Package: taxizedb (via r-universe)

March 3, 2025

Type Package

Title Tools for Working with 'Taxonomic' Databases

Description Tools for working with 'taxonomic' databases, including utilities for downloading databases, loading them into various 'SQL' databases, cleaning up files, and providing a 'SQL' connection that can be used to do 'SQL' queries directly or used in 'dplyr'.

Version 0.3.1

URL https://docs.ropensci.org/taxizedb/, https://github.com/ropensci/taxizedb

BugReports https://github.com/ropensci/taxizedb/issues

License MIT + file LICENSE

Roxygen list(markdown = TRUE)

Encoding UTF-8

Language en-US

Imports curl (>= 2.4), DBI (>= 0.6-1), RSQLite (>= 1.1.2), dplyr (>= 0.7.0), tibble, rlang, readr (>= 1.1.1), dbplyr (>= 1.0.0), magrittr (>= 1.5), hoardr (>= 0.1.0)

Suggests testthat, taxize

RoxygenNote 7.2.3

Config/pak/sysreqs libicu-dev libssl-dev libx11-dev

Repository https://r-multiverse-staging.r-universe.dev

RemoteUrl https://github.com/ropensci/taxizedb

RemoteRef 5732f9777142cd0034b64d5ae8b9aea0605cfaab

RemoteSha 5732f9777142cd0034b64d5ae8b9aea0605cfaab

Contents

`taxizedb-package` *taxizedb*

Description

Taxonomic databases interface

Supported data sources and database structure

All are using SQLite as the database

- NCBI: text files are provided by NCBI, which we stitch into a sqlite db
- ITIS: they provide a sqlite dump, which we use here
- The PlantList: created from stitching together csv files. this source is no longer updated as far as we can tell. they say they've moved focus to the World Flora Online
- Catalogue of Life: created from Darwin Core Archive dump. Using the latest monthly edition via http://www.catalogueoflife.org/DCA_Export/archive.php
- GBIF: created from Darwin Core Archive dump. right now we only have the taxonomy table (called gbif), but will add the other tables in the darwin core archive later
- Wikidata: aggregated taxonomy of Open Tree of Life, GLoBI and Wikidata. On Zenodo, created by Joritt Poelen of GLOBI.
- World Flora Online: http://www.worldfloraonline.org/

Update schedule for databases

- NCBI: since `db_download_ncbi` creates the database when the function is called, it's updated whenever you run the function
- ITIS: since ITIS provides the sqlite database as a download, you can delete the old file and run `db_download_itis` to get a new dump; they I think update the dumps every month or so
- The PlantList: no longer updated, so you shouldn't need to download this after the first download
- Catalogue of Life: a GitHub Actions job runs once a day at 00:00 UTC, building the lastest COL data into a SQLite database thats hosted on Amazon S3
- GBIF: a GitHub Actions job runs once a day at 00:00 UTC, building the lastest COL data into a SQLite database thats hosted on Amazon S3
- Wikidata: last updated April 6, 2018. Scripts are available to update the data if you prefer to do it yourself.
- World Flora Online: since `db_download_wfo` creates the database when the function is called, it's updated whenever you run the function

Links

- NCBI: ftp://ftp.ncbi.nih.gov/pub/taxonomy/
- ITIS: https://www.itis.gov/downloads/index.html
- The PlantList - http://www.theplantlist.org/
- Catalogue of Life: via http://www.catalogueoflife.org/content/annual-checklist-archive
- GBIF: http://rs.gbif.org/datasets/backbone/
- Wikidata: https://zenodo.org/record/1213477
- World Flora Online: http://www.worldfloraonline.org/

Examples

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

# data source: NCBI
db_download_ncbi()
src <- src_ncbi()
df <- tbl(src, "names")
filter(df, name_class == "scientific name")

# data source: ITIS
## download ITIS database
db_download_itis()
## connect to the ITIS database
src <- src_itis()
## use SQL syntax
sql_collect(src, "select * from hierarchy limit 5")
### or pipe the src to sql_collect
src %>% sql_collect("select * from hierarchy limit 5")
```

```
## use dplyr verbs
src %>%
  tbl("hierarchy") %>%
  filter(ChildrenCount > 1000)
## or create tbl object for repeated use
hiers <- src %>% tbl("hierarchy")
hiers %>% select(TSN, level)

# data source: The PlantList
## download tpl datababase
db_download_tpl()
## connecto the tpl database
src <- src_tpl()
## do queries
tpl <- tbl(src, "tpl")
filter(tpl, Family == "Pinaceae")

# data source: Catalogue of Life
## download col datababase
db_download_col()
## connec to the col database
src <- src_col()
## do queries
names <- tbl(src, "taxa")
select(names, taxonID, scientificName)

# data source: GBIF
## download gbif datababase
db_download_gbif()
## connecto the gbif database
src <- src_gbif()
## do queries
df <- tbl(src, "gbif")
select(df, taxonID, scientificName)

# data source: Wikidata
db_download_wikidata()
src <- src_wikidata()
df <- tbl(src, "wikidata")
filter(df, rank_id == "Q7432")

# data source: World Flora Online
db_download_wfo()
src <- src_wfo()
df <- tbl(src, "wfo")
filter(df, taxonID == "wfo-0000000010")

## End(Not run)
```

`children` *Retrieve immediate descendents of a taxon*

Description

Retrieve immediate descendents of a taxon

Usage

```
children(x, db = "ncbi", verbose = TRUE, ...)
```

Arguments

x	(character) Vector of taxon keys for the given database
db	(character) The database to search, one of ncbi, itis, gbif, col, or wfo
verbose	(logical) Print verbose messages
...	Additional arguments passed to database specific function.

Value

list of tibbles with the columns: id, name, rank. This is exactly equivalent to the output of `taxize::children()`

Examples

```
## Not run:
children(c(3700, 2))
children(c(154395, 154357), db="itis")
children("wfo-4000032377", db="wfo")
children(2877951, db="gbif")
children(3960765, db="col") # Abies

## End(Not run)
```

`classification` *Retrieve the taxonomic hierarchies from local database*

Description

This function is equivalent to the `taxize::classification()` function, except that it uses a local database (so is much faster). The output is identical to `taxize::classification()`

Usage

```
classification(x, db = "ncbi", verbose = TRUE, ...)
```

Arguments

x	character) Vector of taxon keys for the given database
db	character) The database to search, one of ncbi, itis, gbif, col, or wfo
verbose	(logical) Print verbose messages
...	Additional arguments passed to database specific classification functions.

Value

list of data.frames with the columns: name, rank, and id. This is exactly equivalent to the output of `taxize::classification()`

Examples

```
## Not run:
classification(c(3702, 9606))
classification(c(154395, 154357), db="itis")
classification(c("wfo-0000291463", "wfo-7000000057"), db="wfo")
classification(2878586, db="gbif")
classification(c(2878586, 2704179), db="gbif")
classification(3960765, db="col") # Abies

## End(Not run)
```

`db_download` *Download taxonomic databases*

Description

Download taxonomic databases

Usage

```
db_download_ncbi(verbose = TRUE, overwrite = FALSE)

db_download_itis(verbose = TRUE, overwrite = FALSE)

db_download_tpl(verbose = TRUE, overwrite = FALSE)

db_download_wfo(verbose = TRUE, overwrite = FALSE)

db_download_col(verbose = TRUE, overwrite = FALSE)

db_download_gbif(verbose = TRUE, overwrite = FALSE)

db_download_wikidata(verbose = TRUE, overwrite = FALSE)
```

Arguments

`verbose`	(logical) Print messages. Default: `TRUE`
`overwrite`	(logical) If `TRUE` force an update by overwriting previously downloaded data. Default: `FALSE`

Details

Downloads sql database, cleans up unneeded files, returns path to sql file

Value

(character) path to the downloaded SQL database

See Also

tdb_cache

Examples

```
## Not run:
# ITIS
# db_download_itis()
# src_itis()

# Plantlist
# db_download_tpl()
# db_download_tpl(overwrite=TRUE) # overwrite - download again
# src_tpl()

# COL
# db_download_col()
# src_col()

# GBIF
# db_download_gbif()
# src_gbif()

# NCBI
# db_download_ncbi()
# src_ncbi()

# Wikidata
# db_download_wikidata()
# db_download_wikidata(overwrite=TRUE) # overwrite - download again
# src_wikidata()

# World Flora Online
# db_download_wfo()
# src_wfo()

## End(Not run)
```

---

db_load-defunct *Load taxonomic databases - NO LONGER NEEDED*

Description

Use db_download then src_taxizedb

Usage

```
db_load_itis(...)

db_load_tpl(...)

db_load_col(...)

db_load_gbif(...)

db_load_ncbi(...)

db_load_wikidata(...)
```

Arguments

...	ignored

---

db_path *database path*

Description

database path

Usage

```
db_path(db)
```

Arguments

db	(character) db name. one of: itis, tpl, col, gbif, ncbi, wikidata, wfo. required

---

downstream *Retrieve all taxa descending from a vector of taxa*

Description

This function is nearly equivalent to the `taxize::downstream()` function

Usage

```
downstream(x, db = "ncbi", verbose = TRUE, ...)
```

Arguments

x	(character) Vector of taxon keys for the given database
db	(character) The database to search, one of ncbi, itis, gbif, col, or wfo
verbose	(logical) Print verbose messages
...	Additional arguments passed to database specific downstream functions

Value

list of data.frames with the columns: childtaxa_id, childtaxa_name, and rank. This is exactly equivalent to the output of `taxize::downstream()`

Examples

```
## Not run:
# get descendents from all ranks
# downstream(c(3700, 9605)) # takes a while

# limit results to species
downstream(c(3700, 9605), downto='species')

# allow ambiguous nodes but no ambiguous species
downstream(
  c(3700, 9605),
  downto='species',
  ambiguous_nodes=FALSE,
  ambiguous_species=TRUE
)

# ITIS
id <- name2taxid('Aves', db = "itis")
downstream(id, db = "itis", downto = "family")
downstream(id, db = "itis", downto = "genus")
id <- name2taxid('Bombus', db = "itis")
downstream(id, db = "itis", downto = "species")

# COL
id <- name2taxid('Chordata', db = "col")
downstream(id, db = "col", downto = "family")

# GBIF
id <- name2taxid('Pinaceae', db = "gbif")
downstream(id, db = "gbif", downto = "genus")

# World Flora Online
id <- name2taxid('Pinaceae', db = "wfo")
downstream(id, db = "wfo", downto = "species")

## End(Not run)
```

name2taxid *Convert species names to taxon IDs*

Description

name2taxid() returns a vector and dies if there are any ambiguous names. name2taxid_map() returns a data.frame mapping names to ids

Usage

```
name2taxid(x, db = "ncbi", verbose = TRUE, out_type = c("uid", "summary"), ...)
```

Arguments

x	(character) Vector of taxon keys for the given database
db	(character) The database to search, one of ncbi, itis, gbif, wfo, or tpl
verbose	(logical) Print verbose messages
out_type	(logical) character "uid" for an ID vector, "summary" for a table with columns 'tax_id' and 'tax_name'.
...	Additional arguments passed to database specific classification functions.

NCBI database

The NCBI taxonomy database includes common names, synonyms and misspellings. However, the database is a little inconsistent. For some species, such as Arabidopsis thaliana, the misspelling Arabidopsis_thaliana is included, but the same is NOT done for humans. However, underscores are supported when querying through entrez, as is done in taxize, which implies entrez is replacing underscores with spaces. So I do the same. A corner case appears when an organism uses underscores as part of the name, not just a standin for space ("haloarchaeon 3A1_DGR"). To deal with this case, we replace underscores with spaces ONLY if there are not spaces in the original name.

Examples

```
## Not run:
name2taxid(c('Arabidopsis thaliana', 'pig'))
name2taxid(c('Arabidopsis thaliana', 'pig'), out_type="summary")
name2taxid(x=c('Arabidopsis thaliana', 'Apis mellifera'), db = "itis")
name2taxid(x=c('Arabidopsis thaliana', 'Apis mellifera'), db = "itis",
 out_type="summary")
name2taxid(x=c('Arabidopsis thaliana', 'Quercus kelloggii'), db = "wfo")
name2taxid(x=c('Arabidopsis thaliana', 'Quercus kelloggii'), db = "wfo",
 out_type="summary")
name2taxid("Austrobaileyaceae", db = "wfo")
name2taxid("Quercus kelloggii", db = "gbif")
name2taxid(c("Quercus", "Fabaceae", "Animalia"), db = "gbif")
name2taxid(c("Abies", "Pinales", "Tracheophyta"), db = "col")
name2taxid(c("Abies mangifica", "Acanthopale aethiogermanica",
```

```
  "Acanthopale albosetulosa"), db = "tpl")

## End(Not run)
```

`sql_collect` *Query and get data back into a data.frame*

Description

Query and get data back into a data.frame

Usage

```
sql_collect(src, query, ...)
```

Arguments

`src`	(src) An `src` object, result of calling `src_itis()`, `src_col()`, or `src_tpl()`
`query`	(character) A SQL query
`...`	further args passed on to `dplyr::tbl()`

Details

we run `dplyr::tbl()`, then `dplyr::collect()`

Examples

```
## Not run:
src <- src_itis()
sql_collect(src, "select * from hierarchy limit 5")
## or pipe the src to sql_collect
src %>% sql_collect("select * from hierarchy limit 5")

## End(Not run)
```

`src_taxizedb` *src - dplyr src objects*

Description

src - dplyr src objects

Usage

```
src_itis(path = db_path("itis"), ...)

src_tpl(path = db_path("tpl"), ...)

src_col(path = db_path("col"), ...)

src_gbif(path = db_path("gbif"), ...)

src_ncbi(path = db_path("ncbi"), ...)

src_wikidata(path = db_path("wikidata"), ...)

src_wfo(path = db_path("wfo"), ...)
```

Arguments

path	(character) path to SQLite database. by default we use the function db_path() to get the path
...	Further args passed on to DBI::dbConnect()

Value

an src object

Examples

```
## Not run:
# src_itis()
# src_tpl()
# src_col()
# src_gbif()
# src_ncbi()
# src_wikidata()
# src_wfo()

## End(Not run)
```

---

taxa_at *Get taxa at specific scientific ranks*

Description

Get taxa at specific scientific ranks

Usage

```
taxa_at(
  x,
  rank,
  db = "ncbi",
  missing = "lower",
  verbose = TRUE,
  warn = TRUE,
  ...
)
```

Arguments

x	(character) Vector of taxon keys (ids) for the given database. required
rank	(character) A target rank for which to fetch data. required
db	(character) The database to search, one of ncbi, itis, gbif, col, or wfo
missing	(character) if no data found at the given rank and input key, should we get the next closest lower than that given in rank, or higher. one of lower (default), higher.
verbose	(logical) Print verbose messages
warn	(logical) If TRUE, raise a warning if any taxon IDs can not be found
...	Additional arguments passed to database specific classification functions

Value

list of data.frame's for each input taxon key, where each data.frame has fields: name, rank, id. When no results found, an empty data.frame

Examples

```
## Not run:
taxa_at(186803, rank = "order", db="ncbi", missing = "lower")
taxa_at(c(186803, 541000, 216572, 186804, 31979,  186806),
 rank = "family", missing = "lower")
taxa_at(c(154395, 154357, 23041, 154396), rank = "family", db="itis")
taxa_at(c('wfo-4000032377', 'wfo-0000541830'), rank = "family", db="wfo")
taxa_at("wfo-7000000057", rank = "order", db="wfo")
taxa_at(2877951, rank = "phylum", db="gbif")
taxa_at(c(2877951, 5386), rank = "family", db="gbif")
taxa_at(c(3960765, 3953606, 3953010), rank = "family", db="col")

## End(Not run)
```

taxid2name *Convert taxon IDs to scientific names*

Description

Convert taxon IDs to scientific names

Usage

```
taxid2name(x, db = "ncbi", verbose = TRUE, warn = TRUE, ...)
```

Arguments

x	(character) Vector of taxon keys for the given database
db	(character) The database to search, one of ncbi, itis, gbif, col, wfo, or tpl
verbose	(logical) Print verbose messages
warn	(logical) If TRUE, raise a warning if any taxon IDs can not be found
...	Additional arguments passed to database specific classification functions

Value

character vector of scientific names

Examples

```
## Not run:
taxid2name(c(3702, 9606))
taxid2name(c(154395, 154357, 23041, 154396), db = "itis")
taxid2name(c('wfo-0000541830', 'wfo-0000291463'), db = "wfo")
taxid2name("wfo-7000000057", db="wfo")
taxid2name(2877951, db="gbif")
taxid2name(c(2877951, 5386), db="gbif")
taxid2name(c(3960765, 3953606, 3953010), db="col")
taxid2name(c("kew-2614538", "kew-2895433", "kew-2615007"), db="tpl")

## End(Not run)
```

taxid2rank *Convert taxon IDs to scientific ranks*

Description

Convert taxon IDs to scientific ranks

Usage

```
taxid2rank(x, db = "ncbi", verbose = TRUE, warn = TRUE, ...)
```

Arguments

x	(character) Vector of taxon keys (name or id) for the given database
db	(character) The database to search, one of ncbi, itis, gbif, col, or wfo
verbose	(logical) Print verbose messages
warn	(logical) If TRUE, raise a warning if any taxon IDs can not be found
...	Additional arguments passed to database specific classification functions

Value

character vector of ranks in the same order as the inputs

Examples

```
## Not run:
taxid2rank(c(3701, 9606))
taxid2rank(c(154395, 154357, 23041, 154396), db="itis")
taxid2rank(c('wfo-4000032377', 'wfo-0000541830'), db="wfo")
taxid2rank("wfo-7000000057", db="wfo")
taxid2rank(2877951, db="gbif")
taxid2rank(c(2877951, 5386), db="gbif")
taxid2rank(c(3960765, 3953606, 3953010), db="col")

## End(Not run)
```

tdb_cache *Caching*

Description

Manage cached taxizedb files with **hoardr**

Details

`cache_delete` only accepts 1 file name, while `cache_delete_all` doesn't accept any names, but deletes all files. For deleting many specific files, use `cache_delete` in a `lapply()` type call

Useful user functions

- `tdb_cache$cache_path_get()` get cache path
- `tdb_cache$cache_path_set()` set cache path
- `tdb_cache$list()` returns a character vector of full path file names
- `tdb_cache$files()` returns file objects with metadata
- `tdb_cache$details()` returns files with details
- `tdb_cache$delete()` delete specific files
- `tdb_cache$delete_all()` delete all files, returns nothing

Examples

```
## Not run:
tdb_cache

# list files in cache
tdb_cache$list()

# delete certain database files
# tdb_cache$delete("file path")
# tdb_cache$list()

# delete all files in cache
# tdb_cache$delete_all()
# tdb_cache$list()

## End(Not run)
```

Index