonym linkages.

### Value

`taxolist`

### See Also

Other List functions: DwC2taxo(), cast_cs_field(), compact_ids(), get_synonyms(), match_lists(), melt_cs_field(), merge_lists(), syn2taxo(), synonymize_subspecies(), taxo2DwC(), taxo2doc(), taxo2syn()
Examples

wikilist <- list_wiki_syn("Abrothrix illutea")
wiki2taxo(wikilist)
Index

* ITIS functions
  get_itis_syn, 15
  list_itis_syn, 19
* List functions
  cast_cs_field, 4
  compact_ids, 7
  DwC2taxo, 9
  get_synonyms, 15
  match_lists, 20
  melt_cs_field, 23
  merge_lists, 25
  syn2taxo, 28
  synonymize_subspecies, 29
  taxo2doc, 30
  taxo2DwC, 32
  taxo2syn, 33
  wiki2taxo, 35
* Name functions
  build_gen_syn, 2
  cast_canonical, 3
  cast_scientificname, 5
  check_scientific, 7
  expand_name, 10
  get_accepted_names, 11
  guess_taxo_rank, 17
  list_higher_taxo, 18
  melt_canonical, 21
  melt_scientificname, 24
  resolve_names, 27
  taxo_fuzzy_match, 34
* Wiki functions
  list_wiki_syn, 20
  build_gen_syn, 2, 4, 6, 7, 11, 13, 17, 19, 22, 24, 27, 35
  cast_canonical, 3, 3, 6, 7, 11, 13, 17, 19, 22, 24, 27, 35
  cast_cs_field, 4, 8, 10, 16, 21, 23, 26, 28, 30–33, 35
  cast_scientificname, 3, 4, 5, 7, 11, 13, 17, 19, 22, 24, 27, 35
  check_scientific, 3, 4, 6, 7, 11, 13, 17, 19, 22, 24, 27, 35
  compact_ids, 5, 7, 10, 16, 21, 23, 26, 28, 30–33, 35
  DwC2taxo, 5, 8, 9, 16, 21, 23, 26, 28, 30–33, 35
  expand_name, 3, 4, 6, 7, 10, 13, 17, 19, 22, 24, 27, 35
  get_accepted_names, 3, 4, 6, 7, 11, 13, 17, 19, 22, 24, 30–33, 35
  get_itis_syn, 15, 19
  get_synonyms, 5, 8, 10, 15, 21, 23, 26, 28, 30–33, 35
  guess_taxo_rank, 3, 4, 6, 7, 11, 13, 17, 19, 22, 24, 27, 35
  list_higher_taxo, 3, 4, 6, 7, 11, 13, 17, 18, 22, 24, 27, 35
  list_itis_syn, 15, 19
  list_wiki_syn, 20, 35
  match_lists, 5, 8, 10, 16, 20, 23, 26, 28, 30–33, 35
  melt_canonical, 3, 4, 6, 7, 11, 13, 17, 19, 21, 24, 27, 35
  melt_cs_field, 5, 8, 10, 16, 21, 23, 26, 28, 30–33, 35
  melt_scientificname, 3, 4, 6, 7, 11, 13, 17, 19, 22, 24, 27, 35
  resolve_names, 3, 4, 6, 7, 11, 13, 17, 19, 22, 24, 27, 35
  syn2taxo, 5, 8, 10, 16, 21, 23, 26, 28, 30–33, 35
synonymize_subspecies, 5, 8, 10, 16, 21, 23, 26, 28, 29, 31–33, 35

taxize, 18
taxo2doc, 5, 8, 10, 16, 21, 23, 26, 28, 30, 30, 32, 33, 35
taxo2DwC, 5, 8, 10, 16, 21, 23, 26, 28, 30, 31, 32, 33, 35
taxo2syn, 5, 8, 10, 16, 21, 23, 26, 28, 30–32, 33, 35
taxo_fuzzy_match, 3, 4, 6, 7, 11, 13, 17, 19, 22, 24, 27, 34
taxotools, 35

wiki2taxo, 5, 8, 10, 16, 21, 23, 26, 28, 30–33, 35