

# Package ‘taxonbridge’

July 4, 2022

**Title** Create Custom Taxonomies Based on the NCBI Taxonomy and GBIF Backbone Taxonomy

**Version** 1.2.2

**Description** The NCBI taxonomy is a popular resource for taxonomic studies but it only contains data on species with sequence data whereas the GBIF has a more extensive coverage of extinct species. Taxonbridge is useful for the creation and analysis of custom taxonomies based on the NCBI taxonomy and GBIF backbone taxonomy.

**License** CC0

**URL** <https://github.com/MoultDB/taxonbridge>

**BugReports** <https://github.com/MoultDB/taxonbridge/issues>

**Encoding** UTF-8

**RoxygenNote** 7.1.2

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Imports** purrr, dplyr, vroom, ggplot2, rje, withr, utils, stringr

**Depends** R (>= 2.10)

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Werner Veldsman [aut, cre] (<<https://orcid.org/0000-0001-9837-8332>>),  
Giulia Campli [aut],  
Sagane Dind [aut],  
Valentine Rech de Laval [aut] (<<https://orcid.org/0000-0002-3020-1490>>),  
Harriet Drage [aut] (<<https://orcid.org/0000-0002-0759-5970>>),  
Robert Waterhouse [aut] (<<https://orcid.org/0000-0003-4199-9052>>),  
Marc Robinson-Rechavi [aut] (<<https://orcid.org/0000-0001-9837-8332>>)

**Maintainer** Werner Veldsman <[wernerpieter.veldsman@unil.ch](mailto:wernerpieter.veldsman@unil.ch)>

**Repository** CRAN

**Date/Publication** 2022-07-04 12:10:02 UTC

## R topics documented:

annotate . . . . .	2
dedupe . . . . .	3
download_gbif . . . . .	4
download_ncbi . . . . .	4
fuzzy_search . . . . .	5
get_inconsistencies . . . . .	6
get_lineages . . . . .	7
get_status . . . . .	8
get_taxa . . . . .	8
get_validity . . . . .	9
load_population . . . . .	10
load_sample . . . . .	11
load_taxonomies . . . . .	11
plot_mdb . . . . .	12
prepare_comparable_rank_dist . . . . .	13
prepare_rank_dist . . . . .	13
term_conversion . . . . .	14
<b>Index</b>	<b>15</b>

---

annotate	<i>Annotate a custom taxonomy</i>
----------	-----------------------------------

---

### Description

Annotate a custom taxonomy

### Usage

```
annotate(x, names, new_column, present = "1", absent = NA)
```

### Arguments

x	A tibble with taxonomic data to be annotated.
names	A character vector containing scientific names that will be matched to scientific names in x.
new_column	A string to be the name of a new column that will contain annotations.
present	A string with the annotation in the case of a match (Defaults to "1").
absent	A string with the annotation in case of no match (Defaults to NA).

**Details**

This method takes as input a character vector with scientific names. If the scientific name(s) in the vector match with scientific names in the tibble, a new column will be created and an annotation of choice will be added to the relevant row in the new column. This method is useful for annotating scientific names with identified ambiguity, duplication or any other characteristic. The character vector could, for example, even contain scientific names that have not been derived with a Taxonbridge method.

**Value**

A tibble that contains an additional column with annotations.

**Examples**

```
sample <- load_sample()
lineages <- get_lineages(sample)
kingdom <- get_validity(lineages, rank = "kingdom", valid = FALSE)
family <- get_validity(lineages, rank = "family", valid = FALSE)
candidates <- list(kingdom, family)
binomials <- get_inconsistencies(candidates, uninomials = FALSE, set = "intersect")
x <- annotate(sample, binomials, new_column = "inconsistencies", "Accepted but ambiguous")
x[!is.na(x$inconsistencies),c("inconsistencies")]
```

---

dedupe

*Remove duplicate scientific names in a taxonomy*


---

**Description**

Remove duplicate scientific names in a taxonomy

**Usage**

```
dedupe(x, ranked = TRUE)
```

**Arguments**

x	A tibble created with <code>load_taxonomies()</code> or <code>load_population()</code> or <code>load_sample()</code> .
ranked	A logical indicating whether duplicates should be removed by certainty of taxonomic status. Defaults to TRUE.

**Details**

This method can be used in one of two ways. By simply passing a tibble as input, duplicates will be stringently removed based on the following order: "accepted", "synonym", "homotypic synonym", "heterotypic synonym", "partite synonym", "doubtful", NA. If however the ranked parameter is set to FALSE, duplicates will only be removed based on the scientific names, but not on taxonomic status, which results in less duplicates being removed.

**Value**

A de-duplicated tibble

**Examples**

```
dedupe(load_sample())
```

---

download_gbif	<i>Download the GBIF backbone taxonomy</i>
---------------	--

---

**Description**

Download the GBIF backbone taxonomy

**Usage**

```
download_gbif()
```

**Details**

This method downloads the GBIF backbone taxonomy archive file to a temporary directory, extracts `Taxon.tsv` from the downloaded archive file, and then removes the archive file.

**Value**

A string containing the path to `Taxon.tsv`.

**Examples**

```
## Not run: download_gbif()
```

---

download_ncbi	<i>Download the NCBI taxonomy</i>
---------------	-----------------------------------

---

**Description**

Download the NCBI taxonomy

**Usage**

```
download_ncbi(taxonkitpath = NA)
```

**Arguments**

`taxonkitpath` A string containing the full path to where `Taxonkit` is installed (optional).

## Details

This method downloads a NCBI taxonomy archive file to a temporary directory, extracts four files (nodes.dmp, names.dmp, merged.dmp and delnodes.dmp) from the downloaded archive file, and then removes the archive file. Further parsing of these four files must be carried out with Taxonkit (<https://bioinf.shenwei.me/taxonkit/download/>), either automatically or manually. If the path to a Taxonkit installation is supplied, Taxonkit is called and the location of the four files is passed to Taxonkit as an argument for automatic parsing. Taxonkit output is saved in the same temporary folder in a file called All.lineages.tsv.gz. If the path to Taxonkit is not supplied, parsing should be carried out manually using the command: `taxonkit list --data-dir=path/to/downloaded/files --ids 1 |`

## Value

A character vector containing paths to the relevant downloaded and unzipped NCBI data dump files, or if the `taxonkitpath` parameter was set, the path to `All.lineages.tsv.gz`.

## Examples

```
## Not run: download_ncbi()
## Not run: download_ncbi(taxonkitpath = "/home/usr/bin/taxonkit")
```

---

fuzzy_search	<i>Match misspelled or partial scientific names</i>
--------------	---

---

## Description

Match misspelled or partial scientific names

## Usage

```
fuzzy_search(
  x,
  term,
  sensitivity = 0,
  allow_term_removal = FALSE,
  force_binomial = FALSE
)
```

## Arguments

<code>x</code>	A tibble created with <code>load_taxonomies()</code> or <code>load_population()</code> or <code>load_sample()</code> .
<code>term</code>	A string consisting of a scientific name.
<code>sensitivity</code>	An integer representing character mismatch tolerance. Defaults to intolerant i.e. <code>sensitivity=0</code> .
<code>allow_term_removal</code>	A logical indicating whether searches against only the first word of <code>term</code> should be carried out if no matches are found. Defaults to <code>FALSE</code> .
<code>force_binomial</code>	A logical indicating whether <code>term</code> should be stripped to a maximum of two words. Defaults to <code>FALSE</code> .

**Details**

The sensitivity parameter sets the number of character mismatches that are tolerated for a match to be reported. The higher the sensitivity, the more matches will be found, but the less relevant they may be. The `allow_term_removal` parameter allows stripping the search query to only retain the characters before the first occurrence of a white space i.e. only the first word of a search query is used during the search. This is useful when "Genus sp." or "Genus indet." is the search query. However, `fuzzy_search()` will always search using the entire search query first and then only proceed to strip terms if no hits are found. On the other hand, if `force_binomial` is set to `TRUE`, the search query will first be limited to the first two words before searching commences. This in turn is useful if the search query includes credit to the publisher e.g. "Birgus latro (Linnaeus, 1767)" or to prevent subspecies names (so-called trinomials) from leading to a match not being made.

**Value**

A list of candidate match(es), if applicable.

**Examples**

```
fuzzy_search(load_sample(), "Miacis deutsch")
fuzzy_search(load_sample(), "Miacis sp.", allow_term_removal = TRUE)
fuzzy_search(load_sample(), "Miacus deutsch", sensitivity = 1)
fuzzy_search(load_sample(), "Miacis deutsch (Smith, 2022)", force_binomial = TRUE)
```

---

get_inconsistencies	<i>Detect candidate inconsistencies and ambiguity between NCBI and GBIF data</i>
---------------------	--

---

**Description**

Detect candidate inconsistencies and ambiguity between NCBI and GBIF data

**Usage**

```
get_inconsistencies(x, uninomials = TRUE, set = "intersect")
```

**Arguments**

x	A <b>list</b> consisting of two tibbles of different ranks that have been passed to <code>get_validated(..., rank = ...)</code> .
uninomials	A logical indicating whether uninomials should be included in the detection. Defaults to <code>TRUE</code> . Note: uninomials are single names (e.g., "Coenobitidae").
set	The type of set operation to be performed on x ("intersect", "union", or "setdiff"). Defaults to <code>intersect</code> . Note: the set difference ("setdiff") argument is order dependent.

**Details**

This method will return the intersect, union, or set difference of a list of two tibbles, and is meant to be used on lists of tibbles that have already been processed with `get_validity()`. A list consisting of a single tibble may be passed to this method for the purpose of retrieving a character vector containing scientific names, however, set operations do not apply to lists consisting of single tibbles.

**Value**

A character vector containing scientific names that exhibit inconsistency or ambiguity.

**Examples**

```
sample <- load_sample()
lineages <- get_lineages(sample)
kingdom <- get_validity(lineages, rank = "kingdom", valid = FALSE)
family <- get_validity(lineages, rank = "family", valid = FALSE)
candidates <- list(kingdom, family)
get_inconsistencies(candidates, uninomials = FALSE, set = "intersect")
```

---

get_lineages	<i>Get entries that have lineage information for both the GBIF and NCBI data</i>
--------------	--

---

**Description**

Get entries that have lineage information for both the GBIF and NCBI data

**Usage**

```
get_lineages(x)
```

**Arguments**

x                    A tibble created with `load_taxonomies()` or `load_population()` or `load_sample()`.

**Value**

A tibble with complete lineage data.

**Examples**

```
get_lineages(load_sample())
```

---

get_status	<i>Filter a custom taxonomy by GBIF taxonomic status/synonym</i>
------------	--

---

**Description**

Filter a custom taxonomy by GBIF taxonomic status/synonym

**Usage**

```
get_status(x, status = "all")
```

**Arguments**

x	A tibble created with load_taxonomies() or load_population() or load_sample().
status	Filter on GBIF assigned status (i.e. NA, "doubtful", "accepted", "proparte synonym", "synonym", "homotypic synonym", "heterotypic synonym"). Can be a string or a vector of strings. Defaults to no filtering.

**Value**

A filtered tibble.

**Examples**

```
get_status(load_sample(), "synonym")
get_status(load_sample(), c("accepted", "doubtful"))
```

---

get_taxa	<i>A helper function to filter on GBIF and NCBI taxa names</i>
----------	--

---

**Description**

A helper function to filter on GBIF and NCBI taxa names

**Usage**

```
get_taxa(
  x,
  kingdom = NA,
  phylum = NA,
  class = NA,
  order = NA,
  family = NA,
  genus = NA,
  species = NA
)
```



**Arguments**

x	A tibble created with load_taxonomies() or load_population() or load_sample().
kingdom	A string consisting of a scientific name.
phylum	A string consisting of a scientific name.
class	A string consisting of a scientific name.
order	A string consisting of a scientific name.
family	A string consisting of a scientific name.
genus	A string consisting of a scientific name.
species	A string consisting of a scientific name.

**Details**

This method will return results if the scientific name of interest is found in either the GBIF or the NCBI. The scientific name does not have to be case sensitive.

**Value**

A filtered tibble.

**Examples**

```
get_taxa(load_sample(), species = "hyalina")
get_taxa(load_sample(), phylum = "ArthroPODA", genus = "BirGus")
```

---

get_validity	<i>Validate entries of a merged taxonomy</i>
--------------	--

---

**Description**

Validate entries of a merged taxonomy

**Usage**

```
get_validity(x, rank = "family", valid = TRUE)
```

**Arguments**

x	A tibble created with load_taxonomies() or load_population() or load_sample().
rank	A string with GBIF rank that will be used to examine a NCBI lineage for validation purposes. Must be kingdom, phylum, class, order or family. Defaults to family. Note: If kingdom is used, the term_conversion() method should first be applied.
valid	A logical indicating whether the returned data should include valid or invalid entries (defaults to TRUE).

**Details**

Taxonbridge matches NCBI and GBIF data by scientific name. This method will use the GBIF rank (kingdom, phylum, class, order or family) and search for this rank name in the matched NCBI lineage. The purpose is to detect scientific names that have different lineage data in the GBIF and NCBI. If the `valid` parameter is set to `TRUE`, this method will not only check the rank names, but also ensure that the GBIF `taxonRank` column and NCBI `ncbi_rank` column matches.

**Value**

A validated tibble.

**Examples**

```
get_validity(load_sample(), valid = TRUE)
```

---

load_population	<i>Load previously merged GBIF and NCBI taxonomies</i>
-----------------	--

---

**Description**

Load previously merged GBIF and NCBI taxonomies

**Usage**

```
load_population(x)
```

**Arguments**

`x` Path to a text file containing previously merged NCBI and GBIF taxonomies (compressed or uncompressed).

**Details**

This method imports a previously merged taxonomy from your file system. An example of a previously merged taxonomy can be downloaded from [https://drive.google.com/file/d/1gpvm9QKd0cuGo\\_cIXPkAgG1B-qfKZZU6/view?usp=sharing](https://drive.google.com/file/d/1gpvm9QKd0cuGo_cIXPkAgG1B-qfKZZU6/view?usp=sharing).

**Value**

A tibble containing merged GBIF and NCBI taxonomic data.

**Examples**

```
## Not run: load_population("path/to/merged_taxonomies")
```

---

load_sample	<i>Load an example of previously merged GBIF and NCBI taxonomies</i>
-------------	--

---

**Description**

Load an example of previously merged GBIF and NCBI taxonomies

**Usage**

```
load_sample()
```

**Details**

This method returns a small subset of previously merged GBIF and NCBI taxonomies. The subset is an example dataset that is only meant to be used to familiarize yourself with `taxonbridge` methods.

**Value**

A tibble containing a sample of merged GBIF and NCBI taxonomic data.

**Examples**

```
load_sample()
```

---

load_taxonomies	<i>Load and merge GBIF and NCBI taxonomic data</i>
-----------------	--

---

**Description**

Load and merge GBIF and NCBI taxonomic data

**Usage**

```
load_taxonomies(GBIF_path, NCBI_path)
```

**Arguments**

GBIF_path	Path to the GBIF backbone taxonomy (compressed or uncompressed).
NCBI_path	Path to the NCBI taxonomy (compressed or uncompressed).

**Details**

This method merges a GBIF `Taxon.tsv` file (see `download_gbif()`) and a Taxonkit (<https://bioinf.shenwei.me/taxonkit/download/>) output file (see `download_ncbi()`).

**Value**

A tibble containing merged GBIF and NCBI taxonomic data.

**Examples**

```
## Not run: load_taxonomies("path/to/GBIF/Taxon.tsv", "path/to/NCBI-Taxonkit/All.lineages.tsv.gz")
## Not run: load_taxonomies(download_gbif(), download_ncbi(taxonkitpath = "/path/to/taxonkit"))
```

---

plot\_mdb

*Generic for plot\_mdb methods*

---

**Description**

Generic for plot\_mdb methods

**Usage**

```
plot_mdb(x)
```

**Arguments**

x An object of the class one\_rank or the class all\_ranks.

**Details**

A generic with methods that plot taxonbridge data types (one\_rank and all\_ranks). These data types are created by using the methods prepare\_rank\_dist() or prepare\_comparable\_rank\_dist().

**Value**

A ggplot2 derived plot

**Examples**

```
plot_mdb(prepare_rank_dist(load_sample(), NCBI = TRUE, GBIF = TRUE))
plot_mdb(prepare_comparable_rank_dist(load_sample()))
plot_mdb(prepare_rank_dist(get_status(load_sample(), status = "synonym"), NCBI = TRUE))
plot_mdb(prepare_comparable_rank_dist(get_validity(get_status(load_sample()), valid = TRUE)))
```

---

prepare\_comparable\_rank\_dist  
*Get comparable NCBI and GBIF taxonomic ranks*

---

**Description**

Get comparable NCBI and GBIF taxonomic ranks

**Usage**

```
prepare_comparable_rank_dist(x, GBIF = TRUE, NCBI = TRUE)
```

**Arguments**

x	A tibble created with load_taxonomies() or load_population() or load_sample().
GBIF	A logical indicating whether GBIF taxonomic ranks are to be retrieved.
NCBI	A logical indicating whether NCBI taxonomic ranks are to be retrieved.

**Details**

This method, like prepare\_rank\_dist(), returns taxonomic ranks aggregated by frequency for data derived from the NCBI, the GBIF, or both. However, this method only retains taxonomic ranks that have at least one NCBI and one GBIF representative.

**Value**

A list of tibble(s) assigned to the S3 class one\_rank or to the S3 class all\_ranks.

**Examples**

```
prepare_comparable_rank_dist(load_sample())  
prepare_comparable_rank_dist(get_status(load_sample()), "accepted", NCBI = FALSE)
```

---

prepare\_rank\_dist *Get all NCBI and GBIF taxonomic ranks*

---

**Description**

Get all NCBI and GBIF taxonomic ranks

**Usage**

```
prepare_rank_dist(x, GBIF = FALSE, NCBI = FALSE)
```

**Arguments**

x	A tibble created with <code>load_taxonomies()</code> or <code>load_population()</code> or <code>load_sample()</code> .
GBIF	A logical indicating whether GBIF taxonomic ranks are to be retrieved.
NCBI	A logical indicating whether NCBI taxonomic ranks are to be retrieved.

**Details**

This method returns taxonomic ranks aggregated by frequency for data derived from the NCBI, the GBIF, or both.

**Value**

A list of tibble(s) assigned to the S3 class `one_rank` or to the S3 class `all_ranks`.

**Examples**

```
prepare_rank_dist(load_sample(), NCBI=TRUE, GBIF=TRUE)
prepare_rank_dist(load_sample(), NCBI=TRUE)
```

---

term\_conversion

*Convert GBIF terms to NCBI terms*

---

**Description**

Convert GBIF terms to NCBI terms

**Usage**

```
term_conversion(x)
```

**Arguments**

x	A tibble created with <code>load_taxonomies()</code> or <code>load_population()</code> or <code>load_sample()</code> .
---	--

**Details**

This method converts GBIF terminology to NCBI terminology where there is no biological provenance for the difference. Specifically, "Animalia" is converted to "Metazoa", and "Plantae" is converted to "Viridiplantae".

**Value**

A tibble with converted terms. The tibble is furthermore annotated with the attribute `converted=TRUE`.

**Examples**

```
term_conversion(load_sample())
```

# Index

[annotate](#), [2](#)

[dedupe](#), [3](#)

[download\\_gbif](#), [4](#)

[download\\_ncbi](#), [4](#)

[fuzzy\\_search](#), [5](#)

[get\\_inconsistencies](#), [6](#)

[get\\_lineages](#), [7](#)

[get\\_status](#), [8](#)

[get\\_taxa](#), [8](#)

[get\\_validity](#), [9](#)

[load\\_population](#), [10](#)

[load\\_sample](#), [11](#)

[load\\_taxonomies](#), [11](#)

[plot\\_mdb](#), [12](#)

[prepare\\_comparable\\_rank\\_dist](#), [13](#)

[prepare\\_rank\\_dist](#), [13](#)

[term\\_conversion](#), [14](#)