Package ‘taxotools’

January 18, 2021

Type Package

Title Tools to Handle Taxonomic Lists

Version 0.0.79

Date 2021-01-16

Maintainer Vijay Barve <`vijay.barve@gmail.com`>

Description Tools include matching and merging taxonomic lists, casting and melting scientific names, managing taxonomic lists from GBIF and ITIS, harvesting names from wikipedia and fuzzy matching.

License CC0

Imports taxize, wikitaxa, plyr, sqldf, stringr, stringdist

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

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 2021-01-18 16:40:02 UTC

R topics documented:

build_gen_syn 2
cast_canonical 3
cast_cs_field 4
check_scientific 5
DwC2taxo 5
expand_name 6
get_accepted_names 7
get_itis_syn 11
guess_taxo_rank 12

list_higher_taxo 12
list_itis_syn 14
list_wiki_syn 14
match_lists 15
melt_canonical 16
melt_cs_field 17
melt_scientificname 18
merge_lists 19
syn2taxo 20
synonymize_subspecies 21
taxotools 22
taxo_fuzzy_match 23
wiki2taxo 24

Index **25**

`build_gen_syn` *Build genetic level synonyms*

Description

Build a genus level synonym list from master list.

Usage

```
build_gen_syn(dat)
```

Arguments

`dat`	master list

Details

This is to #' be passed 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

See Also

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

Examples

```
## Not run:
if(interactive()){
 gen_syn <- build_gen_syn(master)
 }

## End(Not run)
```

`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

```
cast_canonical(
  dat,
  canonical = "canonical",
  genus = "",
  species = "",
  subspecies = ""
)
```

Arguments

`dat`	data frame containing taxonomic list
`canonical`	field name for canonical names
`genus`	field name for Genus field
`species`	field name for Species field
`subspecies`	field name for Subspecies field

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()`, `check_scientific()`, `expand_name()`, `get_accepted_names()`, `guess_taxo_rank()`, `list_higher_taxo()`, `melt_canonical()`, `melt_scientificname()`

Examples

```
## Not run:
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")

## End(Not run)
```

`cast_cs_field` *Build a character (comma) separated List within 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

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

Arguments

`data`	data frame containing primary and secondary data columns
`pri`	Primary field name (repeating values)
`sec`	Secondary field (values would be added to same record, comma separated)
`duplicate`	If true, duplicate entries are allowed in secondary field
`sepchar`	Character separator between the data items. Default is comma

Value

a data frame with two fields Primary and secondary (comma separated list)

See Also

Other List functions: `melt_cs_field()`, `merge_lists()`, `syn2taxo()`, `wiki2taxo()`

Examples

```
## Not run:
scnames <- c("Abrothrix longipilis", "Abrothrix jelskii")
SynList <- list_itis_syn(scnames)
cast_cs_field(SynList,"Name","Syn")

## End(Not run)
```

`check_scientific` *Parse and resolve a scientific name string*

Description

Parse the name using GNR and 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()`, `expand_name()`, `get_accepted_names()`, `guess_taxo_rank()`, `list_higher_taxo()`, `melt_canonical()`, `melt_scientificname()`

Examples

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

`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 • heterotypicSynonym#' • homotypicSynonym
`source`	source of the namelist. Default NA

Details

The name lists downloaded for ITIS website in Darwin Core format has all the required fields. Just needs to be converted and quality checked in terms of missing linkages

Value

names list is taxolist format

See Also

Other list functions: `match_lists()`, `synonymize_subspecies()`

Examples

```
## Not run:
if(interactive()){
 taxolist <- DwC2taxo(namelist)
 }

## End(Not run)
```

`expand_name` *Expands Scientific name*

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

See Also

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

Examples

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

`get_accepted_names` *get_accepted_names*

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,
  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.
`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
`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.

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()`, `check_scientific()`, `expand_name()`, `guess_taxo_rank()`, `list_higher_taxo()`, `melt_canonical()`, `melt_scientificname()`

Examples

```
## Not run:
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 = F)

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

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 = F)

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

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

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 = F)

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

## End(Not run)
```

`get_itis_syn` *Get ITIS Synonyms for a Scientific Name*

Description

Fetch Synonyms using 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

```
## Not run:
get_itis_syn("Abrothrix longipilis")
get_itis_syn("Abditomys latidens")

## End(Not run)
```

`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

See Also

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

Examples

```
guess_taxo_rank("Akodon longipilis")
guess_taxo_rank("Akodon")
guess_taxo_rank("Abrocoma cinerea 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

See Also

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

Examples

```
## Not run:
mylist <- data.frame("canonical" = c("Abrothrix longipilis",
                                    "Mus longipilis",
                                    "Abrothrix jelskii",
                                    "Cardinalis cardinalis",
                                    "Danaus plexippus"),
                    stringsAsFactors = F)

my_taxo_list <- list_higher_taxo(mylist,"canonical")

## End(Not run)
```

`list_itis_syn` *Get ITIS Synonyms for list of names*

Description

Fetch Synonyms from ITIS

Usage

```
list_itis_syn(namelist)
```

Arguments

`namelist`	list of scientific names

Value

a data frame containing names (passed) and synonyms

See Also

Other ITIS functions: `get_itis_syn()`

Examples

```
## Not run:
list_itis_syn("Abrothrix longipilis")
#list_itis_syn(c("Abditomys latidens", "Abeomelomys sevia", "Abrothrix jelskii" ))

## End(Not run)
```

`list_wiki_syn` *Get Wikipedia Synonyms for list of names*

Description

Fetch Synonyms from Wikipedia and clean them for use

Usage

```
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 GBIF backbone taxonomy

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

Examples

```
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

```
match_lists(master, checklist, masterfld, checklistfld)
```

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()`, `synonymize_subspecies()`

Examples

```
## Not run:
master <- data.frame("canonical" = c("Abrothrix longipilis",
                                     "Acodon hirtus",
                                     "Akodon longipilis apta",
                                     "Akodon longipilis castaneus",
                                     "Chroeomys jelskii",
                                     "Acodon jelskii pyrrhotis"),
                    stringsAsFactors = F)
checklist <- data.frame("canonical" = c("Abrothrix longipilis",
                                     "Akodon longipilis apta",
                                     "Akodon longipilis castaneus",
                                     "Abrothrix jelskii",
                                     "Acodon jelskii pyrrhotis"),
                    stringsAsFactors = F)
match_lists(master,checklist,"canonical","canonical")

## End(Not run)
```

`melt_canonical` *Deconstruct canonical names*

Description

Deconstruct canonical names into Genus, Species and Subspecies fields

Usage

```
melt_canonical(dat, canonical = "", genus = "", species = "", subspecies = "")
```

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

Value

a data frame containing Genus, Species and Subspecies fields added or repopulated using data in canonical name field.

See Also

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

Examples

```
## Not run:
mylist <- data.frame("canonical" = c("Abrothrix longipilis",
                                     "Acodon hirtus",
                                     "Akodon longipilis apta",
                                     "AKODON LONGIPILIS CASTANEUS",
                                     "Chroeomys jelskii",
                                     "Acodon jelskii pyrrhotis"),
                    stringsAsFactors = F)
melt_canonical(mylist,"canonical","genus","species","subspecies")

## End(Not run)
```

`melt_cs_field` *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 = ",")
```

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

Value

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

See Also

Other List functions: `cast_cs_field()`, `merge_lists()`, `syn2taxo()`, `wiki2taxo()`

Examples

```
## Not run:
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")

## End(Not run)
```

`melt_scientificname` *Melt scientific name into fields*

Description

Try to 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

`dat`	data frame containing scientific names
`sciname`	column name for scientific names, Default: ”
`genus`	column name for genus, Default: ’genus’
`subgenus`	column name for subgenus, Default: ’subgenus’
`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, species, Subspecies, Author so that the names can be constructed into canonical names for matching

Value

data frame with additional columns for 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()`

Examples

```
## Not run:
if(interactive()){
 melt_scientificname(mylist, sciname="Scientific", genus="genus",
  subgenus="subgenus", species="species",subspecies="subspecies",
  author="author")
 }

## End(Not run)
```

`merge_lists` *merge two lists of names*

Description

Useful in generating a master list of names form multiple sources

Usage

```
merge_lists(master = NULL, checklist = NULL, verbose = TRUE)
```

Arguments

`master`	master list of names
`checklist`	list to be merged
`verbose`	verbose output on the console

Details

Matches names is checklist with names on master

Value

returns three components. First the names to be added, second the names that could not be matched and third the names that matched multiple names in master

See Also

Other List functions: `cast_cs_field()`, `melt_cs_field()`, `syn2taxo()`, `wiki2taxo()`

Examples

```
## Not run:
merge_lists(master = NA,
            checklist = NA,
            verbose = TRUE)

## End(Not run)
```

`syn2taxo` *Synonym list to taxolist*

Description

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

Usage

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

Arguments

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

Details

Converts a synonyms list to taxolist format

Value

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

See Also

Other List functions: `cast_cs_field()`, `melt_cs_field()`, `merge_lists()`, `wiki2taxo()`

Examples

```
## Not run:
if(interactive()){
 taxolist <- syn2taxo(synlist)
 }

## End(Not run)
```

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

Description

used in generating master lists

Usage

```
synonymize_subspecies(master, verbose = FALSE)
```

Arguments

`master`	List of names with a field named canonical
`verbose`	display process messages, Default: FALSE

Details

While dealing with taxonomic names only at specific 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.

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()`, `match_lists()`

Examples

```
## Not run:
newmaster <- synonymize_subspecies(master)

## End(Not run)
```

`taxotools` *taxotools: Tools to handle taxonomic data*

Description

Tools include matching and merging taxonomic lists, casting and melting scientific names, managing taxonomic lists from GBIF and ITIS, harvesting names from wikipedia and fuzzy matching.

List functions

- cast_cs_field
- DwC2taxo
- match_lists
- melt_cs_field
- merge_lists
- syn2taxo
- synonymize_subspecies
- wiki2taxo

Name functions

- build_gen_syn
- cast_canonical
- check_scientific
- expand_name
- get_accepted_names
- guess_taxo_rank
- list_higher_taxo
- melt_canonical
- melt_scientificname
- taxo_fuzzy_match

ITIS functions

- get_itis_syn
- list_itis_syn

Wiki functions

- list_wiki_syn

Citation

- Barve, V., (2021). taxotools: Tools to handle taxonomic data (R package V 0.0.79). Retrieved from https://doi.org/10.5281/zenodo.3934939

`taxo_fuzzy_match` *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.

Examples

```
## Not run:
master <- data.frame("canonical" = c("Abrothrix longipilis",
                                     "Acodon hirtus",
                                     "Akodon longipilis apta",
                                     "Akodon longipilis castaneus",
                                     "Chroeomys jelskii",
                                     "Acodon jelskii pyrrhotis"),
                    stringsAsFactors = F)
 taxo_fuzzy_match("Acodon hirta",master)

## End(Not run)
```

`wiki2taxo` *Wikipedia list to taxo*

Description

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

Usage

```
wiki2taxo(wikisyn)
```

Arguments

`wikisyn`	Wikipedia synonyms list

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: `cast_cs_field()`, `melt_cs_field()`, `merge_lists()`, `syn2taxo()`

Examples

```
## Not run:
if(interactive()){
 wikilist <- list_wiki_syn("Abrothrix illutea")
 wiki2taxo(wikilist)
 }

## End(Not run)
```

Index

* **ITIS functions**
    `get_itis_syn`, 11
    `list_itis_syn`, 14
* **List functions**
    `cast_cs_field`, 4
    `melt_cs_field`, 17
    `merge_lists`, 19
    `syn2taxo`, 20
    `wiki2taxo`, 24
* **Name functions**
    `build_gen_syn`, 2
    `cast_canonical`, 3
    `check_scientific`, 5
    `expand_name`, 6
    `get_accepted_names`, 7
    `guess_taxo_rank`, 12
    `list_higher_taxo`, 12
    `melt_canonical`, 16
    `melt_scientificname`, 18
* **Wiki functions**
    `list_wiki_syn`, 14
* **list functions**
    `DwC2taxo`, 5
    `match_lists`, 15
    `synonymize_subspecies`, 21

`build_gen_syn`, 2, *3*, *5*, *7*, *9*, *12*, *13*, *16*, *18*, *22*

`cast_canonical`, *2*, 3, *5*, *7*, *9*, *12*, *13*, *16*, *18*, *22*
`cast_cs_field`, 4, *17*, *19*, *20*, *22*, *24*
`check_scientific`, *2*, *3*, 5, *7*, *9*, *12*, *13*, *16*, *18*, *22*

`DwC2taxo`, 5, *15*, *21*, *22*

`expand_name`, *2*, *3*, *5*, 6, *9*, *12*, *13*, *16*, *18*, *22*

`get_accepted_names`, *2*, *3*, *5*, *7*, 7, *12*, *13*, *16*, *18*, *22*
`get_itis_syn`, 11, *14*, *22*
`guess_taxo_rank`, *2*, *3*, *5*, *7*, *9*, 12, *13*, *16*, *18*, *22*

`list_higher_taxo`, *2*, *3*, *5*, *7*, *9*, *12*, 12, *16*, *18*, *22*
`list_itis_syn`, *11*, 14, *22*
`list_wiki_syn`, 14, *22*, *24*

`match_lists`, *6*, 15, *21*, *22*
`melt_canonical`, *2*, *3*, *5*, *7*, *9*, *12*, *13*, 16, *18*, *22*
`melt_cs_field`, *4*, 17, *19*, *20*, *22*, *24*
`melt_scientificname`, *2*, *3*, *5*, *7*, *9*, *12*, *13*, *16*, 18, *22*
`merge_lists`, *4*, *17*, 19, *20*, *22*, *24*

`syn2taxo`, *4*, *17*, *19*, 20, *22*, *24*
`synonymize_subspecies`, *6*, *15*, 21, *22*

`taxize`, *13*
`taxo_fuzzy_match`, *22*, 23
`taxotools`, 22, *24*

`wiki2taxo`, *4*, *17*, *19*, *20*, *22*, 24