Package ‘ritis’

Title Integrated Taxonomic Information System Client


Version 1.0.0

License MIT + file LICENSE

URL https://github.com/ropensci/ritis (devel)
https://docs.ropensci.org/ritis/ (docs)

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

LazyData true

Encoding UTF-8

Language en-US

Imports solrium (>= 1.1.4), crul (>= 0.9.0), jsonlite, data.table, tibble

Suggests testthat, webmockr, vcr (>= 0.5.4)

RoxygenNote 7.1.1

X-schema.org-applicationCategory Taxonomy

X-schema.org-keywords taxonomy, biology, nomenclature, JSON, API, web, api-client, identifiers, species, names

X-schema.org-isPartOf https://ropensci.org

NeedsCompilation no

Author Scott Chamberlain [aut, cre] (<https://orcid.org/0000-0003-1444-9135>), rOpenSci [fnd] (https://ropensci.org/)

Maintainer Scott Chamberlain <myrmecocystus@gmail.com>

Repository CRAN

Date/Publication 2021-02-02 07:30:02 UTC
R topics documented:

ritis-package ............................................. 3
accepted_names .......................................... 3
any_match_count ........................................... 4
comment_detail ........................................... 5
common_names ............................................. 6
core_metadata ............................................. 6
coverage ..................................................... 7
credibility .................................................. 8
currency ...................................................... 9
date_data .................................................... 9
description ............................................... 10
experts ....................................................... 11
full_record ............................................... 11
geographic_divisions ..................................... 12
geographic_values ........................................ 13
global_species_completeness ......................... 13
hierarchy ................................................... 14
itis_facet ................................................... 15
itis_group ................................................... 16
itis_highlight ............................................. 16
itis_search ............................................... 17
jurisdiction ............................................... 18
kingdoms .................................................... 19
last_change_date ......................................... 20
lsid2tsn .................................................... 21
other_sources ............................................. 22
parent_tsn ................................................... 22
publications ............................................... 23
rank_name .................................................... 24
rank_names .................................................. 25
record ....................................................... 25
review_year ............................................... 26
scientific_name .......................................... 27
search_anymatch ......................................... 28
search_any_matchpaged .................................... 29
search_common ............................................ 30
search_scientific ......................................... 31
solr .......................................................... 32
solr_fields ................................................. 32
synonym_names .......................................... 33
taxon_authorship ........................................ 33
terms ........................................................ 34
tsn2lsid ..................................................... 35
tsn_by_vernacular_language ......................... 36
unacceptability_reason .................................. 36
usage ........................................................ 37
Description

Interface to Integrated Taxonomic Information (ITIS)

ritis package API

All functions that start with itis_ work with the ITIS Solr API described at https://www.itis.gov/solr_documentation.html, which uses the package solrium, and these functions have you use the solrium function interfaces, so you can pass on parameters to the solrium functions - so the solrium docs are important here.

All other functions work with the ITIS REST API described at https://www.itis.gov/ws_description.html. For these methods, they can grab data in either JSON or XML format. JSON is the default. We parse the JSON to R native format, either data.frame, character string, or list. You can get raw JSON as a character string back, or raw XML as a character string, and then parse yourself with jsonlite or xml2

You’ll also be interested in the taxize book https://taxize.dev/

Terminology

- "monomial": a taxonomic name with one part, e.g, Poa
- "binomial": a taxonomic name with two parts, e.g, Poa annua
- "trinomial": a taxonomic name with three parts, e.g, Poa annua annua

Author(s)

Scott Chamberlain <myrmecocystus@gmail.com>

accepted_names Get accepted names from tsn

Description

Get accepted names from tsn

Usage

accepted_names(tsn, wt = "json", raw = FALSE, ...)
any_match_count

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsn</td>
<td>TSN for a taxonomic group (numeric). Required.</td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;. Required.</td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string. Required. Default: FALSE</td>
</tr>
<tr>
<td>...</td>
<td>curl options passed on to <code>crl::HttpClient</code></td>
</tr>
</tbody>
</table>

Value

Zero row data.frame if the name is accepted, otherwise a data.frame with information on the currently accepted name.

Examples

```r
## Not run:
# TSN accepted - good name, empty data.frame returned
tsn = 208527
accepted_names(tsn)

# TSN not accepted - input TSN is old name, non-empty data.frame returned
tsn = 504239
accepted_names(tsn)

# raw json
tsn = 208527
accepted_names(tsn, raw = TRUE)
```

## End(Not run)

any_match_count

Get any match count.

Description

Get any match count.

Usage

```r
any_match_count(x, wt = "json", raw = FALSE, ...)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>text or taxonomic serial number (TSN) (character or numeric)</td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;. Required.</td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string. Required. Default: FALSE</td>
</tr>
<tr>
<td>...</td>
<td>curl options passed on to <code>crl::HttpClient</code></td>
</tr>
</tbody>
</table>
Value
An integer containing the number of matches the search will return.

Examples
```r
## Not run:
any_match_count(x = 202385)
any_match_count(x = "dolphin")
any_match_count(x = "dolphin", wt = "xml")

## End(Not run)
```

---

**comment_detail**  
*Get comment detail from TSN*

Description
Get comment detail from TSN

Usage
```r
comment_detail(tsn, wt = "json", raw = FALSE, ...)
```

Arguments
- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt** *(character)*: One of "json" or "xml". Required.
- **raw** *(logical)*: Return raw JSON or XML as character string. Required. Default: FALSE
- ... *curl options passed on to* `crl::HttpClient`

Value
A data.frame with results.

Examples
```r
## Not run:
comment_detail(tsn=180543)
comment_detail(tsn=180543, wt = "xml")

## End(Not run)
```
**common_names**  
*Get common names from tsn*

**Description**
Get common names from tsn

**Usage**
```r
common_names(tsn, wt = "json", raw = FALSE, ...)
```

**Arguments**
- `tsn` (numeric) TSN for a taxonomic group. Required.
- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `crl::HttpClient`

**Value**
a data.frame

**Examples**
```r
# Not run:
common_names(tsn=183833)
common_names(tsn=183833, wt = "xml")

# End(Not run)
```

---

**core_metadata**  
*Get core metadata from tsn*

**Description**
Get core metadata from tsn

**Usage**
```r
core_metadata(tsn, wt = "json", raw = FALSE, ...)
```
coverage

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
  ... curl options passed on to `curl::HttpClient`

Examples

```r
## Not run:
# coverage and currency data
core_metadata(tsn=28727)
core_metadata(tsn=28727, wt = "xml")
# no coverage or currency data
core_metadata(183671)
core_metadata(183671, wt = "xml")
## End(Not run)
```

Description

Get coverage from tsn

Usage

```r
coverage(tsn, wt = "json", raw = FALSE, ...)
```

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
  ... curl options passed on to `curl::HttpClient`

Examples

```r
## Not run:
# coverage data
coverage(tsn=28727)
# no coverage data
coverage(526852)
coverage(526852, wt = "xml")
## End(Not run)
```
credibility  

Get credibility rating from tsn

Description
Get credibility rating from tsn

Usage
credibility_rating(tsn, wt = "json", raw = FALSE, ...)

credibility_ratings(wt = "json", raw = FALSE, ...)

Arguments
- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to curl::HttpClient

Details
methods:
- credibility_rating: Get credibility rating for a tsn
- credibility_ratings: Get possible credibility ratings

Value
a data.frame

Examples
```r
## Not run:
credibility_rating(tsn = 526852)
credibility_rating(526852, wt = "xml")
credibility_rating(526852, raw = TRUE)

credibility_ratings()
credibility_ratings(wt = "xml")
credibility_ratings(raw = TRUE)
## End(Not run)
```
currency

Get currency from tsn

Description

Get currency from tsn

Usage

currency(tsn, wt = "json", raw = FALSE, ...)

Arguments

tsn  TSN for a taxonomic group (numeric). Required.
wt   (character) One of "json" or "xml". Required.
raw  (logical) Return raw JSON or XML as character string. Required. Default: FALSE

...  curl options passed on to curl::HttpClient

Value

a data.frame

Examples

## Not run:
# currency data
currency(tsn=28727)
currency(tsn=28727, wt = "xml")
# no currency dat
currency(526852)
currency(526852, raw = TRUE)

## End(Not run)

date_data

Get date data from tsn

Description

Get date data from tsn

Usage

date_data(tsn, wt = "json", raw = FALSE, ...)

...
Arguments

- **tsn** (numeric): TSN for a taxonomic group. Required.
- **wt** (character): One of "json" or "xml". Required.
- **raw** (logical): Return raw JSON or XML as character string. Required. Default: FALSE.
  - `...`: curl options passed on to `crl::HttpClient`.

Examples

```r
## Not run:
date_data(tsn = 180543)
date_data(180543, wt = "xml")
date_data(180543, wt = "json", raw = TRUE)
## End(Not run)
```

Description

Get description of the ITIS service

Usage

```r
description(wt = "json", raw = FALSE, ...)
```

Arguments

- **wt** (character): One of "json" or "xml". Required.
- **raw** (logical): Return raw JSON or XML as character string. Required. Default: FALSE.
  - `...`: curl options passed on to `crl::HttpClient`.

Value

A string, the ITIS web service description

Examples

```r
## Not run:
description()
description(wt = "xml")
## End(Not run)
```
experts  

Get expert information for the TSN.

**Description**

Get expert information for the TSN.

**Usage**

```r
experts(tsn, wt = "json", raw = FALSE, ...)
```

**Arguments**

- `tsn`  
  TSN for a taxonomic group (numeric). Required.
- `wt`  
  (character) One of "json" or "xml". Required.
- `raw`  
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...`  
  curl options passed on to `cru::HttpClient`

**Examples**

```r
## Not run:
exerts(tsn = 180544)
exerts(180544, wt = "xml")
exerts(180544, raw = TRUE)
## End(Not run)
```

full_record  

Get full record from TSN or lsid

**Description**

Get full record from TSN or lsid

**Usage**

```r
full_record(tsn = NULL, lsid = NULL, wt = "json", raw = FALSE, ...)
```

**Arguments**

- `tsn`  
  TSN for a taxonomic group (numeric). Required.
- `lsid`  
  lsid for a taxonomic group (character)
- `wt`  
  (character) One of "json" or "xml". Required.
- `raw`  
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...`  
  curl options passed on to `cru::HttpClient`
geographic_divisions

Examples

## Not run:
# from tsn
full_record(tsn = 50423)
full_record(tsn = 202385)
full_record(tsn = 183833)

full_record(tsn = 183833, wt = "xml")
full_record(tsn = 183833, raw = TRUE)

# from lsid
full_record(lsid = "urn:lsid:itis.gov:itis_tsn:180543")
full_record(lsid = "urn:lsid:itis.gov:itis_tsn:180543")

## End(Not run)

geographic_divisions  Get geographic divisions from tsn

Description

Get geographic divisions from tsn

Usage

geographic_divisions(tsn, wt = "json", raw = FALSE, ...)

Arguments

tsn
  TSN for a taxonomic group (numeric). Required.

wt
  (character) One of "json" or "xml". Required.

raw
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE

... curl options passed on to curl::HttpClient

Examples

## Not run:
geographic_divisions(tsn = 180543)

geographic_divisions(tsn = 180543, wt = "xml")

geographic_divisions(tsn = 180543, wt = "json", raw = TRUE)

## End(Not run)
**geographic_values**  
*Get all possible geographic values*

### Description
Get all possible geographic values

### Usage

```r
geographic_values(wt = "json", raw = FALSE, ...)
```

### Arguments

- `wt` *(character)* One of "json" or "xml". Required.
- `raw` *(logical)* Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `cru::HttpClient`

### Value
character vector of geographic names

### Examples
```r
## Not run:
geographic_values()
geographic_values(wt = "xml")
geographic_values(wt = "json", raw = TRUE)
## End(Not run)
```

---

**global_species_completeness**  
*Get global species completeness from tsn*

### Description
Get global species completeness from tsn

### Usage

```r
global_species_completeness(tsn, wt = "json", raw = FALSE, ...)
```
hierarchy

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to **crl::HttpClient**

Examples

```r
## Not run:
global_species_completeness(tsn = 180541)
global_species_completeness(180541, wt = "xml")
global_species_completeness(180541, wt = "json", raw = TRUE)
## End(Not run)
```

hierarchy

*Get hierarchy down from tsn*

Description

Get hierarchy down from tsn

Usage

- `hierarchy_down(tsn, wt = "json", raw = FALSE, ...)`
- `hierarchy_up(tsn, wt = "json", raw = FALSE, ...)`
- `hierarchy_full(tsn, wt = "json", raw = FALSE, ...)`

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to **crl::HttpClient**

Details

Hierarchy methods:

- `hierarchy_down`: Get hierarchy down from tsn
- `hierarchy_up`: Get hierarchy up from tsn
- `hierarchy_full`: Get full hierarchy from tsn
Examples

```r
## Not run:
## Full down (class Mammalia)
hierarchy_down(tsn=179913)

## Full up (genus Agoseris)
hierarchy_up(tsn=36485)

## Full hierarchy
### genus Liatris
hierarchy_full(tsn=37906)
### get raw data back
hierarchy_full(tsn=37906, raw = TRUE)
### genus Baetis, get xml back
hierarchy_full(100800, wt = "xml")
```

## End(Not run)

### itis_facet

#### ITIS Solr facet

Description

ITIS Solr facet

Usage

```r
itis_facet(..., proxy = NULL, callopts = list())
```

Arguments

- `...`: Arguments passed on to the `params` parameter of the `solr::solr_facet()` function. See `solr_fields` for possible parameters, and examples below
- `proxy`: List of arguments for a proxy connection, including one or more of: `url`, `port`, `username`, `password`, and `auth`. See `crul::proxy()` for help, which is used to construct the proxy connection.
- `callopts`: Curl options passed on to `crul::HttpClient`

Examples

```r
## Not run:
itis_facet(q = "rank:Species", rows = 0, facet.field = "kingdom")$facet_fields

x <- itis_facet(q = "hierarchySoFar:*$Aves$* AND rank:Species AND usage:valid",
    facet.pivot = "nameWInd,vernacular", facet.limit = -1, facet.mincount = 1,
    rows = 0)
head(x$facet_pivot$`nameWInd,vernacular``)
```

## End(Not run)
### itis_group

**ITIS Solr group search**

**Description**

ITIS Solr group search

**Usage**

`itis_group(..., proxy = NULL, callopts = list())`

**Arguments**

- `...`: Arguments passed on to the `params` parameter of the `solrium::solr_group()` function. See `solr_fields` for possible parameters, and examples below
- `proxy`: List of arguments for a proxy connection, including one or more of: url, port, username, password, and auth. See `curl::proxy()` for help, which is used to construct the proxy connection.
- `callopts`: Curl options passed on to `curl::HttpClient`

**Examples**

```r
## Not run:
x <- itis_group(q = "nameWOInd://[A-Za-z0-9]*[\%20]{0,0}*/",
  group.field = 'rank', group.limit = 3)
head(x)
## End(Not run)
```

### itis_highlight

**ITIS Solr highlight**

**Description**

ITIS Solr highlight

**Usage**

`itis_highlight(..., proxy = NULL, callopts = list())`

**Arguments**

- `...`: Arguments passed on to the `params` parameter of the `solrium::solr_highlight()` function. See `solr_fields` for possible parameters, and examples below
- `proxy`: List of arguments for a proxy connection, including one or more of: url, port, username, password, and auth. See `curl::proxy()` for help, which is used to construct the proxy connection.
- `callopts`: Curl options passed on to `curl::HttpClient`
Examples

## Not run:
itis_highlight(q = "rank:Species", hl.fl = 'rank', rows=10)

## End(Not run)

The syntax for this function can be a bit hard to grasp. See https://itis.gov/solr_examples.html for help on generating the syntax ITIS wants for specific searches.

References

https://www.itis.gov/solr_documentation.html

Examples

## Not run:
itis_search(q = "tsn:182662")

# get all orders within class Aves (birds)
z <- itis_search(q = "rank:Class AND nameWOInd:Aves")
hierarchy_down(z$tsn)

# get taxa "downstream" from a target taxon
## taxize and taxizedb packages have downstream() fxns, but
## you can do a similar thing here by iteratively drilling down
## the taxonomic hierarchy
# here, we get families within Aves
library(data.table)
aves <- itis_search(q = "rank:Class AND nameWOInd:Aves")
aves_orders <- hierarchy_down(aves$tsn)
aves_families <- lapply(aves_orders$tsn, hierarchy_down)
rbindlist(aves_families)

# the tila operator
itis_search(q = "nameWOInd:Liquidamber\ styraciflua-0.4")

# matches only monomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ \]{0,0}*/")

# matches only binomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ \]{1,1}[A-Za-z0-9]*/*/")

# matches only trinomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ \]{1,1}[A-Za-z0-9]*[ \]{1,1}[A-Za-z0-9]*/*/")

# matches binomials or trinomials
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ \]{1,1}[A-Za-z0-9]*[ \]{0,1}[A-Za-z0-9]*/*/")

itis_search(q = "nameWOInd:Poa\ annua")

# pagination
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ \]{0,0}*/", rows = 2)
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ \]{0,0}*/", rows = 200)

# select fields to return
itis_search(q = "nameWOInd:/[A-Za-z0-9]*[ \]{0,0}*/",
            fl = c('nameWInd', 'tsn'))

## End(Not run)

## Get jurisdictional origin from tsn

### Description
Get jurisdictional origin from tsn

### Usage
jurisdictional_origin(tsn, wt = "json", raw = FALSE, ...)
jurisdiction_origin_values(wt = "json", raw = FALSE, ...)
jurisdiction_values(wt = "json", raw = FALSE, ...)
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsn</td>
<td>TSN for a taxonomic group (numeric).</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;.</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string.</td>
<td>Yes</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

... curl options passed on to `crl::HttpClient`

Details

Jurisdiction methods:

- **jurisdictional_origin**: Get jurisdictional origin from tsn
- **jurisdiction_origin_values**: Get jurisdiction origin values
- **jurisdiction_values**: Get all possible jurisdiction values

Value

- **jurisdictional_origin**: data.frame
- **jurisdiction_origin_values**: data.frame
- **jurisdiction_values**: character vector

Examples

```r
# Not run:
jurisdictional_origin(tsn=180543)
jurisdictional_origin(tsn=180543, wt = "xml")
jurisdiction_origin_values()
jurisdiction_values()
```

# End(Not run)

---

**kingdoms**

*Get kingdom names from tsn*

Description

Get kingdom names from tsn

Usage

```r
kingdom_name(tsn, wt = "json", raw = FALSE, ...)
kingdom_names(wt = "json", raw = FALSE, ...)
```
Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `crl::HttpClient`

Details

- **kingdom_name**: Get kingdom name for a TSN
- **kingdom_names**: Get all possible kingdom names

Examples

```r
## Not run:
kingdom_name(202385)
kingdom_name(202385, wt = "xml")
kingdom_names()

## End(Not run)
```

---

### last_change_date

**Provides the date the ITIS database was last updated**

Description

Provides the date the ITIS database was last updated

Usage

```r
last_change_date(wt = "json", raw = FALSE, ...)
```

Arguments

- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `crl::HttpClient`

Value

character value with a date
**lsid2tsn**

**Examples**

```r
## Not run:
last_change_date()
last_change_date(wt = "xml")

## End(Not run)
```

---

**lsid2tsn**  
*Gets the TSN corresponding to the LSID, or an empty result if there is no match.*

**Description**

Gets the TSN corresponding to the LSID, or an empty result if there is no match.

**Usage**

```r
lsid2tsn(lsid, wt = "json", raw = FALSE, ...)
```

**Arguments**

- `lsid`  
  (character) Lsid for a taxonomic group. Required.

- `wt`  
  (character) One of "json" or "xml". Required.

- `raw`  
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE

- `...`  
  curl options passed on to `crl::HttpClient`

**Examples**

```r
## Not run:
lsid2tsn(lsid="urn:lsid:itis.gov:itis_tsn:28726")
lsid2tsn(lsid="urn:lsid:itis.gov:itis_tsn:28726", wt = "xml")
lsid2tsn("urn:lsid:itis.gov:itis_tsn:0")
lsid2tsn("urn:lsid:itis.gov:itis_tsn:0", wt = "xml")

## End(Not run)
```
other_sources

Returns a list of the other sources used for the TSN.

Description

Returns a list of the other sources used for the TSN.

Usage

other_sources(tsn, wt = "json", raw = FALSE, ...)

Arguments

tsn        TSN for a taxonomic group (numeric). Required.
wt         (character) One of "json" or "xml". Required.
raw        (logical) Return raw JSON or XML as character string. Required. Default: FALSE
...        curl options passed on to crul::HttpClient

Examples

## Not run:
# results
other_sources(tsn=182662)
# no results
other_sources(tsn=2085272)
# get xml
other_sources(tsn=182662, wt = "xml")

## End(Not run)

parent_tsn

Returns the parent TSN for the entered TSN.

Description

Returns the parent TSN for the entered TSN.

Usage

parent_tsn(tsn, wt = "json", raw = FALSE, ...)

publications

Arguments

  tsn  TSN for a taxonomic group (numeric). Required.
  wt   (character) One of "json" or "xml". Required.
  raw  (logical) Return raw JSON or XML as character string. Required. Default: FALSE

...  curl options passed on to `crul::HttpClient`

Value

  a data.frame

Examples

  ## Not run:
  parent_tsn(tsn = 202385)
  parent_tsn(tsn = 202385, raw = TRUE)
  parent_tsn(tsn = 202385, wt = "xml")

  ## End(Not run)

publications  Returns a list of the publications used for the TSN.

Description

  Returns a list of the publications used for the TSN.

Usage

  publications(tsn, wt = "json", raw = FALSE, ...)

Arguments

  tsn  TSN for a taxonomic group (numeric). Required.
  wt   (character) One of "json" or "xml". Required.
  raw  (logical) Return raw JSON or XML as character string. Required. Default: FALSE

...  curl options passed on to `crul::HttpClient`

Value

  a data.frame
rank_name

Examples

```r
## Not run:
publications(tsn = 70340)
publications(tsn = 70340, wt = "xml")

publications(tsn = 70340, verbose = TRUE)

## End(Not run)
```

---

**rank_name**

*Returns the kingdom and rank information for the TSN.*

---

### Description

Returns the kingdom and rank information for the TSN.

### Usage

```r
rank_name(tsn, wt = "json", raw = FALSE, ...)
```

### Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- **...**: curl options passed on to `crl::HttpClient`

### Value

A data.frame, with rank name and other info

### Examples

```r
## Not run:
rank_name(tsn = 202385)

## End(Not run)
```
**rank_names**

Provides a list of all the unique rank names contained in the database and their kingdom and rank ID values.

**Description**

Provides a list of all the unique rank names contained in the database and their kingdom and rank ID values.

**Usage**

```r
desc(rank_names(wt = "json", raw = FALSE, ...))
```

**Arguments**

- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `cru::HttpClient`

**Value**

A data.frame, with columns:
- kingdomname
- rankid
- rankname

**Examples**

```r
### Not run:
desc(rank_names())
### End(Not run)
```

---

**record**

*Gets a record from an LSID*

**Description**

Gets a record from an LSID

**Usage**

```r
desc(record(lsid, wt = "json", raw = FALSE, ...))
```
review_year

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lsid</td>
<td>lsid for a taxonomic group (character). Required.</td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;. Required.</td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string. Required. Default: FALSE</td>
</tr>
</tbody>
</table>

Details

Gets the partial ITIS record for the TSN in the LSID, found by comparing the TSN in the search key to the TSN field. Returns an empty result set if there is no match or the TSN is invalid.

Value

a data.frame

Examples

```r
## Not run:
record(lsid = "urn:lsid:itis.gov:itis_tsn:180543")
## End(Not run)
```

Description

Returns the review year for the TSN.

Usage

```r
review_year(tsn, wt = "json", raw = FALSE, ...)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsn</td>
<td>TSN for a taxonomic group (numeric). Required.</td>
</tr>
<tr>
<td>wt</td>
<td>(character) One of &quot;json&quot; or &quot;xml&quot;. Required.</td>
</tr>
<tr>
<td>raw</td>
<td>(logical) Return raw JSON or XML as character string. Required. Default: FALSE</td>
</tr>
</tbody>
</table>

Value

a data.frame
Examples

```r
## Not run:
review_year(tsn = 180541)

## End(Not run)
```

---

**scientific_name**

*Returns the scientific name for the TSN. Also returns the component parts (names and indicators) of the scientific name.*

Description

Returns the scientific name for the TSN. Also returns the component parts (names and indicators) of the scientific name.

Usage

```r
scientific_name(tsn, wt = "json", raw = FALSE, ...)
```

Arguments

- `tsn`: TSN for a taxonomic group (numeric). Required.
- `wt`: (character) One of "json" or "xml". Required.
- `raw`: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...`: curl options passed on to `crl::HttpClient`

Value

a data.frame

Examples

```r
## Not run:
scientific_name(tsn = 531894)

## End(Not run)
```
search_anymatch  

Search for any match

Description

Search for any match

Usage

search_anymatch(x, wt = "json", raw = FALSE, ...)

Arguments

x         text or taxonomic serial number (TSN) (character or numeric)
wt        (character) One of "json" or "xml". Required.
raw       (logical) Return raw JSON or XML as character string. Required. Default: FALSE
...       curl options passed on to crul::HttpClient

Value

a data.frame

See Also

search_any_match_paged

Examples

## Not run:
search_anymatch(x = 202385)
search_anymatch(x = "dolphin")
# no results
search_anymatch(x = "Pisces")

## End(Not run)
**search_any_match_paged**

*Search for any matched page*

**Description**

Search for any matched page

**Usage**

```r
search_any_match_paged(
  x,
  pagesize = NULL,
  pagenum = NULL,
  ascend = NULL,
  wt = "json",
  raw = FALSE,
  ...)
```

**Arguments**

- **x**: text or taxonomic serial number (TSN) (character or numeric)
- **pagesize**: An integer containing the page size (numeric)
- **pagenum**: An integer containing the page number (numeric)
- **ascend**: A boolean containing true for ascending sort order or false for descending (logical)
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- **...**: curl options passed on to `crl::HttpClient`

**Value**

- a data.frame
- a data.frame

**See Also**

- `search_anymatch`
Examples

```r
## Not run:
search_any_match_paged(x=202385, pagesize=100, pagenum=1, ascend=FALSE)
search_any_match_paged(x="Zy", pagesize=100, pagenum=1, ascend=FALSE)

## End(Not run)
```

---

search_common  

Search for tsn by common name

Description

Search for tsn by common name

Usage

```r
search_common(x, from = "all", wt = "json", raw = FALSE, ...)
```

Arguments

- `x`  
  text or taxonomic serial number (TSN) (character or numeric)
- `from`  
  (character) One of "all", "begin", or "end". See Details.
- `wt`  
  (character) One of "json" or "xml". Required.
- `raw`  
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...`  
  curl options passed on to `curl::HttpClient`

Details

The `from` parameter:

- `all` - Search against the `searchByCommonName` API route, which searches entire name string
- `begin` - Search against the `searchByCommonNameBeginsWith` API route, which searches for a match at the beginning of a name string
- `end` - Search against the `searchByCommonNameEndsWith` API route, which searches for a match at the end of a name string

Value

a data.frame

See Also

`search_scientific()`
**search_scientific**  

Search by scientific name

**Description**  
Search by scientific name

**Usage**  

```
search_scientific(x, wt = "json", raw = FALSE, ...)  
```

**Arguments**

- **x**: text or taxonomic serial number (TSN) (character or numeric)
- **wt** (character) One of "json" or "xml". Required.
- **raw** (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- **...**: curl options passed on to crul::HttpClient

**Value**  

a data.frame

**See Also**  

search_common

---

**Examples**

```r
## Not run:
search_common("american bullfrog")
search_common("ferret-badger")
search_common("polar bear")

# comparison: all, begin, end
search_common("inch")
search_common("inch", from = "begin")
search_common("inch", from = "end")

# end
search_common("snake", from = "end")

## End(Not run)
```
Examples

```r
## Not run:
solr_search_scientific("Tardigrada")
solr_search_scientific("Quercus douglasii")

## End(Not run)
```

Description

ITIS provides access to their data via their Solr service described at [https://www.itis.gov/solr_documentation.html](https://www.itis.gov/solr_documentation.html). This is a powerful interface to ITIS data as you have access to a very flexible query interface.

Details

See `solr_fields` and [https://www.itis.gov/solr_documentation.html](https://www.itis.gov/solr_documentation.html) for guidance on available fields.

Functions

- `itis_search()` - Search
- `itis_group()` - Group
- `itis_highlight()` - Highlight
- `itis_facet()` - Facet

solr_fields

List of fields that can be used in `solr` functions

Description

Each element in the list has a list of length tree, with:

Format

A list of length 36

Details

- field: the field name, this is the name you can use in your queries
- definition: the definition of the field
- example: an example value

Source

[https://www.itis.gov/solr_documentation.html](https://www.itis.gov/solr_documentation.html)
synonym_names

**Description**

Returns a list of the synonyms (if any) for the TSN.

**Usage**

```r
synonym_names(tsn, wt = "json", raw = FALSE, ...)
```

**Arguments**

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- **...**: curl options passed on to `crl::HttpClient`

**Value**

a data.frame

**Examples**

```r
## Not run:
synonym_names(tsn=183671) # tsn not accepted
synonym_names(tsn=526852) # tsn accepted

## End(Not run)
```

taxon_authorship

**Description**

Returns the author information for the TSN.

**Usage**

```r
taxon_authorship(tsn, wt = "json", raw = FALSE, ...)
```
terms

Arguments

- **tsn**: TSN for a taxonomic group (numeric). Required.
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `crul::HttpClient`

Value

- a data.frame

Examples

```r
## Not run:
taxon_authorship(tsn = 183671)
## End(Not run)
```

Description

Get ITIS terms, i.e., tsn’s, authors, common names, and scientific names

Usage

terms(query, what = "both", wt = "json", raw = FALSE, ...)

Arguments

- **query**: One or more common or scientific names, or partial names
- **what**: One of both (search common and scientific names), common (search just common names), or scientific (search just scientific names)
- **wt**: (character) One of "json" or "xml". Required.
- **raw**: (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ... curl options passed on to `crul::HttpClient`
### tsn2lsid

**tsn2lsid**  
*Gets the unique LSID for the TSN, or an empty result if there is no match.*

**Description**

Gets the unique LSID for the TSN, or an empty result if there is no match.

**Usage**

```r
tsn2lsid(tsn, wt = "json", raw = FALSE, ...)
```

**Arguments**

- `tsn`  
  TSN for a taxonomic group (numeric). Required.
- `wt`  
  (character) One of "json" or "xml". Required.
- `raw`  
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...`  
  curl options passed on to `cru::HttpClient`

**Value**

a character string, an LSID, or NULL if nothing found

**Examples**

```r
## Not run:
tsn2lsid(tsn = 155166)
tsn2lsid(tsn = 333333333)
tsn2lsid(155166, raw = TRUE)
tsn2lsid(155166, wt = "xml")
```

## End(Not run)
tsn_by_vernacular_language

*Get tsn by vernacular language*

**Description**

Get tsn by vernacular language

**Usage**

`tsn_by_vernacular_language(language, wt = "json", raw = FALSE, ...)`

**Arguments**

- `language` A string containing the language. This is a language string, not the international language code (character)
- `wt` (character) One of "json" or "xml". Required.
- `raw` (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- `...` curl options passed on to `crl::HttpClient`

**Value**

a data.frame

**Examples**

```r
## Not run:
tsn_by_vernacular_language(language = "french")
## End(Not run)
```

---

unacceptability_reason

*Returns the unacceptability reason, if any, for the TSN.*

**Description**

Returns the unacceptability reason, if any, for the TSN.

**Usage**

`unacceptability_reason(tsn, wt = "json", raw = FALSE, ...)`
**Arguments**

- **tsn**  
  TSN for a taxonomic group (numeric). Required.
- **wt**  
  (character) One of "json" or "xml". Required.
- **raw**  
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ...  
  curl options passed on to `cru::HttpClient`

**Examples**

```r
## Not run:
usage(tsn = 183671)
## End(Not run)
```

---

**usage**  
*Returns the usage information for the TSN.*

**Description**

Returns the usage information for the TSN.

**Usage**

```r
usage(tsn, wt = "json", raw = FALSE, ...)
```

**Arguments**

- **tsn**  
  TSN for a taxonomic group (numeric). Required.
- **wt**  
  (character) One of "json" or "xml". Required.
- **raw**  
  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
- ...  
  curl options passed on to `cru::HttpClient`

**Examples**

```r
## Not run:
usage(tsn = 526852)
usage(tsn = 526852, raw = TRUE)
usage(tsn = 526852, wt = "xml")
## End(Not run)
```
vernacular_languages  Provides a list of the unique languages used in the vernacular table.

Description

Provides a list of the unique languages used in the vernacular table.

Usage

vernacular_languages(wt = "json", raw = FALSE, ...)

Arguments

wt  (character) One of "json" or "xml". Required.
raw  (logical) Return raw JSON or XML as character string. Required. Default: FALSE
...  curl options passed on to crul::HttpClient

Value

a character vector of vernacular names

Examples

## Not run:
vernacular_languages()

## End(Not run)
# Index

<table>
<thead>
<tr>
<th><em>data</em></th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>solr_fields</td>
<td></td>
</tr>
<tr>
<td>accepted_names</td>
<td>3</td>
</tr>
<tr>
<td>any_match_count</td>
<td>4</td>
</tr>
<tr>
<td>comment_detail</td>
<td>5</td>
</tr>
<tr>
<td>common_names</td>
<td>6</td>
</tr>
<tr>
<td>core_metadata</td>
<td>6</td>
</tr>
<tr>
<td>coverage</td>
<td>7</td>
</tr>
<tr>
<td>credibility</td>
<td>8</td>
</tr>
<tr>
<td><strong>credibility_rating (credibility)</strong></td>
<td>8</td>
</tr>
<tr>
<td>credibility_ratings (credibility)</td>
<td>8</td>
</tr>
<tr>
<td>crul::HttpClient</td>
<td>4–17, 19–31, 33–38</td>
</tr>
<tr>
<td>crul::proxy()</td>
<td>15–17</td>
</tr>
<tr>
<td>currency</td>
<td>9</td>
</tr>
<tr>
<td>date_data</td>
<td>9</td>
</tr>
<tr>
<td>description</td>
<td>10</td>
</tr>
<tr>
<td>experts</td>
<td>11</td>
</tr>
<tr>
<td>full_record</td>
<td>11</td>
</tr>
<tr>
<td>geographic_divisions</td>
<td>12</td>
</tr>
<tr>
<td>geographic_values</td>
<td>13</td>
</tr>
<tr>
<td>global_species_completeness</td>
<td>13</td>
</tr>
<tr>
<td>hierarchy</td>
<td>14</td>
</tr>
<tr>
<td>hierarchy_down (hierarchy)</td>
<td>14</td>
</tr>
<tr>
<td>hierarchy_full (hierarchy)</td>
<td>14</td>
</tr>
<tr>
<td>hierarchy_up (hierarchy)</td>
<td>14</td>
</tr>
<tr>
<td>itis_facet</td>
<td>15</td>
</tr>
<tr>
<td>itis_facet()</td>
<td>32</td>
</tr>
<tr>
<td>itis_group</td>
<td>16</td>
</tr>
<tr>
<td>itis_group()</td>
<td>32</td>
</tr>
<tr>
<td>itis_highlight</td>
<td>16</td>
</tr>
<tr>
<td>itis_highlight()</td>
<td>32</td>
</tr>
<tr>
<td>itis_search</td>
<td>17</td>
</tr>
<tr>
<td>itis_search()</td>
<td>32</td>
</tr>
<tr>
<td>jurisdiction</td>
<td>18</td>
</tr>
<tr>
<td>jurisdiction_origin_values</td>
<td>18</td>
</tr>
<tr>
<td>(jurisdiction)</td>
<td></td>
</tr>
<tr>
<td>jurisdiction_values (jurisdiction)</td>
<td>18</td>
</tr>
<tr>
<td>jurisdictional_origin (jurisdiction)</td>
<td>18</td>
</tr>
<tr>
<td>kingdom_name</td>
<td>19</td>
</tr>
<tr>
<td>kingdom_names</td>
<td>19</td>
</tr>
<tr>
<td>kingdoms</td>
<td></td>
</tr>
<tr>
<td>last_change_date</td>
<td>20</td>
</tr>
<tr>
<td>lsid2tsn</td>
<td>21</td>
</tr>
<tr>
<td>other_sources</td>
<td>22</td>
</tr>
<tr>
<td>parent_tsn</td>
<td>22</td>
</tr>
<tr>
<td>publications</td>
<td>23</td>
</tr>
<tr>
<td>rank_name</td>
<td>24</td>
</tr>
<tr>
<td>rank_names</td>
<td>25</td>
</tr>
<tr>
<td>record</td>
<td>25</td>
</tr>
<tr>
<td>review_year</td>
<td>26</td>
</tr>
<tr>
<td>ritis (ritis-package)</td>
<td>3</td>
</tr>
<tr>
<td>ritis-package</td>
<td>3</td>
</tr>
<tr>
<td>scientific_name</td>
<td>27</td>
</tr>
<tr>
<td>search_anymatch</td>
<td>28, 29</td>
</tr>
<tr>
<td>search_anymatch_paged</td>
<td>28, 29</td>
</tr>
<tr>
<td>search_common</td>
<td>30, 31</td>
</tr>
<tr>
<td>search_scientific</td>
<td>31</td>
</tr>
<tr>
<td>search_scientific()</td>
<td>30</td>
</tr>
<tr>
<td>solr</td>
<td>32, 32</td>
</tr>
<tr>
<td>solr_fields</td>
<td>15–17, 32, 32</td>
</tr>
<tr>
<td>solrium::solr_facet()</td>
<td>15</td>
</tr>
<tr>
<td>solrium::solr_group()</td>
<td>16</td>
</tr>
<tr>
<td>solrium::solr_highlight()</td>
<td>16</td>
</tr>
<tr>
<td>solrium::solr_search()</td>
<td>17</td>
</tr>
<tr>
<td>synonym_names</td>
<td>33</td>
</tr>
<tr>
<td>taxon_authorship</td>
<td>33</td>
</tr>
<tr>
<td>terms</td>
<td>34</td>
</tr>
</tbody>
</table>
tsn2lsid, 35
tsn_by_vernacular_language, 36
unacceptability_reason, 36
usage, 37
vernacular_languages, 38