

# Package ‘ctrdata’

December 14, 2024

**Type** Package

**Title** Retrieve and Analyze Clinical Trials in Public Registers

**Version** 1.20.0

**Imports** jsonlite, httr, curl (>= 5.1.0), clipr, xml2, nodbi (>= 0.10.7), stringi, lubridate, jqr, dplyr, zip, V8, readr, digest, countrycode, htmlwidgets

**URL** <https://cran.r-project.org/package=ctrdata>,  
<https://rfhb.github.io/ctrdata/>

**BugReports** <https://github.com/rfhb/ctrdata/issues>

**Description** A system for querying, retrieving and analyzing protocol- and results-related information on clinical trials from four public registers, the 'European Union Clinical Trials Register' ('EUCTR', <<https://www.clinicaltrialsregister.eu/>>), 'ClinicalTrials.gov' (<<https://clinicaltrials.gov/>> and also translating queries the retired classic interface), the 'ISRCTN' (<<http://www.isrctn.com/>>) and the 'European Union Clinical Trials Information System' ('CTIS', <<https://euclinicaltrials.eu/>>). Trial information is downloaded, converted and stored in a database ('PostgreSQL', 'SQLite', 'DuckDB' or 'MongoDB'; via package 'nodbi'). Documents in registers associated with trials can also be downloaded. Other functions identify deduplicated records, easily find and extract variables (fields) of interest even from complex nested data as used by the registers, merge variables and update queries. The package can be used for meta-analysis and trend-analysis of the design and conduct as well as of the results of clinical trials.

**License** MIT + file LICENSE

**RoxygenNote** 7.3.2

**Suggests** devtools, knitr, rmarkdown, RSQLite, tibble, mongolite, tinytest (>= 1.2.1), R.rsp, RPostgres, duckdb

**VignetteBuilder** R.rsp

**NeedsCompilation** no

**Encoding** UTF-8

**Language** en-GB

**Author** Ralf Herold [aut, cre] (<<https://orcid.org/0000-0002-8148-6748>>)

**Maintainer** Ralf Herold <ralf.herold@mailbox.org>

**Repository** CRAN

**Date/Publication** 2024-12-14 10:00:02 UTC

## Contents

ctrdata . . . . .	2
ctrdata-registers . . . . .	3
ctrFindActiveSubstanceSynonyms . . . . .	5
ctrGetQueryUrl . . . . .	6
ctrLoadQueryIntoDb . . . . .	7
ctrOpenSearchPagesInBrowser . . . . .	10
ctrShowOneTrial . . . . .	11
dbFindFields . . . . .	13
dbFindIdsUniqueTrials . . . . .	14
dbGetFieldsIntoDf . . . . .	15
dbQueryHistory . . . . .	17
dfMergeVariablesRelevel . . . . .	17
dfName2Value . . . . .	18
dfTrials2Long . . . . .	20

<b>Index</b>	<b>22</b>
--------------	-----------

---

ctrdata	<i>ctrdata: get started, database connection, function overview</i>
---------	---

---

## Description

A package for aggregating and analysing information on clinical studies, and for obtaining documents, from public registers

### 1 - Database connection

Package `ctrdata` retrieves trial information and stores it in a database collection. Therefore, a database connection object has to be given to parameter `con` for several `ctrdata` functions. The connection object is built using `nodbi` which allows to use different database backends. Specifying a `collection = "<my collection's name>"` is necessary for package `ctrdata`. A connection object (here called `dbc`) is created in almost identical ways for these supported backends:

<i>Database</i>	<i>Connection object</i>
MongoDB	<code>dbc &lt;- nodbi::src_mongo(db = "my_db", collection = "my_coll")</code>

```
DuckDB      dbc <- nodbi::src_duckdb(dbname = "my_db", collection = "my_coll")
SQLite      dbc <- nodbi::src_sqlite(dbname = "my_db", collection = "my_coll")
PostgreSQL  dbc <- nodbi::src_postgres(dbname = "my_db"); dbc[["collection"]] <- "my_coll"
```

## 2 - Operate on a clinical trial register

[ctrOpenSearchPagesInBrowser](#), [ctrLoadQueryIntoDb](#) (load trial records into database collection); see [ctrdata-registers](#) for details on registers and how to search.

## 3 - Get a data frame from the database collection

[ctrShowOneTrial](#) (show widget to explore structure, fields and data of a trial), [dbFindFields](#) (find names of fields of interest in trial records in a collection), [dbGetFieldsIntoDf](#) (create a data frame with fields of interest from collection), [dbFindIdsUniqueTrials](#) (get de-duplicated identifiers of clinical trials' records to subset a data frame).

## 4 - Operate on a data frame with trial information

[dfTrials2Long](#) (convert fields with nested elements into long format), [dfName2Value](#) (get values for variable(s) of interest).

### Author(s)

Ralf Herold <[ralf.herold@mailbox.org](mailto:ralf.herold@mailbox.org)>

### See Also

Useful links:

- <https://cran.r-project.org/package=ctrdata>
- <https://rfhb.github.io/ctrdata/>
- Report bugs at <https://github.com/rfhb/ctrdata/issues>

### Description

Registers of clinical trials from which protocol- and result-related information can be retrieved and analysed with package [ctrdata](#), last updated 2024-12-06.

## 1 - Overview

- **EUCTR**: The EU Clinical Trials Register contains more than 44,200 clinical trials (at least one investigational medicinal product, IMP; in the European Union and beyond; no new trials, and results of trials continue to be added)
- **CTIS**: The EU Clinical Trials Information System started in January 2023 for new clinical trials. It includes more than 7,900 publicly accessible trials. How to automatically get the CTIS search query URL: [here](#)
- **CTGOV2**: ClinicalTrials.gov includes more than 518,000 interventional and observational studies
- **ISRCTN**: The ISRCTN Registry includes more than 25,700 interventional and observational health studies

## 2 - Notable changes

CTGOV was retired on 2024-06-25; ctrdata subsequently translates CTGOV queries to CTGOV2 queries. The new website (CTGOV2) can be used with ctrdata since 2023-08-27. CTIS was relaunched on 2024-06-17, changing the data structure and search syntax, to which ctrdata was updated. CTIS can be used with ctrdata since 2023-03-25. More information on changes: [here](#).

## 3 - References

Material	EUCTR	CTGOV2	ISRCTN	CTIS
Home page	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
About	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
Terms & conditions, disclaimer	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
How to search	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
Search interface	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
Expert / advanced search	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
Glossary	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	
FAQ, caveats, issues	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
Definitions of fields	<a href="#">link</a>	<a href="#">link</a> , <a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>
Example*	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>

## 4 - Example and ctrdata motivation

\*The example is an expert search for interventional trials primarily with neonates, investigating infectious conditions. It shows that searches in registers may not be sufficient to identify the sought trials:

- The CTGOV2 search retrieves trials conducted exclusively with neonates.
- EUCTR retrieves trials with neonates, but not only those exclusively in neonates.
- ISRCTN retrieves studies with interventions other than medicines.
- CTIS retrieves trials that mention the words neonates and infection. To show search results, see [here](#)

To address this, trials can be retrieved with `ctrLoadQueryIntoDb` into a database collection and in a second step can be selected, based on values of relevant fields of all retrieved trial information, for example:

- EUCTR field `f115_children_211years` for age criteria
- ISRCTN field `interventions.intervention.interventionType` for type of study
- CTIS fields `ageGroup` and `authorizedApplication.authorizedPartI.medicalConditions.medicalCondition`

`ctrdata` helps identifying fields with function `dbGetFieldsIntoDf`.

### Author(s)

Ralf Herold <ralf.herold@mailbox.org>

---

ctrFindActiveSubstanceSynonyms

*Find synonyms of an active substance*

---

### Description

An active substance can be identified by a recommended international nonproprietary name (INN), a trade or product name, or a company code(s). To find likely synonyms, the function retrieves from CTGOV2 the field `protocolSection.armsInterventionsModule.interventions.otherNames`. Note this is not free of error and should be checked manually.

### Usage

```
ctrFindActiveSubstanceSynonyms(activessubstance = "", verbose = FALSE)
```

### Arguments

`activessubstance`

An active substance, in an atomic character vector

`verbose`

Print number of studies found in CTGOV2 for 'activessubstance'

### Value

A character vector of the active substance (input parameter) and synonyms, or NULL if active substance was not found and may be invalid

**Examples**

```
## Not run:

ctrFindActiveSubstanceSynonyms(activessubstance = "imatinib")
# [1] "imatinib"                "Carcemia"                "Cemivil"
# [4] "CGP 57148"              "CGP-57148B"             "CGP57148B"
# [7] "Gleevac"                "gleevec"                 "Gleevec (Imatinib Mesylate)"
# [10] "Glevec"                 "glivec"                 "Imatinib"
# [13] "imatinib mesylate"     "Imatinib-AFT"           "IND # 55666"
# [16] "NSC #716051"           "NSC-716051"            "QTI571"
# [19] "STI571"                "STI 571"               "STI-571"
# [22] "STI571"                "tyrosine kinase inhibitors"

## End(Not run)
```

---

ctrGetQueryUrl	<i>Get query details</i>
----------------	--------------------------

---

**Description**

Extracts query parameters and register name from parameter 'url' or from the clipboard, into which the URL of a register search was copied.

**Usage**

```
ctrGetQueryUrl(url = "", register = "")
```

**Arguments**

url	URL such as from the browser address bar. If not specified, clipboard contents will be checked for a suitable URL. For automatically copying the user's query of a register in a web browser to the clipboard, see <a href="#">here</a> . Can also contain a query term such as from <code>dbQueryHistory()["query-term"]</code> . Can also be an identifier of a trial, which based on its format will indicate to which register it relates.
register	Optional name of register (one of "EUCTR", "CTGOV2" "ISRCTN" or "CTIS") in case 'url' is a query term but not a full URL

**Value**

A data frame (or tibble, if tibble is loaded) with column names 'query-term' and 'query-register'. The data frame (or tibble) can be passed as such as parameter 'query-term' to [ctrLoadQueryIntoDb](#) and as parameter 'url' to [ctrOpenSearchPagesInBrowser](#).

## Examples

```
# user copied into the clipboard the URL from
# the address bar of the browser that shows results
# from a query in one of the trial registers
if (interactive()) try(ctrGetQueryUrl(), silent = TRUE)

# extract query parameters from search result URL
# (URL was cut for the purpose of formatting only)
ctrGetQueryUrl(
  url = paste0(
    "https://classic.clinicaltrials.gov/ct2/results?",
    "cond=&term=AREA%5BMaximumAge%5D+RANGE%5B0+days%2C+28+days%5D",
    "&type=Intr&rslt=&age_v=&gndr=&intr=Drugs%2C+Investigational",
    "&titles=&outc=&spons=&lead=&id=&cntry=&state=&city=&dist=",
    "&locn=&phase=2&rsub=&strd_s=01%2F01%2F2015&strd_e=01%2F01%2F2016",
    "&prcd_s=&prcd_e=&sfpd_s=&sfpd_e=&rfpd_s=&rfpd_e=&lupd_s=&lupd_e=&sort="
  )
)

# other examples
ctrGetQueryUrl("https://www.clinicaltrialsregister.eu/ctr-search/trial/2007-000371-42/results")
ctrGetQueryUrl("https://euclinicaltrials.eu/ctis-public/view/2022-500041-24-00")
ctrGetQueryUrl("https://classic.clinicaltrials.gov/ct2/show/NCT01492673?cond=neuroblastoma")
ctrGetQueryUrl("https://clinicaltrials.gov/ct2/show/NCT01492673?cond=neuroblastoma")
ctrGetQueryUrl("https://clinicaltrials.gov/study/NCT01467986?aggFilters=ages:child")
ctrGetQueryUrl("https://www.isrctn.com/ISRCTN70039829")

# using identifiers of single trials
ctrGetQueryUrl("70039829")
ctrGetQueryUrl("ISRCTN70039829")
ctrGetQueryUrl("NCT00617929")
ctrGetQueryUrl("2022-501142-30-00")
ctrGetQueryUrl("2012-003632-23")
```

---

ctrLoadQueryIntoDb      *Load and store register trial information*

---

## Description

Retrieves information on clinical trials from registers and stores it in a collection in a database. Main function of [ctrdata](#) for accessing registers. A collection can store trial information from different queries or different registers. Query details are stored in the collection and can be accessed using [dbQueryHistory](#). A previous query can be re-run, which replaces or adds trial records while keeping any user annotations of trial records.

## Usage

```
ctrLoadQueryIntoDb(
```

```

queryterm = NULL,
register = "",
querytoupdate = NULL,
forcetoupdate = FALSE,
euctrresults = FALSE,
euctrresultshistory = FALSE,
ctgov2history = FALSE,
documents.path = NULL,
documents.regexp = "prot|sample|statist|sap_|p1ar|p2ars|icf|ctalett|lay|^[0-9]+ ",
annotation.text = "",
annotation.mode = "append",
only.count = FALSE,
con = NULL,
verbose = FALSE,
...
)

```

### Arguments

queryterm	Either a string with the full URL of a search query in a register, or the data frame returned by the <a href="#">ctrGetQueryUrl</a> or the <a href="#">dbQueryHistory</a> functions, or, together with parameter register, a string with query elements of a search URL. The query details are recorded in the collection for later use to update records. For "CTIS", queryterm can be an empty string to obtain all trial records. For automatically copying the user's query of a register in a web browser to the clipboard, see <a href="#">here</a>
register	String with abbreviation of register to query, either "EUCTR", "CTGOV2", "IS-RCTN" or "CTIS". Not needed if queryterm provides a full URL to query results or provides a single trial identifier.
querytoupdate	Either the word "last", or the row number of a query in the data frame returned by <a href="#">dbQueryHistory</a> that should be run to retrieve any new or update trial records since this query was run the last time. This parameter takes precedence over queryterm. For "EUCTR", updates are available only for the last seven days; the query is run again if more time has passed since it was run last. Does not work with "CTIS" at this time.
forcetoupdate	If TRUE, run again the query given in querytoupdate, irrespective of when it was run last. Default is FALSE.
euctrresults	If TRUE, also download available results when retrieving and loading trials from EUCTR. This slows down this function. (For "CTGOV2" and "CTIS", available results are always retrieved and loaded into the collection.)
euctrresultshistory	If TRUE, also download available history of results publication in "EUCTR." This is quite time-consuming. Default is FALSE.
ctgov2history	For trials from CTGOV2, retrieve historic versions of the record. Default is FALSE, because this is a time-consuming operation. Use n for n from all versions (recommended), 1 for the first (original) version, -1 for the last-but-one version, "n:m" for the nth to the mth versions, or TRUE for all versions of the trial record



to be retrieved. Note that for register CTIS, historic versions were available in the ‘applications’ field only before the register’s relaunch on 2024-06-17.

<code>documents.path</code>	If this is a relative or absolute path to a directory that exists or can be created, save any documents into it that are directly available from the register ("EUCTR", "CTGOV2", "ISRCTN", "CTIS") such as PDFs on results, analysis plans, spreadsheets, patient information sheets, assessments or product information. Default is NULL, which disables saving documents. For "EUCTR", sets <code>euctrresults = TRUE</code> since documents are available only with results.
<code>documents.regexp</code>	Regular expression, case insensitive, to select documents by filename, if saving documents is requested (see <code>documents.path</code> ). If set to NULL, empty placeholder files are saved for every document that could be saved, which is useful to get an overview on the number and types of documents available for download. Default is <code>"prot sample statist sap_ p1ar p2ars icf ctalett lay ^[0-9]+"</code> . Used with "CTGOV2", "ISRCTN" and "CTIS" (for "EUCTR", all documents are downloaded since they are few and have non-canonical filenames.)
<code>annotation.text</code>	Text to be including into the field "annotation" in the records retrieved with the query that is to be loaded into the collection. The contents of the field "annotation" for a trial record are preserved e.g. when running this function again and loading a record of a with an annotation, see parameter <code>annotation.mode</code> .
<code>annotation.mode</code>	One of "append" (default), "prepend" or "replace" for new <code>annotation.text</code> with respect to any existing annotation for the records retrieved with the query that is to be loaded into the collection.
<code>only.count</code>	Set to TRUE to return only the number of trial records found in the register for the query. Does not load trial information into the database. Default is FALSE.
<code>con</code>	A database connection object, created with <code>nodbi</code> . See section ‘1 - Database connection’ in <code>ctrdata</code> .
<code>verbose</code>	Printing additional information if set to TRUE; default is FALSE.
<code>...</code>	Do not use (capture deprecated parameters).

### Value

A list with elements ‘n’ (number of trial records newly imported or updated), ‘success’ (a vector of `_id`’s of successfully loaded records), ‘failed’ (a vector of identifiers of records that failed to load) and ‘queryterm’ (the query term used). The returned list has several attributes (including database and collection name, as well as the query history of this database collection) to facilitate documentation.

### Examples

```
## Not run:

dbc <- nodbi::src_sqlite(collection = "my_collection")

# Retrieve protocol- and results-related information
```

```

# on two specific trials identified by their EU number
ctrLoadQueryIntoDb(
  queryterm = "2005-001267-63+OR+2008-003606-33",
  register = "EUCTR",
  euctrresults = TRUE,
  con = dbc
)

# Count ongoing interventional cancer trials involving children
# Note this query is a classical CTGOV query and is translated
# to a corresponding query for the current CTGOV2 webinterface
ctrLoadQueryIntoDb(
  queryterm = "cond=cancer&recr=Open&type=Intr&age=0",
  register = "CTGOV",
  only.count = TRUE,
  con = dbc
)

# Retrieve all information on more than 40 trials
# that are labelled as phase 3 and that mention
# either neuroblastoma or lymphoma from ISRCTN,
# into the same collection as used before
ctrLoadQueryIntoDb(
  queryterm = paste0(
    "https://www.isrctn.com/search?",
    "q=neuroblastoma+OR+lymphoma&filters=phase%3APhase+III"),
  con = dbc
)

# Retrieve information trials in CTIS mentioning neonates
ctrLoadQueryIntoDb(
  queryterm = paste0("https://euclinicaltrials.eu/ctis-public/",
    "search#searchCriteria={%22containAll%22:%22%22,",
    "%22containAny%22:%22neonates%22,%22containNot%22:%22%22}"),
  con = dbc
)

## End(Not run)

```

---

ctrOpenSearchPagesInBrowser

*Open register to show query results or search page*

---

### **Description**

Open advanced search pages of register(s), or execute search in browser

### **Usage**

```
ctrOpenSearchPagesInBrowser(url = "", register = "", copyright = FALSE)
```

**Arguments**

url	of search results page to show in the browser. To open the browser with a previous search, the output of <a href="#">ctrGetQueryUrl</a> or <a href="#">dbQueryHistory</a> can be used. Can be left as empty string (default) to open the advanced search page of register.
register	Register(s) to open, "EUCTR", "CTGOV2", "ISRCTN" or "CTIS". Default is empty string, and this opens the advanced search page of the register(s).
copyright	(Optional) If set to TRUE, opens only the copyright pages of all registers.

**Value**

(String) Full URL corresponding to the shortened url in conjunction with register if any, or invisibly TRUE if no url is specified.

**Examples**

```
# Open all and check copyrights before using registers
ctrOpenSearchPagesInBrowser(copyright = TRUE)

# Open specific register advanced search page
ctrOpenSearchPagesInBrowser(register = "CTGOV2")
ctrOpenSearchPagesInBrowser(register = "CTIS")
ctrOpenSearchPagesInBrowser(register = "EUCTR")
ctrOpenSearchPagesInBrowser(register = "ISRCTN")

# Open all queries that were loaded into demo collection
dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

dbh <- dbQueryHistory(con = dbc)

for (r in seq_len(nrow(dbh))) {
  ctrOpenSearchPagesInBrowser(dbh[r, ])
}
```

---

ctrShowOneTrial      *Show full structure and all data of a trial*

---

**Description**

If used interactively, the function shows a widget of all data in the trial as a tree of field names and values. The widget opens in the default browser. Fields names and values can be search and selected. Selected fields can be copied to the clipboard for use with function [dbGetFieldsIntoDf](#). The trial is retrieved with [ctrLoadQueryIntoDb](#) if no database con is provided or if the trial is not in database con.

## Usage

```
ctrShowOneTrial(identifier = NULL, con = NULL)
```

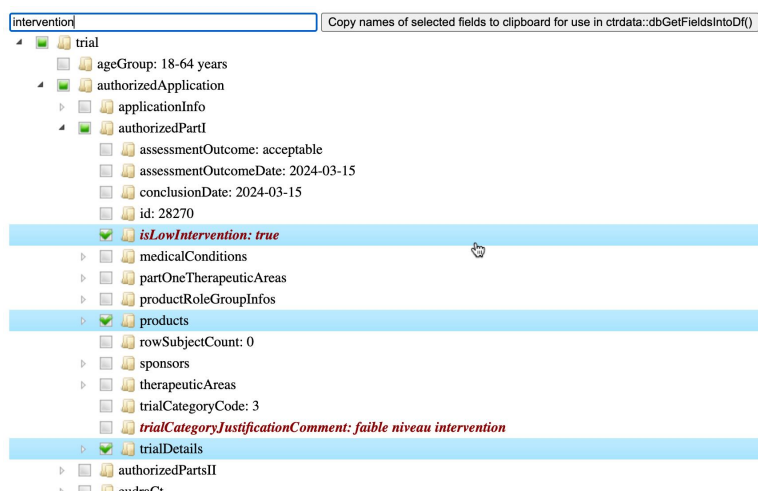
## Arguments

**identifier** A trial identifier string

**con** A database connection object, created with `nodbi`. See section ‘1 - Database connection’ in [ctrdata](#).

## Details

This is the widget for CTIS trial 2022-501142-30-00:



## Value

Invisibly, the trial data for constructing an HTML widget.

## Examples

```
dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

# all such identifiers work
id <- "2014-003556-31"
id <- "2014-003556-31-SE"
id <- "76463425"
id <- "ISRCTN76463425"
id <- "NCT03431558"
id <- "2022-501142-30-00"

# the id also works with
```

```
# ctrGetQueryUrl(url = id) and
# ctrLoadQueryIntoDb(queryterm = id, ...)

# show widget for user to explore and search content as well as to
# select fields of interest and to click on "Copy names of selected
# fields to clipboard..." to use them with dbGetFieldsIntoDf()
ctrShowOneTrial(identifier = id, con = dbc)

# get sample of identifiers of trials in database
sample(dbFindIdsUniqueTrials(con = dbc), 5L)
```

---

dbFindFields

*Find names of fields in the database collection*


---

## Description

Given part of the name of a field of interest to the user, this function returns the full field names used in records that were previously loaded into a collection (using [ctrLoadQueryIntoDb](#)). Only names of fields that have a value in the collection can be returned. Set `sample = FALSE` to force screening all records in the collection for field names, see below. See [ctrShowOneTrial](#) to interactively find fields.

## Usage

```
dbFindFields(namepart = ".*", con, sample = TRUE, verbose = FALSE)
```

## Arguments

namepart	A character string (can be a regular expression, including Perl-style) to be searched among all field names (keys) in the collection, case-insensitive. The default <code>".*"</code> lists all fields.
con	A database connection object, created with <code>nodbi</code> . See section '1 - Database connection' in <a href="#">ctrdata</a> .
sample	If <code>TRUE</code> (default), uses a sample of only 5 trial records per register to identify fields, to rapidly return a possibly incomplete set of field names. If <code>FALSE</code> , uses all trial records in the collection, which will take more time with more trials but ensures to return all names of all fields in the collection.
verbose	If <code>TRUE</code> , prints additional information (default <code>FALSE</code> ).

## Details

The full names of child fields are returned in dot notation (e.g., `clinical_results.outcome_list.outcome.measure.classification`). In addition, names of parent fields (e.g., `clinical_results`) are returned. Data in parent fields is typically complex (nested), see [dfTrials2Long](#) for easily handling it. For field definitions of the registers, see "Definition" in [ctrdata-registers](#). Note: When `dbFindFields` is first called after [ctrLoadQueryIntoDb](#), it will take a moment.

**Value**

Vector of strings with full names of field(s) found, ordered by register and alphabet, see examples. Names of the vector are the names of the register holding the respective fields. The field names can be fed into `dbGetFieldsIntoDf` to extract the data for the field(s) from the collection into a data frame.

**Examples**

```
dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

dbFindFields(namepart = "date", con = dbc)[1:5]

# view all 3350+ fields from all registers:

allFields <- dbFindFields(con = dbc, sample = FALSE)

if (interactive()) View(data.frame(
  register = names(allFields),
  field = allFields))
```

---

`dbFindIdsUniqueTrials` *Get identifiers of deduplicated trial records*

---

**Description**

Records for a clinical trial can be loaded from more than one register into a collection. This function returns deduplicated identifiers for all trials in the collection, respecting the register(s) preferred by the user. All registers are recording identifiers also from other registers, which are used by this function to provide a vector of identifiers of deduplicated trials.

**Usage**

```
dbFindIdsUniqueTrials(
  preferregister = c("EUCTR", "CTGOV", "CTGOV2", "ISRCTN", "CTIS"),
  prefermemberstate = "DE",
  include3rdcountrytrials = TRUE,
  con,
  verbose = FALSE
)
```

**Arguments**

- `preferregister` A vector of the order of preference for registers from which to generate unique `_id`'s, default `c("EUCTR", "CTGOV", "CTGOV2", "ISRCTN", "CTIS")`
- `prefermemberstate`  
Code of single EU Member State for which records should returned. If not available, a record for DE or lacking this, any random Member State's record for the trial will be returned. For a list of codes of EU Member States, please see vector `countriesEUCTR`. Specifying "3RD" will return the Third Country record of trials, where available.
- `include3rdcountrytrials`  
A logical value if trials should be retained that are conducted exclusively in third countries, that is, outside the European Union. Ignored if `prefermemberstate` is set to "3RD".
- `con` A database connection object, created with `nodbi`. See section '1 - Database connection' in [ctrdata](#).
- `verbose` If TRUE, prints out the fields of registers used to find corresponding trial records

**Details**

Note that the content of records may differ between registers (and, for "EUCTR", between records for different Member States). Such differences are not considered by this function.

**Value**

A named vector with strings of keys (field "`_id`") of records in the collection that represent unique trials, where names correspond to the register of the record.

**Examples**

```
dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

dbFindIdsUniqueTrials(con = dbc)[1:10]
```

---

`dbGetFieldsIntoDf`      *Create data frame of specified fields from database collection*

---

**Description**

Fields in the collection are retrieved from all records into a data frame (or tibble). Within a given trial record, a fields can be hierarchical and structured, that is, nested. Th function uses the field names to appropriately type the values that it returns, harmonising original values (e.g. "Information not present in EudraCT" to 'NA', "Yes" to 'TRUE', "false" to 'FALSE', date strings to dates or

time differences, number strings to numbers). The function simplifies the structure of nested data and may concatenate multiple strings in a field using " / " (see example) and may have widened the returned data frame with additional columns that were recursively expanded from simply nested data (e.g., "externalRefs" to columns "externalRefs.doi", "externalRefs.eudraCTNumber" etc.). For an alternative way for handling the complex nested data, see [dfTrials2Long](#) followed by [dfName2Value](#) for extracting the sought variable(s).

### Usage

```
dbGetFieldsIntoDf(fields = "", con, verbose = FALSE, ...)
```

### Arguments

fields	Vector of one or more strings, with names of sought fields. See function <a href="#">dbFindFields</a> for how to find names of fields and <a href="#">ctrShowOneTrial</a> for interactively selecting field names. Dot path notation ("field.subfield") without indices is supported. If compatibility with 'nodbi::src_postgres()' is needed, specify fewer than 50 fields, consider also using parent fields e.g., "a.b" instead of 'c("a.b.c.d", "a.b.c.e")', accessing sought fields with <a href="#">dfTrials2Long</a> followed by <a href="#">dfName2Value</a> or other R functions.
con	A database connection object, created with nodbi. See section '1 - Database connection' in <a href="#">ctrdata</a> .
verbose	Printing additional information if set to TRUE; (default FALSE).
...	Do not use (captures deprecated parameter stopifnodata)

### Value

A data frame (or tibble, if tibble is loaded) with columns corresponding to the sought fields. A column for the records' '\_id' will always be included. The maximum number of rows of the returned data frame is equal to, or less than the number of trial records in the database collection.

### Examples

```
dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

# get fields that are nested within another field
# and can have multiple values with the nested field
dbGetFieldsIntoDf(
  fields = "b1_sponsor.b31_and_b32_status_of_the_sponsor",
  con = dbc)

# fields that are lists of string values are
# returned by concatenating values with a slash
dbGetFieldsIntoDf(
  fields = "keyword",
  con = dbc)
```



---

dbQueryHistory	<i>Show history of queries loaded into a database collection</i>
----------------	--

---

**Description**

Show history of queries loaded into a database collection

**Usage**

```
dbQueryHistory(con, verbose = FALSE)
```

**Arguments**

con	A database connection object, created with <code>nodbi</code> . See section ‘1 - Database connection’ in <a href="#">ctrdata</a> .
verbose	If TRUE, prints additional information (default FALSE).

**Value**

A data frame (or tibble, if `tibble` is loaded) with columns: ‘query-timestamp’, ‘query-register’, ‘query-records’ (note: this is the number of records loaded when last executing [ctrLoadQueryIntoDb](#), not the total record number) and ‘query-term’, with one row for each time that [ctrLoadQueryIntoDb](#) loaded trial records into this collection.

**Examples**

```
dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

dbQueryHistory(con = dbc)
```

---

dfMergeVariablesRelevel
-------------------------

*Merge variables, keeping type, and optionally relevel factors*

---

**Description**

Merge variables in a data frame such as returned by [dbGetFieldsIntoDf](#) into a new variable, and optionally also map its values to new levels.

**Usage**

```
dfMergeVariablesRelevel(df = NULL, colnames = "", levelslist = NULL)
```

**Arguments**

df	A <a href="#">data.frame</a> with the variables (columns) to be merged into one vector.
colnames	A vector of names of columns in 'df' that hold the variables to be merged, or a selection of columns as per <a href="#">select</a> .
levelslist	A names list with one slice each for a new value to be used for a vector of old values (optional).

**Value**

A vector, with the type of the columns to be merged

**Examples**

```

dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

df <- dbGetFieldsIntoDf(
  fields = c("overall_status", "x5_trial_status"),
  con = dbc
)

statusvalues <- list(
  "ongoing" = c("Recruiting", "Active", "Ongoing"),
  "completed" = c("Completed", "Prematurely Ended", "Terminated"),
  "other" = c("Withdrawn", "Suspended", "No longer available")
)

dfMergeVariablesRelevel(
  df = df,
  colnames = 'contains("status")',
  levelslist = statusvalues
)

```

---

dfName2Value

*Get value for variable of interest*


---

**Description**

Get information for variable of interest (e.g., clinical endpoints) from long data frame of protocol- or result-related trial information as returned by [dfTrials2Long](#). Parameters 'valuenam', 'wherenam' and 'wherevalue' are matched using Perl regular expressions and ignoring case.

**Usage**

```
dfName2Value(df, valuenam = "", wherenam = "", wherevalue = "")
```

**Arguments**

df	A data frame (or tibble) with four columns ('_id', 'identifier', 'name', 'value') as returned by <a href="#">dfTrials2Long</a>
valuename	A character string for the name of the field that holds the value of the variable of interest (e.g., a summary measure such as "endPoints.*tendencyValue.value")
wherename	(optional) A character string to identify the variable of interest among those that repeatedly occur in a trial record (e.g., "endPoints.endPoint.title")
wherevalue	(optional) A character string with the value of the variable identified by 'wherename' (e.g., "response")

**Value**

A data frame (or tibble, if tibble is loaded) that includes the values of interest, with columns '\_id', 'identifier', 'name', 'value' and 'where' (with the contents of 'wherevalue' found at 'wherename'). Contents of 'value' are strings unless all its elements are numbers. The 'identifier' is generated by function [dfTrials2Long](#) to identify matching elements, e.g. endpoint descriptions and measurements.

**Examples**

```

dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

dfwide <- dbGetFieldsIntoDf(
  fields = c(
    ## ctgov - typical results fields
    # "clinical_results.baseline.analyzed_list.analyzed.count_list.count",
    # "clinical_results.baseline.group_list.group",
    # "clinical_results.baseline.analyzed_list.analyzed.units",
    "clinical_results.outcome_list.outcome",
    "study_design_info.allocation",
    ## eucr - typical results fields
    # "trialInformation.fullTitle",
    # "baselineCharacteristics.baselineReportingGroups.baselineReportingGroup",
    # "trialChanges.hasGlobalInterruptions",
    # "subjectAnalysisSets",
    # "adverseEvents.seriousAdverseEvents.seriousAdverseEvent",
    "endPoints.endPoint",
    "subjectDisposition.recruitmentDetails"
  ), con = dbc
)

dflong <- dfTrials2Long(df = dfwide)

## get values for the endpoint 'response'
dfName2Value(
  df = dflong,
  valuename = paste0(
    "clinical_results.*measurement.value|",

```

```

      "clinical_results.*outcome.measure.units|",
      "endPoints.endPoint.*tendencyValue.value|",
      "endPoints.endPoint.unit"
    ),
    where_name = paste0(
      "clinical_results.*outcome.measure.title|",
      "endPoints.endPoint.title"
    ),
    where_value = "response"
  )
)

```

---

dfTrials2Long

---

*Convert data frame with trial records into long format*


---

## Description

The function works with protocol- and results- related information. It converts lists and other values that are in a data frame returned by [dbGetFieldsIntoDf](#) into individual rows of a long data frame. From the resulting long data frame, values of interest can be selected using [dfName2Value](#). The function is particularly useful for fields with complex content, such as node field "clinical\_results" from EUCR, for which [dbGetFieldsIntoDf](#) returns as a multiply nested list and for which this function then converts every observation of every (leaf) field into a row of its own.

## Usage

```
dfTrials2Long(df)
```

## Arguments

df                    Data frame (or tibble) with columns including the trial identifier (`_id`) and one or more variables as obtained from [dbGetFieldsIntoDf](#)

## Value

A data frame (or tibble, if tibble is loaded) with the four columns: `'_id'`, `'identifier'`, `'name'`, `'value'`

## Examples

```

dbc <- nodbi::src_sqlite(
  dbname = system.file("extdata", "demo.sqlite", package = "ctrdata"),
  collection = "my_trials",
  RSQLite::SQLITE_RO)

dfwide <- dbGetFieldsIntoDf(
  fields = "clinical_results.participant_flow",
  con = dbc)

```

```
dfTrials2Long(df = dfwide)
```

# Index

## \* data

ctrdata-registers, 3

## \* package

ctrdata, 2

ctrdata, 2, 3, 7, 9, 12, 13, 15–17

ctrdata-package (ctrdata), 2

ctrdata-registers, 3, 3, 13

ctrFindActiveSubstanceSynonyms, 5

ctrGetQueryUrl, 6, 8, 11

ctrLoadQueryIntoDb, 3, 5, 6, 7, 11, 13, 17

ctrOpenSearchPagesInBrowser, 3, 6, 10

ctrShowOneTrial, 3, 11, 13, 16

data.frame, 18

dbFindFields, 3, 13, 16

dbFindIdsUniqueTrials, 3, 14

dbGetFieldsIntoDf, 3, 5, 11, 14, 15, 17, 20

dbQueryHistory, 6–8, 11, 17

dfMergeVariablesRelevel, 17

dfName2Value, 3, 16, 18, 20

dfTrials2Long, 3, 13, 16, 18, 19, 20

nodbi::src\_duckdb, 3

nodbi::src\_mongo, 2

nodbi::src\_postgres, 3

nodbi::src\_sqlite, 3

select, 18