

Package: tantivyr (via r-universe)

July 16, 2026

Title Fast Full-Text Search for R with 'Tantivy'

Version 0.1.0

Description Index data frames and document collections and run fast full-text search entirely on your machine. 'tantivyr' wraps the 'Tantivy' 'Rust' search engine (a 'Lucene'-inspired library) to provide 'BM25' ranking, structured filters, snippet highlighting and incremental updates over an on-disk or in-memory index. First-class support is provided for stemming and stop words in Portuguese and English, making it well suited to public documents, news clippings, extracted 'PDF' text, transcripts and legal acts.

License MIT + file LICENSE

URL <https://strategicprojects.github.io/tantivyr/>,
<https://github.com/StrategicProjects/tantivyr>

BugReports <https://github.com/StrategicProjects/tantivyr/issues>

Encoding UTF-8

Roxygen list(markdown = TRUE)

SystemRequirements Cargo (Rust's package manager), rustc

Depends R (>= 4.2)

Imports cli, rlang (>= 1.1.0), tibble, tidyselect

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

Config/testthat/edition 3

Config/rextendr/version 0.5.0

VignetteBuilder knitr

Biarch true

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libclang-dev

Repository <https://strategicprojects.r-universe.dev>

Date/Publication 2026-07-04 12:25:58 UTC

RemoteUrl <https://github.com/strategicprojects/tantivyr>

RemoteRef HEAD

RemoteSha 2c6f30ae6ccae2b2d7219b10a758d4bd63518ceb

Contents

tantivy_version	2
tnt_add	3
tnt_commit	3
tnt_count	4
tnt_delete	5
tnt_field	6
tnt_index	7
tnt_index_df	8
tnt_index_info	9
tnt_num_docs	10
tnt_schema	10
tnt_search	11
tnt_stemmers	12
tnt_update	13
Index	14

tantivy_version	<i>Version of the bundled 'Tantivy' engine</i>
-----------------	--

Description

Report the version string of the 'Tantivy' 'Rust' search engine that this package was built against.

Usage

`tantivy_version()`

Value

A length-one character vector giving the version of the bundled 'Tantivy' 'Rust' search engine (for example `"0.26.0"`).

tnt_add	<i>Add documents to an index</i>
---------	----------------------------------

Description

Adds the rows of data as documents. Only columns whose names match a schema field are used; other columns are ignored. Additions become searchable after [tnt_commit\(\)](#).

Usage

```
tnt_add(idx, data)
```

Arguments

idx	A <code>tnt_index</code> .
data	A data frame whose columns map to schema fields by name.

Value

The `tnt_index`, invisibly (so calls can be piped).

See Also

[tnt_commit\(\)](#), [tnt_update\(\)](#)

Examples

```
sch <- tnt_schema(id = tnt_i64(), body = tnt_text(stemmer = "english"))
idx <- tnt_index(schema = sch)
df <- data.frame(id = 1:2, body = c("the quick fox", "lazy dogs sleep"))
idx |> tnt_add(df) |> tnt_commit()
tnt_num_docs(idx)
```

tnt_commit	<i>Commit pending changes</i>
------------	-------------------------------

Description

Flushes buffered additions and deletions to the index and refreshes the reader so they become visible to [tnt_search\(\)](#).

Usage

```
tnt_commit(idx)
```

Arguments

idx A tnt_index.

Value

The tnt_index, invisibly.

Examples

```
sch <- tnt_schema(body = tnt_text(stemmer = "english"))
idx <- tnt_index(schema = sch)
tnt_add(idx, data.frame(body = "hello")) |> tnt_commit()
```

tnt_count	<i>Count matching documents</i>
-----------	---------------------------------

Description

Returns the total number of documents matching a query and optional filter, ignoring any result limit.

Usage

```
tnt_count(idx, query = "", fields = NULL, filter = NULL)
```

Arguments

idx A tnt_index.

query A query string in tantivy's **query syntax**. The empty string "" matches all documents (useful with filter).

fields **<tidy-select>** Text fields searched for bare (unqualified) query terms. Defaults to all indexed text fields.

filter Either a tantivy query string, or a comparison expression such as year >= 2020 & source == "globo". Supported operators: ==, %in%, >, >=, <, <=, combined with & and |.

Value

A single numeric count.

See Also

[tnt_search\(\)](#)

Examples

```
idx <- tnt_index_df(
  data.frame(t = c("apple pie", "apple tart", "banana bread")),
  text = t, stemmer = "english"
)
tnt_count(idx, "apple")
```

tnt_delete

*Delete documents by field value***Description**

Marks for deletion every document whose field equals the given value(s), using a `field == value` expression. Deletions take effect after `tnt_commit()`. For reliable deletion the field should be an exact field: a numeric/date field, or a text field created with `stemmer = "raw"`.

Usage

```
tnt_delete(idx, condition)
```

Arguments

idx	A <code>tnt_index</code> .
condition	An expression of the form <code>field == value</code> . <code>value</code> may be a vector, deleting all matching documents.

Value

The `tnt_index`, invisibly.

See Also

[tnt_update\(\)](#)

Examples

```
sch <- tnt_schema(id = tnt_i64(), body = tnt_text(stemmer = "english"))
idx <- tnt_index(schema = sch)
tnt_add(idx, data.frame(id = 1:3, body = c("a", "b", "c"))) |> tnt_commit()
tnt_delete(idx, id == 2) |> tnt_commit()
tnt_num_docs(idx)
```

tnt_field	<i>Define schema fields</i>
-----------	-----------------------------

Description

These constructors describe a single field in a [tnt_schema\(\)](#). Each returns a lightweight `tnt_field` object.

Usage

```
tnt_text(
    stored = TRUE,
    indexed = TRUE,
    fast = FALSE,
    stemmer = "none",
    stopwords = FALSE
)

tnt_i64(stored = TRUE, indexed = TRUE, fast = FALSE)

tnt_u64(stored = TRUE, indexed = TRUE, fast = FALSE)

tnt_f64(stored = TRUE, indexed = TRUE, fast = FALSE)

tnt_bool(stored = TRUE, indexed = TRUE, fast = FALSE)

tnt_date(stored = TRUE, indexed = TRUE, fast = FALSE)

tnt_json(stored = TRUE, indexed = TRUE)
```

Arguments

stored	Logical. Keep the original value so it is returned by tnt_search() . Defaults to TRUE.
indexed	Logical. Index the field so it can be searched or filtered. Defaults to TRUE.
fast	Logical. Build a columnar "fast field" enabling fast filtering and ordering (required by <code>order_by</code> in tnt_search()). Defaults to FALSE, except <code>tnt_text(fast = ...)</code> which has no fast field by default.
stemmer	Stemming/tokenization for a text field. One of "none" (plain tokenization, lower-cased), "raw" (exact, untokenized — ideal for ids and exact filters), or a Snowball language such as "portuguese" or "english". See tnt_stemmers() .
stopwords	Logical. Remove stop words for the chosen language. Bundled for Portuguese and English. Defaults to FALSE.

Value

A `tnt_field` object.

See Also[tnt_schema\(\)](#)**Examples**

```
tnt_schema(
  id    = tnt_i64(),
  title = tnt_text(stemmer = "portuguese", stopwords = TRUE),
  body  = tnt_text(stemmer = "portuguese"),
  date  = tnt_date()
)
```

tnt_index

*Create or open a search index***Description**

Opens an existing on-disk index, or creates a new one from a [tnt_schema\(\)](#). With path = NULL an in-memory index is created (useful for tests and transient work).

Usage

```
tnt_index(path = NULL, schema = NULL, overwrite = FALSE, heap_mb = 128)
```

Arguments

path	Directory for an on-disk index, or NULL for an in-memory index. If the directory already contains an index it is opened (unless overwrite = TRUE).
schema	A tnt_schema() . Required when creating a new index; ignored when opening an existing one.
overwrite	Logical. If TRUE, an existing index at path is deleted and recreated from schema. Defaults to FALSE.
heap_mb	Indexing memory budget in MB (minimum 15). Defaults to 128.

Value

A tnt_index object.

See Also

[tnt_index_df\(\)](#), [tnt_add\(\)](#), [tnt_search\(\)](#)

Examples

```
sch <- tnt_schema(
  id   = tnt_i64(),
  title = tnt_text(stemmer = "english"),
  body  = tnt_text(stemmer = "english")
)
idx <- tnt_index(schema = sch) # in-memory
idx
```

tnt_index_df

Index a data frame in one call

Description

A convenience wrapper that infers a schema from data, creates an index, adds every row and commits. Text columns are made searchable; filter columns are indexed for filtering and ordering; all other columns are stored so they are returned by [tnt_search\(\)](#).

Usage

```
tnt_index_df(
  data,
  text,
  filters = NULL,
  stemmer = "none",
  stopwords = FALSE,
  stored = TRUE,
  path = NULL,
  overwrite = FALSE,
  heap_mb = 128
)
```

Arguments

data	A data frame.
text	< tidy-select > Columns to index as full-text fields.
filters	< tidy-select > Columns to index for filtering/ordering (their type is inferred). Optional.
stemmer, stopwords	Stemming and stop-word options applied to all text columns. See tnt_text() .
stored	Logical. Store text columns so they are returned by searches.
path, overwrite, heap_mb	Passed to tnt_index() .

Value

A committed tnt_index object.

See Also

[tnt_index\(\)](#), [tnt_search\(\)](#)

Examples

```
df <- data.frame(
  id = 1:2,
  title = c("Orçamento público aprovado", "Reforma tributária avança"),
  year = c(2023L, 2024L)
)
idx <- tnt_index_df(df, text = title, filters = year, stemmer = "portuguese")
tnt_search(idx, "orcamento")
```

tnt_index_info	<i>Index schema as a tibble</i>
----------------	---------------------------------

Description

Index schema as a tibble

Usage

```
tnt_index_info(idx)
```

Arguments

idx A tnt_index.

Value

A tibble with one row per field: name, kind, stored, indexed, tokenizer.

Examples

```
idx <- tnt_index_df(data.frame(t = "hi"), text = t)
tnt_index_info(idx)
```

tnt_num_docs	<i>Number of searchable documents</i>
--------------	---------------------------------------

Description

Number of searchable documents

Usage

```
tnt_num_docs(idx)
```

Arguments

idx A tnt_index.

Value

A single numeric value (committed document count).

Examples

```
idx <- tnt_index_df(data.frame(t = "hello world"), text = t)
tnt_num_docs(idx)
```

tnt_schema	<i>Create a search schema</i>
------------	-------------------------------

Description

Combine named [tnt_field](#) definitions into a schema that can be passed to [tnt_index\(\)](#).

Usage

```
tnt_schema(...)
```

Arguments

... Named field definitions, e.g. title = tnt_text().

Value

A tnt_schema object (a named list of tnt_fields).

See Also

[tnt_field](#), [tnt_index\(\)](#)

Examples

```
tnt_schema(
  id    = tnt_i64(),
  title = tnt_text(stemmer = "english"),
  body  = tnt_text(stemmer = "english")
)
```

tnt_search

Search an index

Description

Runs a BM25 full-text search and returns the top matches as a tibble. Supports structured filters, result ordering and snippet highlighting.

Usage

```
tnt_search(
  idx,
  query = "",
  limit = 10L,
  fields = NULL,
  filter = NULL,
  highlight = NULL,
  snippet_chars = 150L,
  order_by = NULL,
  desc = TRUE
)
```

Arguments

idx	A tnt_index.
query	A query string in tantivy's query syntax . The empty string "" matches all documents (useful with filter).
limit	Maximum number of results. Defaults to 10.
fields	<tidy-select> Text fields searched for bare (unqualified) query terms. Defaults to all indexed text fields.
filter	Either a tantivy query string, or a comparison expression such as year >= 2020 & source == "globo". Supported operators: ==, %in%, >, >=, <, <=, combined with & and .
highlight	<tidy-select> Stored text fields to return highlighted snippets for. One <field>_snippet column is added per selected field, with matches wrapped in tags.
snippet_chars	Maximum snippet length in characters. Defaults to 150.
order_by	<tidy-select> A single numeric/date <i>fast</i> field to order by instead of BM25 score. When set, score is NA.
desc	Logical. Order descending (the default) when order_by is set.

Value

A tibble with a score column, every stored field, and any requested *_snippet columns.

See Also

[tnt_count\(\)](#), [tnt_index_df\(\)](#)

Examples

```
df <- data.frame(
  id = 1:3,
  title = c("Quick brown fox", "Lazy dog", "Brown bear"),
  year = c(2019L, 2021L, 2023L)
)
idx <- tnt_index_df(df, text = title, filters = year, stemmer = "english")
tnt_search(idx, "brown")
tnt_search(idx, "brown", filter = year >= 2021)
tnt_search(idx, "fox", highlight = title)
```

tnt_stemmers

List supported stemmer languages

Description

List supported stemmer languages

Usage

```
tnt_stemmers()
```

Value

A character vector of language names accepted by the stemmer argument of [tnt_text\(\)](#).

Examples

```
tnt_stemmers()
```

tnt_update	<i>Update documents (delete then re-add)</i>
------------	--

Description

Replaces documents by deleting any existing document that shares a key value (the `by` field) with a row of data, then adding the rows of data. The `by` field must be an exact field (numeric/date, or text with `stemmer = "raw"`). Call `tnt_commit()` afterwards.

Usage

```
tnt_update(idx, data, by)
```

Arguments

<code>idx</code>	A <code>tnt_index</code> .
<code>data</code>	A data frame of replacement documents.
<code>by</code>	<code><tidy-select></code> A single key column present in both data and the schema.

Value

The `tnt_index`, invisibly.

See Also

`tnt_add()`, `tnt_delete()`

Examples

```
sch <- tnt_schema(id = tnt_i64(), body = tnt_text(stemmer = "english"))
idx <- tnt_index(schema = sch)
tnt_add(idx, data.frame(id = 1:2, body = c("old one", "old two"))) |>
  tnt_commit()
tnt_update(idx, data.frame(id = 1L, body = "new one"), by = id) |>
  tnt_commit()
```

Index

`tantivy_version`, 2
`tnt_add`, 3
`tnt_add()`, 7, 13
`tnt_bool(tnt_field)`, 6
`tnt_commit`, 3
`tnt_commit()`, 3, 5, 13
`tnt_count`, 4
`tnt_count()`, 12
`tnt_date(tnt_field)`, 6
`tnt_delete`, 5
`tnt_delete()`, 13
`tnt_f64(tnt_field)`, 6
`tnt_field`, 6, 10
`tnt_i64(tnt_field)`, 6
`tnt_index`, 7
`tnt_index()`, 8–10
`tnt_index_df`, 8
`tnt_index_df()`, 7, 12
`tnt_index_info`, 9
`tnt_json(tnt_field)`, 6
`tnt_num_docs`, 10
`tnt_schema`, 10
`tnt_schema()`, 6, 7
`tnt_search`, 11
`tnt_search()`, 3, 4, 6–9
`tnt_stemmers`, 12
`tnt_stemmers()`, 6
`tnt_text(tnt_field)`, 6
`tnt_text()`, 8, 12
`tnt_u64(tnt_field)`, 6
`tnt_update`, 13
`tnt_update()`, 3, 5